بسم الله الرحمن الرحیم

برنامه‌نویسی مقدماتی

(ویژوال‌بیسیک)

شاخهٔ : کاردانش

زمینهٔ : خدمات

گروه تحصیلی : کامپیوتر

زیرگروه : کامپیوتر

رشتهٔ مهارتی : طراحی صفحات وب، تولید چندرسانه‌ای، تصویرسازی رایانه‌ای، برنامه‌نویسی پایگاه داده

شمارهٔ رشتهٔ مهارتی : ۳۱۲ـ۱۰۱ـ۱۷ـ۳ تا ۳۱۵ـ۱۰۱ـ۱۷ـ۳

کد رایانه‌ای رشتهٔ مهارتی : ۶۱۳۷، ۶۱۳۸، ۶۱۳۹، ۶۱۴۰

نام استاندارد مهارتی مبنا : رایانه‌کار پیشرفته و برنامه‌نویسی ویژوال بیسیک مقدماتی

کد استاندارد متولی : ۸۷ـ۱۵/۱/۲ ف. هـ و ۸۴/۸۰/۱/۳/۱ـ۰

شمارهٔ درس : نظری : ۰۴۶۸، ۰۴۹۰ و عملی : ۰۴۶۹ ، ۰۴۹۱

عنوان و نام پدیدآور	:	برنامه‌نویسی مقدماتی (ویژوال بیسیک)[کتاب‌های درسی] ۶۱۲/۳،نظارت بر تألیف و تصویب محتوا : دفتر تألیف کتاب‌های درسی فنی و حرفه‌ای و کاردانش؛ مؤلف : منصور ولی‌نژاد[برای] وزارت آموزش و پرورش، سازمان پژوهش و برنامه‌ریزی آموزشی.
مشخصات نشر	:	تهران : شرکت چاپ و نشر کتاب‌های درسی ایران، ۱۳۹۲.
مشخصات ظاهری	:	۴۴۲ ص. : مصور، جدول.
شابک	:	۹۷۸ـ۹۶۴ـ۰۵ـ۲۱۸۴ـ۷
وضعیت فهرست‌نویسی	:	فیپا
موضوع	:	۱ـ ویژوال بیسیک (زبان برنامه‌نویسی کامپیوتر) ـ راهنمای آموزشی(متوسطه) ۲ـ ویژوال بیسیک (زبان برنامه‌نویسی کامپیوتر) ـ آزمون‌ها و تمرین‌ها (متوسطه)
شناسه افزوده	:	ولی‌نژاد ، منصور ، ۱۳۴۵ـ. الف ـ سازمان پژوهش و برنامه‌ریزی آموزشی. ب ـ دفتر تألیف کتاب‌های درسی فنی و حرفه‌ای و کاردانش. ج ـ اداره کلّ نظارت بر نشر و توزیع مواد آموزشی.
رده‌بندی کنگره	:	۱۳۹۰ ۴۳ ب ۹ و QA ۷۶/۷۳/
رده‌بندی دیویی	:	۶۱۲/۳ک۳۷۳/
شماره کتاب‌شناسی ملی	:	۲۲۹۲۹۴۱

۱۳۹۲

همکاران محترم و دانش‌آموزان عزیز :

پیشنهادات و نظرات خود را دربارهٔ محتوای این کتاب بـه نشانی تهران ـ صندوق پستی شمارهٔ ۴۸۷۴/۱۵ دفتر تألیف کتاب‌های درسی فنی و حرفه‌ای و کاردانش، ارسال فرمایند.

پیام‌نگار (ایمیل) **tvoccd@roshd.ir**

وب‌گاه (وب‌سایت) **www.tvoccd.medu ir**

وزارت آموزش و پرورش
سازمان پژوهش و برنامه‌ریزی آموزشی

برنامه‌ریزی محتوا و نظارت بر تألیف : **دفتر تألیف کتاب‌های درسی فنی و حرفه‌ای و کاردانش**

نام کتاب : **برنامه‌نویسی مقدماتی (ویژوال بیسیک) ـ ۶۱۲/۳**

مؤلف : **مهندس منصور ولی‌نژاد**

ویراستار ادبی : **شیوا غمگسار**

نظارت بر چاپ و توزیع : **ادارهٔ کلّ نظارت بر نشر و توزیع مواد آموزشی**

تهران : خیابان ایرانشهر شمالی ـ ساختمان شمارهٔ ۴ آموزش و پرورش (شهید موسوی)

تلفن : ۹ـ۸۸۸۳۱۱۶۱، دورنگار : ۸۸۳۰۹۲۶۶، کدپستی : ۱۵۸۴۷۴۷۳۵۹،

وب‌سایت : www.chap.sch ir

محتوای این کتاب در کمیسیون تخصصی رشته کامپیوتر دفتر تألیف کتاب‌های درسی فنی و حرفه‌ای و کاردانش با عضویت : بتول عطاران، محمدرضا شکرریز، محمدرضا یمقانی، سیدحمیدرضا ضیایی، زهرا عسگری، افشین اکبری و سیدسعید میرباقری تأیید شده است.

صفحه‌آرا : **آمنه درویش**

طراح جلد : **بیتا اشرفی‌مقدم**

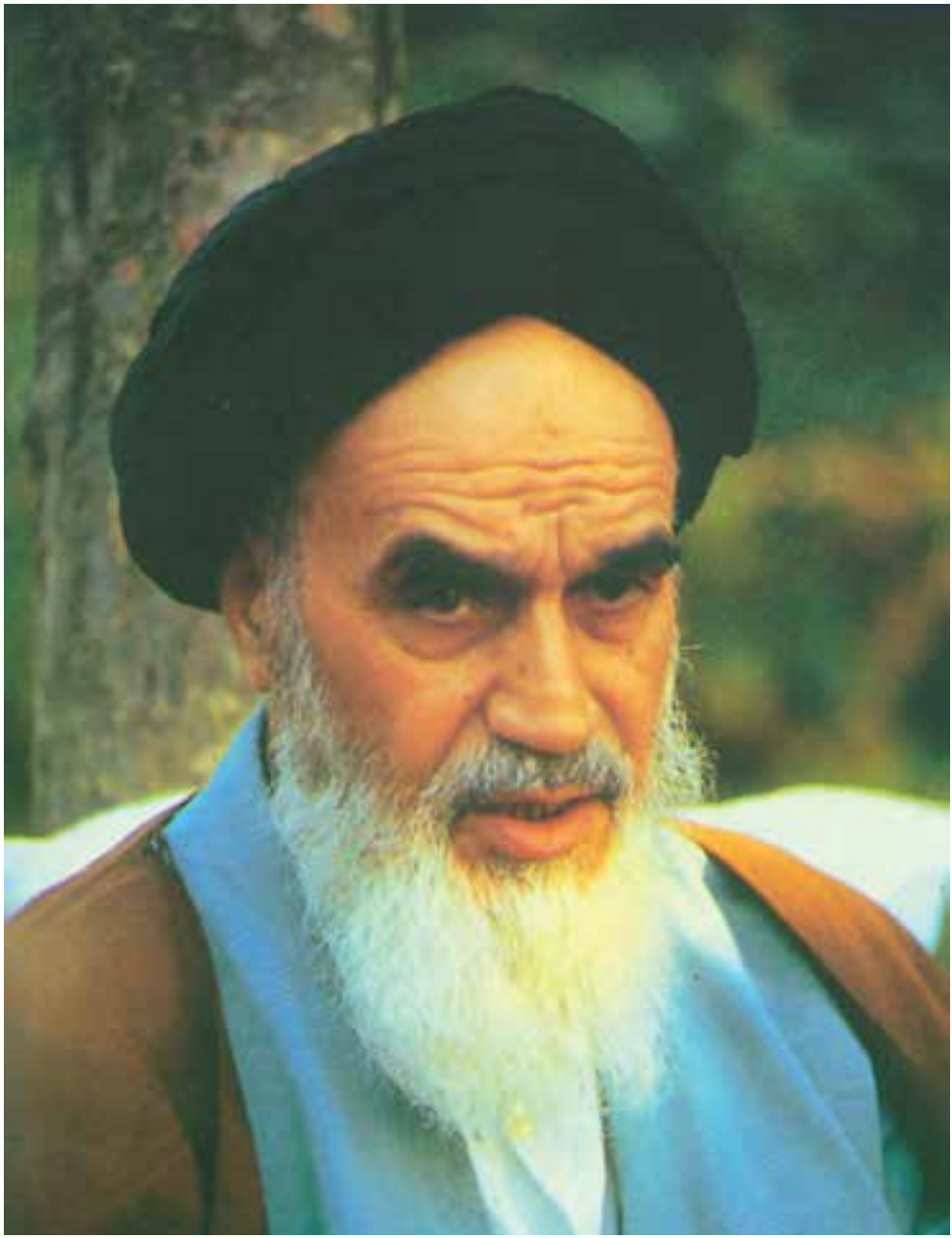
ناشر : **شرکت چاپ و نشر کتاب‌های درسی ایران : تهران ـ کیلومتر ۱۷ جادّهٔ مخصوص کرج ـ خیابان ۶۱ (داروپخش)**

تلفن : ۵ ـ ۴۴۹۸۵۱۶۱، دورنگار : ۴۴۹۸۵۱۶۰، صندوق پستی : ۱۳۹ـ۳۷۵۱۵

چاپخانه : **شرکت افست «سهامی عام»**

سال انتشار و نوبت چاپ : **چاپ اوّل برای سازمان ۱۳۹۲**

شابک ۷ـ۲۱۸۴ـ۰۵ـ۹۶۴ـ۹۷۸ **ISBN 978-964-05-2184-7**



بدانید مادام که در احتیاجات صنایع پیشرفته، دست خود را پیش دیگران دراز کنید و به دریوزگی عمر را بگذارنید، قدرت ابتکار و پیشرفت در اختراعات در شما شکوفا نخواهد شد.

امام خمینی «قدّس سرّه الشّریف»

مجموعه کتاب‌های درسی رشته کامپیوتر شاخه کاردانش
(استاندارد وزارت فرهنگ و ارشاد اسلامی)

رشته تصویرسازی	رشته طراحی صفحات وب	رشته تولید چندرسانه‌ای
مفاهیم پایه فناوری اطلاعات	مفاهیم پایه فناوری اطلاعات	مفاهیم پایه فناوری اطلاعات
سیستم‌عامل مقدماتی	سیستم‌عامل مقدماتی	سیستم‌عامل مقدماتی
اطلاعات و ارتباطات	اطلاعات و ارتباطات	اطلاعات و ارتباطات
واژه‌پرداز Word 2007	واژه‌پرداز Word 2007	واژه‌پرداز Word 2007
صفحه گسترده Excel 2007	صفحه گسترده Excel 2007	صفحه گس‌ترده Excel 2007
ارایه مطالب PowerPoint 2007	ارایه مطالب PowerPoint 2007	ارایه مطالب PowerPoint 2007
نرم‌افزارهای اداری تکمیلی	نرم‌افزارهای اداری تکمیلی	نرم‌افزارهای اداری تکمیلی
بانک اطلاعاتی Access 2007	بانک اطلاعاتی Access 2007	بانک اطلاعاتی Access 2007
سیستم‌عامل پیشرفته	سیستم‌عامل پیشرفته	سیستم‌عامل پیشرفته
برنامه‌نویسی مقدماتی	برنامه‌نویسی مقدماتی	برنامه‌نویسی مقدماتی
طراحی امور گرافیکی با رایانه	طراحی امور گرافیکی با رایانه	طراحی امور گرافیکی با رایانه
کاربر FreeHand	کاربر Flash	کاربر Flash
کاربر CorelDraw	طراحی صفحات وب مقدماتی	کاربر Director
	طراحی صفحات وب پیشرفته	میکس رایانه‌ای

مجموعه کتاب‌های درسی رشته کامپیوتر شاخه کاردانش
(استاندارد وزارت کار و امور اجتماعی)

رشته تصویرسازی	رشته طراحی صفحات وب	رشته تولید چندرسانه‌ای	رشته برنامه‌نویسی پایگاه داده
مفاهیم پایه فناوری اطلاعات	مفاهیم پایه فناوری اطلاعات	مفاهیم پایه فناوری اطلاعات	مفاهیم پایه فناوری اطلاعات
سیستم‌عامل مقدماتی	سیستم‌عامل مقدماتی	سیستم‌عامل مقدماتی	سیستم‌عامل مقدماتی
اطلاعات و ارتباطات	اطلاعات و ارتباطات	اطلاعات و ارتباطات	اطلاعات و ارتباطات
سیستم‌عامل پیشرفته	سیستم‌عامل پیشرفته	سیستم‌عامل پیشرفته	سیستم‌عامل پیشرفته
واژه‌پرداز Word 2007	واژه‌پرداز Word 2007	واژه‌پرداز Word 2007	واژه‌پرداز Word 2007
صفحه گسترده Excel 2007	صفحه گسترده Excel 2007	صفحه گسترده Excel 2007	صفحه گسترده Excel 2007
ارایه مطالب PowerPoint 2007	ارایه مطالب PowerPoint 2007	ارایه مطالب PowerPoint 2007	ارایه مطالب PowerPoint 2007
نرم‌افزارهای اداری تکمیلی	نرم‌افزارهای اداری تکمیلی	نرم‌افزارهای اداری تکمیلی	نرم‌افزارهای اداری تکمیلی
طراح گرافیک رایانه‌ای	طراح گرافیک رایانه‌ای	طراح گرافیک رایانه‌ای	برنامه‌نویسی مقدماتی
شهروند الکترونیکی	نرم‌افزار گرافیکی Flash Mx	شهروند الکترونیکی	برنامه‌نویسی ویژوال‌بیسیک پیشرفته (جلد اول)
نرم‌افزار گرافیکی FreeHand	طراحی مقدماتی صفحات وب	نرم‌افزار گرافیکی Director	برنامه‌نویسی ویژوال‌بیسیک پیشرفته (جلد دوم)
نرم‌افزار گرافیکی CorelDraw	رایانه کار Interdev	تدوین فیلم و صدا SSP	مدیریت پایگاه داده
نرم‌افزار گرافیکی Flash Mx	رایانه کار Dreamweaver	نرم‌افزار گرافیکی Flash Mx	مهارت عمومی برنامه‌نویسی
	رایانه کار CIW	نرم‌افزار گرافیکی Authorware	

فهرست مطالب

مقدمه

یکـــی از عمده‌ترین اهداف طراحی و تولید کامپیوترها انجام عملیات ذخیره‌ســـازی، بازیابی داده‌ها و اطلاعات و انجام انواع محاســـبات به وســـیله آن‌هاســـت. برای تحقق بخشـــیدن به این اهداف وجود دو جزء اصلی یعنی سخت‌افزار و نرم‌افزار الزامی است.

بـــا پیدایش اولین کامپیوترها نیاز به وجود برنامه‌هایی که بتوان با به‌کارگیری آن‌ها کامپیوترها را مورد اســـتفاده قرار داد، احساس شـــد و ازآن‌جا که پردازش در کامپیوترها براساس مبنای باینری یا همان ۰ و ۱ اســـت، اولین نرم‌افزارهایی که توســـط متخصصین طراحی گردید به زبان ماشـــین (۰ و۱) نوشته شد. این روش تولید نرم‌افزار کار دشوار و وقت‌گیری بودکه محدودیت‌های زیادی را دربرداشت، بنابراین متخصصان علوم کامپیوتر تصمیم گرفتند تا نرم‌افزارهایی را تولید کنند که بتوانند با استفاده از آن‌ها هر نوع برنامه‌ای را با سرعت و دقت به زبان ماشین تبدیل نمایند. این امر منجر به تولید شاخه ویژه‌ای از نرم‌افزارها به‌نام زبان‌های برنامه‌نویسی شد. بدین‌ترتیب به موازات رشد و تکامل صنعت سخت‌افزار، زبان‌های برنامه‌نویسی کامپیوتر نیز خط سیر تکاملی خود را از زبان ماشـــین و اســـمبلی به زبان‌های برنامه‌نویسی سطح بالا، ساخت یافته، شئ‌گرا و ویژوال، طی کرده و هر روزه زبان‌های برنامه‌نویســـی کاربردی‌تری را در اختیار برنامه‌نویســـان قرار دادند. در حال‌حاضر محدوده زبان‌های برنامه‌نویسی بسیار گسترده شده و با ظهور سیستم‌عامل‌های ویندوز و رایج‌شدن شبکه‌های کامپیوتری بخصوص اینترنت، این مسأله شدت بیشتری پیدا کرده است.

این پیمانه مهارتی با توجه به نیازهای آموزشـــی در رابطه با روش حل مســـایل و زبان‌های برنامه‌نویسی سطح بالا و مدرن که قابلیت برنامه‌نویسی در محیط سیستم عامل ویندوز را نیز داشته باشند، تألیف شده است. زبان برنامه‌نویسی ویژوال‌بیسیک نسخه ۶ یکی از رایج‌ترین و کارامدترین زبان‌های برنامه‌نویسی در دنیاست که از ویژگی‌های بالایی در برنامه‌نویسی حرفه‌ای برخوردار است.

این پیمانه مهارتی شـــامل چهارده واحدکار اســـت که به صورت خودآموز تهیه شـــده است. در واحدکار اول و دوم دانش‌آموزان با مفاهیم اساســـی درباره الگوریتم و فلوچارت آشـــنا شده و چگونگـــی یافتن راه‌حل مناســـب برای حل مســـایل مختلف را به همراه مثـــال و تمرین‌های متعدد فرامی‌گیرند. از واحد کار ســـوم تا چهاردهم فراگیر می‌تواند مطالب را به صورت تئوری همراه با تصاویر مناســـب مطالعه کرده و ســـپس مطالب مطالعه شده را با تمرین‌های مرتبط و به صورت عملی و مرحله‌ای انجام دهد. به علاوه ســـعی شده است که با ارایه آزمون‌های نظری و عملی در پایان هر واحدکار و یک آزمون نظری و عملی در پایان پیمانه مهارتی، مطالب مجدداً مرور شـــوند. برای بازدهی بالاتر در فراگیری این پیمانه مهارتی تسلط کافی در دروس پیش‌نیاز ماننـــد مبانـــی و فناوری کامپیوتر، سیســـتم عامل مقدماتی و پیشـــرفته (ترجیحاً Windows XP) الزامی است.

مؤلف

Vbbook۶@yahoo.com

پیش‌آزمون

۱ – کدام کلید ترکیبی برای فعال کردن منوی Start به کار می‌رود؟

الف– Alt+F4 ب– Ctrl+Esc

ج– Alt+Esc د– Ctrl+C

۲ – از کــدام برنامه در ویندوز برای اضافه‌کردن یک زبان جدید به مجموعه زبان‌های موجود استفاده می‌شود؟

الف– Keyboard ب– Regional and Language Options

ج– Folder Options د– Regional Options

۳ – کدام کلید ترکیبی برای حرکت کردن بین پنجره‌های باز مناسب است؟

الف– Ctrl+V ب– Alt+Tab

ج– Alt+Esc د– Ctrl+X

واحد کار اول

هدف جزیی

توانایی حل مسایل و طراحی الگوریتم مناسب برای آن‌ها

هدف‌های رفتاری

پس از مطالعه این واحد کار از فراگیر انتظار می‌رود که:

۱- نحوه شناخت و بررسی مسایل مختلف را توضیح دهد.

۲- مفهوم الگوریتم را تعریف کند و ویژگی‌های آن را بیان نماید.

۳- انواع دستورالعمل‌ها را در الگوریتم بیان کند و کاربرد آن‌ها را توضیح دهد.

۴- مفهوم عملگر را بداند و انواع عملگرها را بیان کند.

۵- عملگرهای ریاضی، منطقی و مقایسه‌ای و کاربرد آن‌ها را بیان کند و حق تقدم آن‌ها را نسبت به یکدیگر توضیح دهد.

۶- بتواند انواع الگوریتم‌ها را برای مسایل متفاوت طراحی نماید.

کلیات

روش حل مسایل با استفاده از روش‌ها و تحلیل‌های ریاضی و منطقی، اولین بار به وسیله دانشمند بزرگ ایرانی ـ خوارزمی ـ مورد توجه قرار گرفت، وی علاوه بر مباحث مختلفی که در علوم ریاضی و نجوم طرح کرد، اساس روشی را در حل مسایل بنا نهاد که در آینده مدت‌ها مورد استفاده برنامه‌نویسان کامپیوتر قرار گرفت. نام الگوریتم نیز به افتخار وی و از عبارت «الخوارزمی» گرفته شده است.

در این پیمانه مهارتی با نحوه تجزیه و تحلیل انواع مختلف مسایل و جستجو و طراحی راه حل‌های مناسب برای آن‌ها آشنا خواهید شد و در پایان ارایه راه‌حل به دست آمده با توجه به قواعد علم الگوریتم را می‌آموزید.

۱–۱ شناخت مسایل و ارایه راه‌حل مناسب برای حل آن ها

انسان از آغاز آفرینش تاکنون همواره در مسیر زندگی خود با مشکلات و مسائل مختلفی روبه‌رو بوده و برای حل مشکلات خود، همواره راه‌حل‌های متفاوتی را تجربه کرده است. از مشکلاتی نظیر خوراک، پوشاک، محل زندگی تا حل مسائل علمی، فلسفی، ریاضی و نظایر آن‌ها؛ بنابراین با توجه به نوع مسایل و تجربیات و پیشرفت‌های علمی، سعی در ارائه راه‌حل‌های جدید کرده است. پیدا کردن راه‌حل برای یک مسأله به نوع آن بستگی دارد، بعضی از مسأله‌ها به سادگی قابل حل بوده اما بعضی دیگر هنوز هم به سادگی قابل حل نیستند.

برای حل هرگونه مسأله جدا از نوع آن می‌توان موارد زیر را در نظر گرفت:

۱– شناخت دقیق مسأله

۲– تجزیه و تحلیل مسأله

۳– طراحی راه‌حل

۱–۱–۱ شناخت مسأله

برای شناخت بهتر یک مسأله باید سه عامل مهم را درنظر بگیرید: مقادیر معلوم، خواسته‌های مسأله (مجهولات) و عملیات محاسباتی.

مقادیر معلوم داده‌ها: مقادیری که در اختیار مسأله قرار می‌گیرند و برای رسیدن به هدف موردنظر در مسأله موردنیاز هستند.

ارتباط بین داده‌ها و مجهول‌ها (محاسبات): برای رسیدن به نتایج موردنظر معمولاً لازم است تا عملیاتی را روی مقادیر معلوم انجام دهید؛ بخش عمده‌ای از این عملیات با استفاده از فرمول‌های مختلف انجام می‌شود؛ البته محاسبات می‌توانند با توجه به روابط منطقی که بین مقادیر معلوم و خواسته‌های مسأله وجود دارند، انجام گیرند.

خواسته‌های مسأله (مجهولات): مقادیری هستند که معمولاً در اثر انجام عملیات روی مقادیر معلوم حاصل می‌شوند، البته مجهولات می‌توانند از روابط منطقی که در حل مسأله دخالت می‌کنند نیز به وجود آمده و مورد استفاده قرار گیرند.

به عنوان مثال فرض کنید می‌خواهیم محیط یک دایره به شعاع دلخواه را محاسبه کنیم. برای حل این‌مسأله با روش ارایه شده، ابتدا مقادیر معلوم را مورد توجه قرار می‌دهیم؛ همان‌طورکه می‌دانید برای محاسبه محیط هر دایره باید شعاع آن را در اختیار داشته باشیم، بنابراین شعاع دایره (R) به عنوان تنها داده موردنیاز برای حل مسأله کافی است.

محاسباتی که برای رسیدن به محیط دایره لازم است، در واقع فرمول زیر خواهد بود:

$$P=۲\times ۳/۱۴ \times R$$

با استفاده از این فرمول ها روابط بین داده های ورودی و نتایج موردنظر به خوبی تعیین می شود و بالاخره به عنوان مرحله آخر، خواسته مسأله (مجهول) که همان محیط(P) دایره است به‌دست می‌آید.

تمرین:
مقادیر معلوم، محاسبات و خواسته‌های مسأله (مجهولات) را در محاسبه محیط و مساحت یک مستطیل دلخواه مشخص کنید

۲-۱-۱ تجزیه و تحلیل مسأله

معمولاً جستجوی راه‌حل مناسب برای یک مسأله، به‌سادگی امکان‌پذیر نیست. گاهی اوقات راه‌حل مسأله به‌سادگی قابل تشخیص است، مانند محاسبه مجموع دو عدد یا محاسبه مساحت یک دایره اما در بعضی از مواقع به‌دلیل پیچیده بودن مسأله، لازم است

اســت مسأله اصلی به چند مســأله کوچک‌تر تقسیم شود که به آن‌ها زیرمسأله می‌گویند. ســپس هر زیرمسأله مجدداً بررسی شــود تا در صورت دشواری مجدداً به زیرمسأله‌های کوچک‌تر تقسیم شود و این عمل آن‌قدر ادامه یابد تا زیرمسأله‌ها به ساده‌ترین شکل حل شــوند. در این حالت می‌توان هر زیرمســأله را جداگانه حل کرد و با کنار هم قرار دادن آن‌ها، به حل مسأله اصلی دسترسی پیدا کرد. به عنوان مثال اگر مسأله ساخت یک اتومبیل باشد پیدا کردن راه‌حل برای ساخت آن در مرحله اول، کار دشواری است اما اگر اتومبیل را به بخش‌های مختلف مثل موتور، جعبه‌دنده و اتاق و بدنه تقسیم کنیم و هر یک از این اجزا را مجدداً به کوچک‌ترین جزء، آن‌گاه با طراحی و ساخت اجزای کوچک‌تر می‌توان اجزای بزرگ‌تر را ایجاد کرد و در نهایت با ترکیب اجزای اصلی، طراحی و ســاخت یک اتومبیل امکان‌پذیر خواهد شد.

۳–۱–۱ طراحی راه‌حل

پس از تحلیل مسأله، برای ارائه یک راه‌حل می‌توان به دو روش عمل کرد:

۱– استفاده از تجربیات و راه‌حل‌های موجود که در حل مسائل دیگر به کار گرفته شده‌اند.

۲– اســتفاده از روش‌هــای تفکر منطقی و الگوریتمی که حل مســأله براســاس آن صورت گرفته و به صورت مرحله به مرحله انجام می‌شــود. به عنوان مثال فرض کنید می‌خواهیم تعداد اعداد زوج بین اعداد ۱ تا ۲۰ را محاســبه کنیم، یک روش ســاده این اســت که اعداد ۱ تا ۲۰ را نوشته و تعداد اعداد زوج را بشمارید، اما با استفاده از روش تفکر منطقی می‌توان به این نتیجه رســید که با توجه به این‌که اعداد زوج یکی در میان قرار می‌گیرند با تقســیم عدد ۲۰ بر ۲ به تعداد ۱۰عدد دست می‌یابید که از روش اول هم به همین نتیجه خواهید رسید.

۲–۱ الگوریتم

۱–۲–۱ تعریف الگوریتم

بــه مجموعــه ای از دســتورالعمل‌ها که با زبــان دقیق و قابل فهم بــه همراه جزییات لازم و بــه صــورت مرحلــه بــه مرحله بــه گونه‌ای اجرا شــده که هــدف خاصی (حل مســأله) را دنبال کنند و شــروع و خاتمه آن ها نیز مشــخص باشــد، الگوریتم می‌گویند.

می‌توان الگوریتم را به یک ماشین تشبیه کرد که مقادیر معلوم را دریافت کرده، روی آن‌ها محاسباتی را انجام می‌دهد و در پایان خواسته‌های مسأله (مجهولات) را ارایه می‌کند.

در واقع رابطه نزدیکی بین مفهوم الگوریتم و نحوه کار کامپیوتر در حل مسایل وجود دارد، بنابراین با استفاده از روش الگوریتم می‌توانید حل مسایل را به گونه‌ای طراحی کنید که برای تبدیل به زبان کامپیوتر نیز قابل فهم باشد. به طور معمول با مفهوم الگوریتم آشنا هستید و از آن استفاده می‌کنید، به عنوان مثال وقتی هر روز برای کسب علم و دانش به مدرسه می‌روید، اعمالی را به ترتیب و به صورت دقیق و کامل انجام می‌دهید یعنی ابتدا از خواب بیدار می‌شوید، دست و صورت خود را می‌شویید، صبحانه می‌خورید و سپس لباس مناسب به تن کرده و بعد از برداشتن وسایل و کتاب‌های موردنیاز از خانه خارج می‌شوید، مسافتی را طی کرده و به مدرسه و کلاس خود می‌روید. اگر هر یک از این اعمال را قبل یا بعد از عمل دیگر انجام دهید، هدف موردنظر که شکل درست به مدرسه رفتن است، انجام نمی‌شود.

می‌توان این مراحل را به شکل خلاصه و قابل فهم‌تری بیان کرد:

- بیدارشدن از خواب
- شستن دست و صورت
- خوردن صبحانه
- پوشیدن لباس مدرسه
- برداشتن کتاب‌های درسی و دفتر و قلم
- خروج از خانه
- رفتن به مدرسه و ورود به کلاس درس

تمرین:

مراحل تهیه نان در نانوایی را به صورت مرحله به مرحله بنویسید.

۲-۲-۱ شرایط الگوریتم

پس از آشنایی با مفهوم الگوریتم، لازم است ویژگی های یک الگوریتم را بشناسید.

الف- استفاده از زبان ساده، دقیق و قابل فهم: این ویژگی سبب می‌شود تا در انجام دستورالعمل‌ها همواره یک برداشت یکسان حاصل شود، در غیراین صورت برداشت‌های متفاوت سبب خواهد شد تا دستورالعمل ها نتایج متفاوتی را به وجود آورند. زبان الگوریتم نیز می‌تواند یکی از زبان‌های گفتاری و نوشتاری مانند فارسی، انگلیسی و ... باشد.

ب- استفاده از جزییات کافی: این ویژگی سبب می‌شود تا دستورالعمل‌ها به طور کامل اجرا شوند. وجود موارد نامشخص یا ارایه دستورالعمل ها به صورت کلی و مبهم سبب مخدوش شدن نتایج خواهد شد.

ج- شروع و پایان الگوریتم: در یک الگوریتم باید شروع دستورالعمل‌ها مشخص باشد. هر الگوریتم یک نقطه شروع دارد که به عنوان اولین دستورالعمل از آن استفاده می‌شود، به علاوه پایان الگوریتم نیز باید تعیین شود. یک الگوریتم می‌تواند بیش از یک نقطه پایان داشته باشد.

د- ترتیب انجام دستورالعمل‌ها: یکی از ویژگی‌های مهم یک الگوریتم ترتیب اجرای دستورالعمل‌ها است؛ اگر این کار به درستی انجام نشود، پیش‌بینی نتیجه کار مشخص نخواهد بود. در یک الگوریتم ترتیب انجام عملیات با استفاده از شماره گذاری دستورالعمل‌ها از بالا به پایین انجام می‌شود که البته در صورت نیاز می‌توان ترتیب اجرای دستورالعمل‌ها را نیز تغییر داد. در مباحث بعد به این مسأله خواهیم پرداخت.

نکته: اگر در حل مسایل سه عامل اصلی را به دقت مشخص کنید، طراحی یک الگوریتم کارزیاد دشواری نخواهد بود. اگر مقادیر معلوم، خواسته‌های مسأله، فرمول‌ها و روابط ریاضی و منطقی بین آن‌ها را به درستی تعیین کنید، نوشتن الگوریتم‌های مختلف آسان‌تر خواهد شد.

هـ - جامع بودن: الگوریتم باید به شکلی طراحی شود که با توجه به صورت مسأله و مفروضات آن در تمام حالت‌ها، نتایج مناسب و صحیحی را ارایه کرده و در حالت‌های خاص یا داده‌های ورودی متفاوت، نتایج درستی را ایجاد کند.

نکته: بهتر اســت دریک الگوریتم از دستورات اضافه که سبب افزایش حجم الگوریتم می‌شود، خودداری نمایید چرا که این کار الگوریتم را شلوغ کرده و باعث سردرگمی می‌شود.

۳-۲-۱ انواع دستورالعمل‌ها در الگوریتم

الگوریتم مجموعه‌ای از دستورالعمل‌هاســت. دســتورالعمل‌ها انواع مختلفی دارند که عبارتند از:

الف- دستورالعمل‌های ورودی
ب- دستورالعمل‌های خروجی
ج- دستورالعمل‌های محاسباتی
د- دستورالعمل‌های شرطی
هـ - دستورالعمل‌های تکرار (حلقه‌ها)

الف- دستورالعمل‌های ورودی

این دســتورالعمل‌ها برای دریافت داده‌های ورودی اســتفاده می‌شوند و معمولاً برای اجرای آن‌ها از عباراتی مانند «بخوان، دریافت‌کن یا بگیر» استفاده می‌شود.

ب- دستورالعمل‌های خروجی

این دســتورالعمل‌ها برای نمایــش اطلاعات خروجی یا پیام‌هــای موردنیاز به منظور راهنمایی کاربر روی صفحه نمایش یا چاپ آن‌ها به وســیله چاپگر اســتفاده می‌شوند و معمولاً برای اجرای آن‌ها عباراتی مانند «نمایش بده یا چاپ‌کن» به کار می‌روند.

ج- دستورالعمل‌های محاسباتی

این دســتورالعمل‌ها در واقع نحوه ارایه و استفاده از فرمول‌ها و انجام عملیات ریاضی و محاسباتی را تعیین می‌کنند و معمولاً برای اجرای آن‌ها از همان شکلی که در ریاضیات وجود دارد، استفاده می‌شود یعنی در سمت راست تساوی عملیات محاسباتی و در سمت چپ تســاوی نام یک متغیر به کار می‌رود؛ البته به جای علامت تســاوی از علامت فلش نیز استفاده می‌شود.

تعریف متغیر: متغیرها مکان‌هایی هستند که توانایی نگهداری و ذخیره‌سازی انواع‌داده را دارند. متغیرها در الگوریتم همان کاربرد ریاضی خود را دارا هستند و علاوه بر اعداد می‌توانند متن یا مقادیر منطقی درست یا نادرست، تاریخ، ساعت و نظایر آن ها را نگهداری کنند.

نحوه استفاده از عملگرهای ریاضی در الگوریتم: برخی از عملیاتی که در الگوریتم انجام می گیرد، عملیات ریاضی و محاسباتی است که برای انجام این گونه دستورات لازم است از عملگرهای ریاضی استفاده شود.

تعریف عملگر: در واقع عملگر یک یا مجموعه عملیاتی است که از پیش در الگوریتم آماده شده است و برای استفاده از آن باید از علایمی که به صورت قراردادی تعریف شده‌اند، استفاده کنید مانند عملگر جمع که می‌تواند دو عدد یا دو متغیر یا ترکیبی از آن ها را با هم جمع و حاصل آن را ارایه کند. اعداد و متغیرهایی که عملگر روی آن ها عملیات انجام می‌دهد، عملوند نامیده می‌شوند. عملگرهای ریاضی در جدول ۱-۱ ارایه شده‌اند.

جدول ۱-۱

عملگر	عنوان	مثال	اولویت
( )	پرانتز	۱۰=۴+ (۲×۳)	۱
*	ضرب	۶=۳×۲	۲
/	تقسیم	۲=۶/۱۲	۲
\	خارج قسمت تقسیم صحیح	۵=۳ \ ۱۷	۳
Mod	باقی‌مانده تقسیم صحیح	۴=۵ mod ۱۴	۴
+	جمع	۴۹=۱۴+ ۳۵	۵
-	تفریق	۳=۷-۱۰	۵

عملگرهای ارایه شده در جدول ۱-۱ نسبت به هم دارای حق تقدم در اجرا هستند، به عبارت دیگر در یک عبارت ریاضی که از چند عملگر استفاده شده است، عملگرها

از سمت چپ عبارت ریاضی به ترتیب اجرا می‌شوند. اما همیشه این‌گونه نیست و در هنگام استفاده از چند عملگر ریاضی، اولویت آن‌ها مطابق جدول ۱-۱ از بالا به پایین درنظر گرفته می‌شود، به عنوان مثال عبارت ۱۰/ ۵ × ۳+۲ را در نظر بگیرید؛ در این عبارت ابتدا عدد ۳ در ۵ ضرب شده و سپس حاصل‌ضرب آن یعنی ۱۵، بر عدد ۱۰ تقسیم می‌شود چرا که عملگرهای ضرب و تقسیم نسبت به عملگر جمع حق تقدم دارند. در پایان، حاصل تقسیم یعنی عدد ۱/۵ با عدد ۲ جمع و در نتیجه عدد ۳/۵ به دست می‌آید.

توجه کنید که عملگرهای ضرب و تقسیم نسبت به هم، جمع و تفریق نیز نسبت به هم دارای اولویت یکسانی هستند و اولویت آن‌ها نسبت به هم از چپ به راست است. به عنوان مثال عبارت ۲×۶-۴۰ +۱۰ را درنظر بگیرید؛ در این عبارت ابتدا عدد ۶ در ۲ ضرب شده، سپس عبارت ۴۰+۱۰ محاسبه می‌شود که نتیجه آن عدد ۵۰ خواهد بود و بعد عمل تفریق انجام می‌شود و حاصل تفریق ۱۲-۵۰ یعنی ۳۸ به دست می‌آید.

البته اولویت عملگرها را می‌توان با پرانتز تغییر داد به عنوان مثال اگر عبارت قبل را با پرانتز به شکل ۲×(۶-۴۰ +۱۰) تغییر دهید ابتدا عبارت داخل پرانتز محاسبه می‌شود سپس نتیجه در عدد ۲ ضرب خواهد شد.

تمرین:

ترتیب انجام عملیات و حاصل عبارات زیر را تعیین کنید:

۱۴-۳×۲) / ۵ + ۲)

((۲۷+۴)×۱۲ -۷) +۸ / ۲

مثال ۱: فرض کنید می‌خواهید مسأله محاسبه محیط یک مستطیل با طول L و عرض W دلخواه را به صورت یک الگوریتم بنویسید، قبلاً این مسأله را بررسی کرده و سه عامل مقادیر معلوم، فرمول‌های محاسباتی و خواسته‌های مسأله (مجهولات) و رابطه بین آن‌ها را تعیین کرده‌اید، بنابراین الگوریتم حل این مسأله به صورت زیر خواهد بود:

	توضیح
۱-شروع الگوریتم	شروع الگوریتم
۲- L را دریافت کن	ورود داده‌ها
۳- W را دریافت کن	ورود داده‌ها
۳ - P ← ۲ ×(L + W)	محاسبه محیط
۴- P را نمایش بده	نمایش خروجی
۵- پایان	پایان الگوریتم

تمرین:

الگوریتمی بنویسید که محیط یک دایره با شعاع دلخواه R را محاسبه کرده و نمایش دهد.

مثال ۲: الگوریتمی بنویسید تا حاصل تقسیم یک عدد دلخواه را بر عدد دلخواه دیگری محاسبه کرده و نمایش دهد.

در این مثال ورودی‌ها شامل دو عدد هستند و محاسبات لازم برای رسیدن به نتیجه، عمل ریاضی تقسیم می‌باشد و خواسته مسأله نیز نتیجه تقسیم دو عدد بر یکدیگر است، بنابراین الگوریتم مربوطه بدین صورت خواهد بود.

	توضیح
۱-شروع	شروع
۲- X را دریافت کن	ورود داده‌ها
۳- Y را دریافت کن	ورود داده‌ها
۳ - $Z \leftarrow X/Y$	محاسبه حاصل تقسیم
۴- Z را نمایش بده	نمایش خروجی
۵- پایان	پایان الگوریتم

در این الگوریتم پس از شروع الگوریتم در مراحل ۲ و ۳ دو عدد از ورودی دریافت شده و در متغیرهای X و Y ذخیره می‌شوند. در مرحله ۴ دو متغیر بر هم تقسیم شده و نتیجه تقسیم در متغیر Z ذخیره می‌شود. در مرحله ۵ حاصل تقسیم روی صفحه نمایش داده می‌شود و در مرحله ۶ الگوریتم پایان می‌یابد.

تمرین:

الگوریتمی بنویسید تا میانگین سه عدد دلخواه را محاسبه کرده و نمایش دهد.

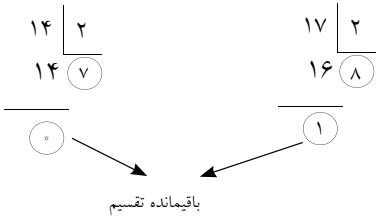
الگوریتمی بنویسید که باقیمانده تقسیم دو عدد دلخواه را بر هم محاسبه کرده و نمایش دهد.

د – دستورالعمل‌های شرطی

گاهی اوقات لازم است با مقایسه مقادیر داده‌های ورودی، اطلاعات خروجی، متغیرها و ... عملیاتی را در الگوریتم هدایت کرده و دستورالعمل‌های خاصی را اجرا کنید یا روند اجرای الگوریتم را با اتخاذ تصمیمات مناسبی کنترل نمایید. در این صورت می‌توانید از دستورات شرطی استفاده کنید.

مثال ۳: الگوریتمی بنویسید که زوج و فرد بودن هر عدد دلخواه را مشخص کند.

در این مسأله داده ورودی شامل یک عدد است و تشخیص زوج یا فرد بودن عدد وارد شده خواسته (مجهول) مسأله است. اگر باقیمانده تقسیم یک عدد بر عدد ۲ صفر شود عدد زوج است و اگر باقیمانده یک باشد عدد موردنظر فرد می‌باشد.



با استفاده از عملگر Mod می‌توان تقسیم‌های فوق را به صورت زیر نوشت:

۱۴ MoD ۲ ⟶ ۰

۱۷ MOD ۲ ⟶ ۱

در این مرحله با این‌که نکات لازم برای حل مسأله به دست آمده است اما یک مشکل هنوز وجود دارد که یک عدد در هر لحظه نمی‌تواند هم زوج و هم فرد باشد، بنابراین مجبور خواهید بود در زمانی که عدد زوج است یک خروجی و وقتی عدد فرد است خروجی دیگری نمایش داده شود. درچنین حالت‌هایی لازم است از دستورات شرطی استفاده کنید. نحوه استفاده از یک دستور شرطی به صورت زیر است:

شکل کلی دستورالعمل شرطی به یکی از صورت‌های زیر است.

اگر شرط (ها) آن‌گاه دستور(ات)

اگر شرط (ها) آن‌گاه دستور(ات) در غیر این صورت دستور(ات)

در حالت اول، ابتدا شرط یا شرط‌های ارایه شده بررسی می‌شوند و در صورتی‌که نتیجه بررسی درست باشد، دستور یا دستورات پس از «آن‌گاه» اجرا می‌شوند، در غیراین‌صورت (نادرست بودن شرط بررسی شده) دستورالعملی که پس از دستورالعمل شرطی قرار گرفته، اجرا خواهد شد بدون آن‌که دستور یا دستورات پس از «آن‌گاه» اجرا شود.

در حالت دوم، شکل کامل‌تری از دستورالعمل شرطی را ملاحظه می‌کنید که در آن ابتدا شرط یا شرط‌ها مورد بررسی قرار می‌گیرند؛ اگر نتیجه ارزیابی آن‌ها درست باشد، دستورالعمل شرط مانند حالت اول رفتار می‌کند اما اگر نتیجه ارزیابی شرط یا شرط‌ها نادرست باشد بدون آن‌که دستورات بخش «آن‌گاه» اجرا شوند، دستورات موجود در بخش «درغیر این‌صورت» اجرا خواهند شد سپس دستورالعملی که پس از دستور شرطی قرار دارد، اجرا می‌شود.

جدول ۲ - ۱

عملگر	عنوان	مثال	نتیجه
=	تساوی	۴=۴	درست
>=	بزرگ‌تر یا مساوی	۵ >=۲	درست
<=	کوچک‌تر یا مساوی	۱۰۰ <=۱۵	نادرست
>	بزرگ‌تر	۲۹>۷۴	نادرست
<	کوچک‌تر	۷<۳	نادرست
<>	نامساوی	۱۵<>۸	درست

بنابراین الگوریتم موردنظر بدین صورت خواهد بود:

	توضیح
۱- شروع	شروع الگوریتم
۲ - Xرا دریافت کن	ورود داده
۳ - R ← X MoD ۲	مقدار باقیمانده تقسیم را در R ذخیره می‌کند.
۴ - اگر R = ۰ آن‌گاه «عدد زوج است.» را نمایش بده.	در صورتی‌که R برابر صفر باشد عبارت «عدد زوج است.» را نمایش می‌دهد.
۵- اگر R = ۱ آن‌گاه «عدد فرد است.» را نمایش بده.	در صورتی‌که R برابر یک باشد عبارت «عدد فرد است.» را نمایش می‌دهد.
۶- پایان	پایان الگوریتم

پس از شـــروع الگوریتم، در مرحله ۲ یک عدد از ورودی دریافت شـــده و در متغیر X ذخیره می‌شود سپس در مرحله ۳ باقیمانده تقسیم X بر ۲ محاسبه شده و در متغیر R ذخیره می‌شود. در مرحله ۴ با استفاده از یک دستور شرطی مقدار متغیر R با مقدار صفر مقایسه می‌شود در صورتی که مقدار متغیر نیز صفر باشد نتیجه بررسی شرط درست (True) و در نتیجه عدد زوج خواهد بود و عبارت «عدد زوج است.» نمایش داده می‌شود اما اگر مقدار متغیر R صفر نباشد نتیجه بررسی شرط نادرست (False) بوده و دستور بعد از آن‌گاه اجرا نمی‌شـــود. در مرحله ۵ مجدداً مقدار R بررســـی می‌شود و در صورتی‌که مقدار آن برابر با یک باشد «عبارت عدد فرد است.» نمایش داده خواهد شد و در صورتی‌که مقدار آن برابر با یک نباشد عبارت مزبور نمایش داده نمی‌شود.

تمرین:

الگوریتمی بنویسید که بخش‌پذیری یک عدد را بر عدد دیگر تعیین کند

بـــا توجه به مطالب گفته شـــده و این نکته که یک عـــدد نمی‌تواند هم زوج و هم فرد باشد الگوریتم مثال قبل را به‌صورت زیر می‌توان خلاصه‌تر نمود و به جای استفاده از دو دستورالعمل شرطی مسأله را با یک دستورالعمل شرطی حل کرد.

	توضیح
۱- شروع	شروع الگوریتم
۲ - X را دریافت کن	ورود داده
۳ ـ ۲ X MoD ← R	مقدار باقیمانده تقسیم را در R ذخیره می‌کند.
۴ - اگر R = ۰ آن‌گاه «عدد زوج است.» را نمایش بده ، در غیر این صورت« عدد فرد است. »را نمایش بده	در صورتی‌که مقدار R برابر با صفر باشد عبارت «عدد زوج است.»را نمایش می‌دهد، در غیر این‌صورت عبارت «عدد فرد است» را نمایش می‌دهد.
۵- پایان	پایان الگوریتم

در این الگوریتم جدید از یک دســـتورالعمل شـــرطی استفاده شـــده است اما در نتیجه اجـــرای الگوریتم هیچ تغییری به وجود نمی‌آیـــد، در این حالت اگر مقدار باقیمانده یعنی متغیر R برابر با صفر باشـــد دســـتوری که در بخش «آن‌گاه» قرار دارد اجرا می‌شود اما اگر مقدار متغیر R برابر با صفر نباشد دستور بخش « در غیر این صورت» اجرا خواهد شد.

تمرین:

الگوریتم بخش‌پذیری دو عدد بر یکدیگر را با یک دستورالعمل شرطی بنویسید.

مثال ۴: الگوریتمی بنویسید که سه عدد دلخواه را دریافت کرده و بزرگ‌ترین مقدار را در بین آن‌ها تعیین کند و نمایش دهد.

در این مسأله سه داده ورودی وجود دارند که آن‌ها را اعداد A ، B و C در نظر می‌گیریم، برای پیدا کردن بزرگ‌ترین مقدار در بین این سه عدد می‌توانیم از روش مقایسه استفاده کنیم؛ به این منظور ابتدا فرض می‌کنیم مقدار ذخیره شده در متغیر A بزرگ‌ترین مقدار است و مقدار آن را در متغیر دیگری به نام MAX ذخیره می‌کنیم، به این دلیل که متغیرهای دیگر یعنی B و C را با مقایسه با این مقدار (متغیر MAX) بررسی کرده و بزرگ‌تر بودن یا نبودن آن‌ها را از مقدار اول مشخص می‌کنیم یعنی متغیر B را با MAX مقایسه کرده و در صورت بزرگ‌تر بودن مقدار B از MAX، متغیر B را در MAX ذخیره می‌کنیم، اگر به همین ترتیب این عملیات را برای متغیر C نیز انجام دهیم، در پایان بزرگ‌ترین عدد در متغیر MAX ذخیره خواهد شد.

با توجه به مطالب گفته شده الگوریتم موردنظر به صورت زیر خواهد بود:

	توضیح
۱- شروع	شروع الگوریتم
۲- اعداد A، B و C را دریافت کن	ورود داده
۳ - MAX ← A	مقدار متغیر A در MAX ذخیره می‌شود.
۴- اگر B > MAX آن‌گاه MAX ← B	در صورتی‌که مقدار B از MAX بزرگ‌تر باشد مقدار B را در MAX ذخیره می‌کند.
۵ - اگر C > MAX آن‌گاه MAX ← C	در صورتی‌که مقدار C از MAX بزرگ‌تر باشد مقدار C را در MAX ذخیره می‌کند.
۶ - MAX را نمایش بده	نمایش خروجی
۷ - پایان	پایان الگوریتم

تمرین:

الگوریتمی بنویسید که چهار عدد دلخواه را دریافت کرده و کوچک‌ترین و بزرگ‌ترین مقدار در بین آن‌ها را به دست آورد، سپس آن را برای اعداد مختلف اجرا و آزمایش کنید.

نکته در صورت نیاز، می‌توان یک دستورالعمل شرطی را در بخش دستورات بعد از «آن‌گاه» یا «در غیر این‌صورت» دستورالعمل شرطی دیگری قرار داد؛ در این حالت به مجموعه دستورالعمل‌های شرطی، دستورالعمل‌های شرطی تودرتو نیز می‌گویند.

عملگرهای منطقی

گاهی اوقات ممکن است لازم باشد در دستورالعمل‌های شرطی که در حل یک الگوریتم به کار می‌روند بیش از یک شرط مورد بررسی قرار گیرند و با توجه به نتیجه بررسی تمامی شرط‌های ذکر شده سایر دستورالعمل‌ها اجرا شوند. در چنین حالت‌هایی می‌توان از عملگرهای منطقی برای ترکیب دو یا چند شرط استفاده کرد.

مثال ۵: الگوریتمی بنویسید که مقدار مصرف برق یک مشترک دلخواه را دریافت کرده و براساس جدول ۳-۱، رتبه میزان مصرف وی را مشخص کند.

جدول ۳-۱

ردیف	میزان مصرف (کیلو وات)	رتبه
۱	۰ تا ۵۰	کم مصرف
۲	۵۰ تا ۲۰۰	عادی
۳	۲۰۰ به بالا	پرمصرف

برای طراحی الگوریتمی که بتواند رتبه مصرف برق یک مشترک را مشخص کند شرایط مختلفی در نظر گرفته شده است. برای حل این مثال دو روش وجود دارد: روش اول استفاده از دستورالعمل‌های شرطی تودرتو و روش دوم استفاده از عملگرهای منطقی است.

در روش اول لازم است تا برای ردیف‌های ۱ و ۲ در جدول ۳-۱ از یک دستورالعمل شرطی در بخش «آن‌گاه» دستورالعمل شرطی دیگری استفاده شود تا هر دو شرط موردنظر در این ردیف‌ها مورد ارزیابی قرار گیرند. البته برای ردیف ۳ می‌توان از یک دستورالعمل شرطی به تنهایی استفاده کرد. با توجه به این توضیحات می‌توانید الگوریتم زیر را بنویسید:

در این الگوریتم kw به عنوان میزان مصرف برق در نظر گرفته شده است.

۱- شروع

۲- kw را دریافت کن.

۳- اگر kw > = ۰ آن‌گاه اگر kw < ۵۰ آن‌گاه «مشترک کم مصرف است.» را نمایش بده.

۴- اگر kw > = ۵۰ آن‌گاه اگر kw <۲۰۰ آن‌گاه «مشترک عادی است.» را نمایش بده.

۵- اگر kw > =۲۰۰ آن‌گاه «مشترک پرمصرف است. » را نمایش بده.

۶- پایان

به طور کلی سه نوع عملگر منطقی که در الگوریتم مورد استفاده قرار می‌گیرند در جدول ۴-۱ ارائه شده‌اند.

جدول ۴-۱

مفهوم	عملگر
«و» منطقی	AND
«یا» منطقی	OR
نقیض	NOT

وقتی دو یا چند شرط با عملگر منطقی AND با هم ترکیب می‌شوند، نتیجه مقایسه، زمانی درست (True) خواهد بود که نتیجه تمام شرط‌ها و مقایسه‌ها True باشند؛ به عنوان مثال شرط‌های X > ۰ و Y < ۵ را در نظر بگیرید. اگر بخواهیم این دو شرط را با یکدیگر ترکیب کنیم نتیجه ترکیب آن‌ها زمانی درست (True) خواهد بود که نتیجه بررسی هر دو شرط درست (True) باشد. به عنوان نمونه اگر مقدار X برابر ۲ و Y برابر ۱ باشد هر دو شرط درست (True) بوده و نتیجه ترکیب آن‌ها نیز درست (True) خواهد بود.

اما اگر نتیجه بررسی شرط X > ۰ یا Y < ۵ یا هر دو شرط نادرست (False) باشد، نتیجه ترکیب آن‌ها نیز نادرست (False) خواهد بود. به عنوان نمونه اگر مقدار X برابر با ۵- و مقدار Y برابر با ۱ باشد شرط X > ۰ نادرست (False) و شرط Y < ۵ درست (True) خواهد بود و در نتیجه ترکیب آن‌ها نیز نادرست(False) می‌شود. بنابراین جدول درستی عملگر منطقی AND مطابق جدول ۵-۱ است.

درصورتی‌که دو یا چند شرط را با عملگر منطقی OR با هم ترکیب کنید، نتیجه شرط زمانی نادرست (False) خواهد بود که نتیجه تمام شرط‌ها نادرست (False) باشد و در سایر حالات، نتیجه درست (True) خواهد بود؛ بنابراین جدول درستی عملگر منطقی OR مطابق جدول ۶-۱ است.

جدول ۵-۱

X > ۰	Y < ۵	X > ۰ AND Y < ۵
T	F	F
F	T	F
F	F	F
T	T	T

جدول ۶-۱

X > ۰	Y < ۵	X > ۰ OR Y < ۵
T	F	T
F	T	T
F	F	F
T	T	T

اگر از عملگر NOT روی یک شرط استفاده شود نتیجه درستی آن‌را معکوس می‌کند یعنی اگر نتیجه شرط درست (True) باشد آن‌را به نادرست (False) و اگر نتیجه شرط نادرست (False) باشد آن‌را به درست (True) تبدیل می‌کند. جدول درستی عملگر منطقی NOT مطابق جدول ۷-۱ می‌باشد.

جدول ۷-۱

X > ۰	NOT X > ۰
T	F
F	T

نکته در جدول‌های ۵-۱، ۶-۱ و ۷-۱، T بیانگر درست (True) و F بیانگر نادرست (False) می‌باشد.

تمرین:

الگوریتمی بنویسید تا معدل یک دانش‌آموز را دریافت کرده و رتبه وی را براساس شرایطی که در ادامه می‌آید، تعیین کند:

الف– درصورتی که میانگین نمرات بیشتر از ۱۸ باشد، رتبه ممتاز برای وی اعلام شود.

ب– درصورتی که میانگین نمرات بین ۱۶ تا ۱۸ باشد، رتبه خوب برای وی اعلام شود.

ج– درصورتی که میانگین نمرات بین ۱۲ تا ۱۶ باشد، رتبه متوسط برای وی اعلام شود.

د – درصورتی که میانگین نمرات کمتر از ۱۲ باشد، رتبه ضعیف برای وی اعلام شود.

هـ – دستورالعملهای تکرار (حلقه‌ها)

استفاده از دستورالعمل‌هایی که تاکنون‌فراگرفته‌اید در حل بعضی از مسایل کافی نیست و لازم است تا برخی از دستورالعمل‌ها را به دفعات تکرار نمایید؛ در این موارد از دستورالعمل تکرار یا همان حلقه‌ها استفاده کنید.

مثال ۶: الگوریتمی بنویسید که مجموع اعداد ۱ تا ۱۰ را محاسبه نموده و نمایش دهد.

اگر برای حل این مسأله بخواهید از دستورالعمل‌هایی که تاکنون ارائه شده‌اند استفاده کنید در این حالت الگوریتمی به صورت زیر به‌دست می‌آید.

۱ – شروع

۲ – SUM ← ۰

۳ – SUM = ۱ + ۲+ ۳ + ۴ + ۵ + ۶ + ۷ + ۸ + ۹ + ۱۰

۴ – SUM را نمایش بده

۵ – پایان

با این حال مشاهده می‌کنید که الگوریتم شکل نامناسبی دارد و اگر داده‌ها زیادتر شوند استفاده از این روش عملاً غیرممکن است. برای حل این مشکل می‌توان دستورالعمل تکرار را به کار برد.

دستورالعمل حلقه از اجزای مختلفی تشکیل می‌شود که عبارتند از:

شـــمارنده حلقه: یک متغیر عددی است که تعداد دفعات تکرار دستورالعمل‌ها را در حلقه‌کنترل می‌کند. مقدار شمارنده در هر بار اجرای حلقه افزایش یا کاهش می‌یابد.

مقدار اولیه: مقدار اولیه حلقه قبل از شروع حلقه تعیین می‌شود و به وسیله آن می‌توان مقدار اولیه را برای شمارنده حلقه تعیین کرد.

شـــرط حلقه: برای کنترل تعداد دفعات تکرار حلقه، باید از یک شـــرط استفاده کرد. شـــرط موجود در حلقه، نقطه پایان تکرار دســـتورالعمل‌ها را در حلقه مشخص می‌کند و باید به‌گونه‌ای تنظیم شود تا از ایجاد حلقه نامحدود جلوگیری کند. برای ایجاد شرط در یک حلقه می‌توان از دستورالعمل‌های شرطی استفاده کرد.

دستورات حلقه: بخش دیگر در حلقه، دستورالعمل‌هایی هستند که داخل حلقه تکرار می‌شوند. این دستورالعمل‌ها با توجه به نیاز مسأله انتخاب می‌شوند.

بنابراین می‌توان الگوریتم مثال ۸ را به صورت زیر را ارائه نمود:

	توضیح
۱- شروع	
۲- ۱ ← N	تنظیم مقدار اولیه برای شمارنده حلقه
۳- ۰ ← SUM	
۴- SUM + N ← SUM	
۵- N + ۱ ← N	محاسبه مجموع اعداد
۶- اگر N ← ۱۰ آن‌گاه به مرحله۴ برو	افزایش شمارنده حلقه
۷- SUM را نمایش بده	شرط حلقه
۸- پایان	دستورات حلقه

در این الگوریتم پس از شـــروع، ابتدا عدد ۱ در متغیر N ذخیره می‌شـــود که به عنوان شـــمارنده حلقه از آن اســـتفاده می‌شـــود. در مرحله بعد مقدار صفر در متغیر SUM ذخیره می‌شود که برای محاسبه مجموع اعداد از آن استفاده می‌شود. در مرحله ۴ مجموع اعداد محاسبه می‌شود.

در مرحله بعد مقدار شمارنده حلقهN یک واحد افزایش می‌یابد، در مرحله ۶ یک

دستورالعمل شرطی مقدار شمارنده را بررسی می‌کند تا تعداد دفعات تکرار دستورالعمل‌ها از مقدار ۱۰ بیشتر نشود. در این حالت اگر مقدار شمارنده کوچک‌تر یا مساوی ۱۰ باشد اجرای الگوریتم به مرحله ۴ منتقل می‌شود. دستورات مراحل ۴، ۵ و ۶ آن‌قدر تکرار می‌شوند تا مقدار شمارنده از مقدار ۱۰ بیشتر شود و در نتیجه اجرای حلقه ایجاد شده خاتمه می‌یابد و اجرای الگوریتم با اجرای مرحله ۷ ادامه یافته و با نمایش میانگین اعداد در مرحله ۸ خاتمه می‌یابد. اگر الگوریتم را اجرا و آزمایش کنید جدول بررسی عملکرد آن به صورت جدول ۸–۱ خواهد بود.

جدول ۸-۱

خروجی	SUM	N
۵۵	~~۰~~	~~۱~~
	~~۱~~	~~۲~~
	~~۳~~	~~۳~~
	~~۶~~	~~۴~~
	~~۱۰~~	~~۵~~
	~~۱۵~~	~~۶~~
	~~۲۱~~	~~۷~~
	~~۲۸~~	~~۸~~
	~~۳۶~~	~~۹~~
	~~۴۵~~	~~۱۰~~
	۵۵	۱۱

مثال ۷: الگوریتمی بنویسید که اعداد طبیعی کوچک‌تر از ۵۰ را نمایش دهد.

	توضیح
۱– شروع	
۲– ۱ ⟵ N	تنظیم مقدار اولیه برای شمارنده حلقه
۳– N را نمایش بده	نمایش خروجی
۴– N + ۱ ⟵ K	افزایش شمارنده حلقه } دستورات حلقه
۵– اگر N < ۵۰ آن‌گاه به مرحله ۳ برو	شرط حلقه
۶– پایان	

مثال ۸: الگوریتمی بنویسیدکه اعداد زوج کوچک‌تر یا مساوی عدد طبیعی و دلخواه N را نمایش دهد. این الگوریتم مشابه الگوریتم قبلی است با این تفاوت که مقدار شمارنده از ۲ شروع شده و هر بار به میزان ۲ واحد افزایش می‌یابد.

	توضیح
۱- شروع	
۲- N را دریافت کن	
۳- اگر N=۱ آن‌گاه پایان	خاتمه الگوریتم در صورتی‌که داده ورودی عدد یک باشد.
۴- ۲ ⟵ K	
۵- K را نمایش بده	
۶- K ⟵ K + ۲	
۷- اگر K<=N آن‌گاه برو به مرحله ۵	
۸- پایان	

تمرین:
الگوریتمی بنویسید که مضارب کوچک‌تر از ۱۰۰ عدد ۳ را نمایش دهد.

مثال ۹: الگوریتمی بنویسید که مجموع و تعداد اعداد طبیعی موجود بین اعداد طبیعی دلخواه M و N را نمایش دهد (با فرض این که M کوچک‌تر از N باشد).

۱- شروع
۲- M و N را دریافت کن
۳- ۰ ⟵ SUM و ۰ ⟵ NO
۴- M+۱ ⟵ K
۵- SUM+K ⟵ SUM
۶- NO+۱ ⟵ NO
۷- K+۱ ⟵ K
۸- اگر K<N آن‌گاه برو به مرحله ۵
۹- SUM و NO را نمایش بده
۱۰- پایان

نکته‌ای که باید در این الگوریتم بدان اشاره کنیم نحوه محاسبه مجموع و تعداد اعداد

طبیعی است. برای این کار از متغیر NO برای شمارش تعداد اعداد بین M و N استفاده می‌شود. متغیر K برای ایجاد اعداد طبیعی بین M و N به کار می‌رود.

در صورت اجرای این الگوریتم به ازای M=۳ و N=۹، جدول بررسی عملکرد به صورت زیر خواهد بود:

جدول ۹-۱

M	N	K	SUM	NO	خروجی
۳	~~۹~~	~~۳~~	~~۰~~	~~۰~~	
	~~۳~~	~~۵~~	~~۴~~	~~۱~~	
	~~۳~~	~~۶~~	~~۹~~	~~۲~~	
	~~۳~~	~~۷~~	~~۱۵~~	~~۳~~	
	~~۳~~	~~۸~~	~~۲۲~~	~~۴~~	
	۳	۹	۳۰	۵	۳۰ ۵

تمرین:

الگوریتم قبل را به گونه‌ای تنظیم کنید که بدون در نظر گرفتن فرض M<N، توانایی ارایه پاسخ صحیح را داشته باشد، به عبارت دیگر در صورتیکه کاربر عدد بزرگ‌تر را برای M و عدد کوچک‌تر را برای N وارد کند، پاسخ الگوریتم درست باشد.

مثـــال ۱۰: الگوریتمی بنویســـید که مقســـوم علیه‌های عدد طبیعـــی و دلخواه M را محاســـبه کند. به این منظور باید از یک حلقه اســـتفاده کنید و متغیر M را هر بار بر یک متغیر K که به عنوان مقســـوم علیه در نظر گرفته می‌شود و مقدار اولیه آن ۱ است، تقسیم نمایید ســـپس باقیمانده این تقســـیم را بررسی کنید. در صورتی‌که مقدار باقیمانده مساوی صفر باشـــد به این معنی است که متغیر K میتواند مقسوم‌علیه متغیرM باشد، بنابراین متغیر K نمایش داده می‌شود. برای محاسبه باقیمانده تقسیم صحیح نیز می‌توانید از عملگر Mod استفاده کنید.

اکنون الگوریتم موردنظر به صورت زیر خواهد بود:

۱ – شروع

۲ – M را دریافت کن

۳ – ۱ ⟵ K

۴ – اگر M Mod K=۰ آن‌گاه K را نمایش بده

۵– K ⟵ K + ۱

۶ – اگر K<=M آن‌گاه برو به مرحله ۴

۷ – پایان

مثال ۱۱: الگوریتمی بنویسیدکه مجموع ارقام هر عدد طبیعی و دلخواه را محاسبه کرده و نمایش دهد.

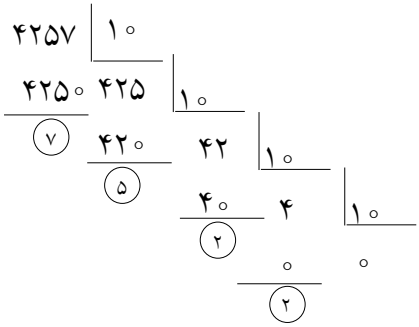
برای طراحی الگوریتم قبل و محاســـبه مجموع ارقام یک عدد، ابتدا باید ارقام عدد را یکـــی یکـــی از آن جدا کنید؛ به این منظور می‌توانید از عملگر Mod و محاســـبه باقیمانده تقسیم عدد مربوطه بر عدد ۱۰ استفاده نمایید، سپس خارج قسمت عدد مربوطه را که بر ۱۰ تقســـیم می‌شود، محاسبه کنید. از این عدد می‌توانید برای به دست آوردن رقم بعدی در عدد اصلی اســـتفاده نماییـــد. این عملیات می‌تواند در یک حلقه تا رســـیدن به خارج قسمت صفر ادامه یابد. برای محاسبه خارج قسمت تقسیم صحیح نیز می‌توانید از عملگر \ استفاده کنید؛ به عنوان نمونه به این مثال توجه کنید:

۱۷ | ۵
۱۵ (۳) ← خارج قسمت تقسیم صحیح
۲ ← باقیمانده تقسیم صحیح

۱۷ \ ۵ = ۳

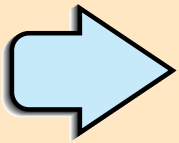
با توجه به مطالب ارایه شده، اگر بخواهیم ارقام یک عدد را از آن جدا کنیم روش ریاضی این عملیات به صورت بعد خواهد بود:

به عنوان مثال اگر عدد مورد نظر ۴۲۵۷ باشد:



بنابراین می‌توان الگوریتم زیر را برای حل این مثال ارایه کرد:

۱ – شروع
۲ – M را دریافت کن
۳ – SUM ← 0
۴– DIGIT ← M Mod 10
۵ – SUM ← SUM + DIGIT
۶ – M ← M \ 10
۷ – اگر M <> 0 آن‌گاه برو به مرحله ۴
۸ – SUM را نمایش بده
۹ – پایان



Learn in English

Algorithm

To make a computer do anything, you have to write a computer program. To write a computer program, you have to tell the computer, step by step, exactly what you want it to do. The computer then «executes» the program, following each step mechanically, to accomplish the end goal.

When you are telling the computer *what* to do, you also get to choose how it's going to do it. That's where computer algorithms come in. The algorithm is the basic technique used to get the job done. Let's follow an example to help get an understanding of the algorithm concept.

Let's say that you have a friend arriving at the airport, and your friend needs to get from the airport to your house. Here are four different algorithms that you might give your friend for getting to your home:

❊ **The taxi algorithm**

1 - Go to the taxi stand

Get in a taxi

2 - Give the driver my address

❊ **The call-me algorithm:**

1 - When your plane arrives, call my cell phone.

2 - Meet me outside baggage claim.

❊ **The rent-a-car algorithm:**

1 - Take the shuttle to the rental car place.

2 - Rent a car.

3 - Follow the directions to get to my house.

❊ **The bus algorithm:**

1 - Outside baggage claim.

2 - Transfer to bus on Main Street.

3 - Get off on 20 street.

4 - Walk two blocks north to my house.

All four of these algorithms accomplish exactly the same goal, but each algorithm does it in completely different way. Each algorithm also has a different cost and a different travel time. Taking a taxi, for example, is probably the fastest way, but also the most expensive. Taking the bus is definitely less expensive, but a whole lot slower. You choose the algorithm based on the circumstances.

واژه‌نامه

Circumstance	وضعیت
Driver	راننده
Execute	اجرا کردن
Expensive	گران
Fast	سریع
Program	برنامه
Rent	اجاره

خلاصه مطالب

- برای دستیابی به روش حل مسایل، شناسایی سه پارامتر مقادیر معلوم، محاسبات و خواسته‌های مسأله (مجهولات) کار را آسانتر می‌کند.
- الگوریتم مجموعه‌ای از دستورالعمل‌هاست که به صورت مرحله به مرحله اجرا می‌شوند و هدف مشخصی را دنبال می‌کنند و دارای شروع و خاتمه می‌باشند.
- انواع دستورالعمل‌ها در الگوریتم عبارتند از: ورودی، محاسباتی، خروجی، شرطی و حلقه ها.
- دستورالعمل‌های ورودی برای دریافت داده‌ها و دستورالعمل‌های خروجی برای نمایش داده و اطلاعات خروجی یا پیام‌های موردنیاز استفاده می‌شوند.
- دستورالعمل‌های محاسباتی برای انجام عملیات ریاضی و محاسباتی استفاده می شوند.
- دستورالعمل‌های شرطی برای انجام مقایسه‌ها و کنترل روند اجرای الگوریتم استفاده می‌شوند.
- دستورالعمل‌های تکرار (حلقه‌ها) برای ایجاد تکرار اجرای دستورالعمل‌ها استفاده می‌شوند.
- عملگر، یک یا مجموعه عملیاتی است که در ماشین الگوریتم به صورت از پیش آماده تعریف شده است.

آزمون نظری

۱ – نتیجه اجرای الگوریتم زیر چیست؟

۱ – شروع

۲ – SUM ← ۰

۳ – I ←۱

۴ – SUM ← SUM + I

۵ – I ← I + ۱

۶ – اگر I < = ۱۰۰ آن‌گاه برو به مرحله ۴

۷ – SUM را نمایش بده

۸ – پایان

الف– اعداد ۱ تا ۹۹ را نمایش می‌دهد

ب– مجموع اعداد ۰ تا ۹۹ را نمایش می‌دهد.

ج– مجموع اعداد ۱ تا ۱۰۰را نمایش می‌دهد.

د– مجموع اعداد ۱ تا ۹۹ را نمایش می‌دهد.

۲ – کدام گزینه در رابطه با حاصل الگوریتم زیر درست است؟

۱ – شروع کن

۲ – I ←۲

۳ ـ I را نمایش بده

۴ – K ← I × ۲

۵ – I ← I + ۲

۶ ـ K را نمایش بده

۷ ـ اگر I < ۱۰ آن‌گاه برو به مرحله ۵ درغیراین‌صورت برو به مرحله ۸

۸ ـ پایان

الف– ۲ و ۴ و ۴ و ۴ و ۴

ب– ۲ و ۴ و ۶ و ۸

ج– ۲ و ۴ و ۶ و ۸ و ۱۰

د– ۲ و ۴ و ۴ و ۴

۳ – نتیجه اجرای الگوریتم زیر چیست؟

۱ – شروع

۲ – I ←۱

۳ – اگر I Mod۲=۰ آن‌گاه I را نمایش بده.

۴- I ← I + ۱

۵ ـ اگر I <۱۰۰ آن‌گاه برو به مرحله ۳

۶ ـ پایان

الف- اعدد بین صفر و ۱۰۰

ب- اعداد طبیعی فرد کوچک‌تر از ۱۰۰

ج- اعداد طبیعی زوج کوچک‌تر از ۱۰۰

د- اعداد بین ۱ و ۱۰۰

۴ – کدام عملگر از اولویت بالاتری برخوردار است؟

الف- + ب- / ج- Mod د- -

۵ – در صورتیکه X = ۵ ، Y= -۱۰ و Z =۲۷ باشد، نتیجه عبارت زیر چیست؟

(X>۰ یا Y<۲۰) و (Z>=X)

الف- درست ب- نادرست ج- ۱- د- ۰

۶ – حاصل عبارت زیر کدام است؟

۵+۱۰-۲۰/۴×۳

الف- ۰ ب- ۳/۷۵ ج- ۳ د- ۱۳

۷ – حاصل عبارت زیر چیست؟

۱۷ Mod ۳

الف- ۲ ب- ۳ ج- ۴ د- ۵

۸ – حاصل عبارت زیر چیست؟

۲۵ \ ۴

الف- ۱ ب- ۲ ج- ۵ د- ۶

۹ – در کدام مرحله از مراحل حل یک مسأله ارتباط بین داده‌ها و مجهول‌ها مشخص می‌شود؟

الف- شناخت مسأله ب- تجزیه مسأله

ج- طراحی راه‌حل د- تحلیل مسأله

۱۰ – نتیجه اجرای الگوریتم زیر چیست؟

۱ – شروع

۲ – K ← ۱ و SUM ← ۰

۳ – N را دریافت کن

۴ – اگر N Mod K=۰ آن‌گاه SUM←SUM+K

۵ – K ◄— K + ۱

۶ – اگر K <= N آن‌گاه برو به مرحله ۴

۷ –SUM را نمایش بده

۸ – پایان

الف– مجموع باقیمانده تقسیم N بر K ب– مجموع مقسوم‌علیه‌های عدد N

ج– مجموع مقسوم‌علیه‌های عدد K د– مجموع مقسوم‌علیه‌های عدد K بر N

۱۱ –The algorithm is the basic technique used to write a ..... for running in the computer.

a– commands

b– statments

c– program

d– step by step

۱۲– الگوریتم را تعریف کنید و برای آن یک مثال بزنید.

۱۳– مراحل حل مسأله را نام ببرید.

۱۴– انواع دستورالعمل‌ها را در الگوریتم نام ببرید.

۱۵– شرایط اصلی یک الگوریتم خوب را توضیح دهید.

۱۶– کاربرد عملگرهای منطقی و انواع آن‌ها را توضیح دهید.

۱۷– دستورالعمل‌های شرطی و کاربرد آن‌ها را توضیح دهید.

۱۸– دستورالعمل‌های تکرار و کاربرد آن‌ها را توضیح دهید.

۱۹– عوامل مؤثر در شناخت و حل مسائل را توضیح دهید.

۲۰– منظور از جامع بودن الگوریتم چیست؟

۲۱– عوامل اصلی در شناخت دقیق یک مسأله را توضیح دهید.

آزمون عملی

۱ – الگوریتمی بنویسید که مضارب کوچک‌تر از ۱۰۰۰ عدد ۵ را محاسبه کرده و نمایش دهد.

۲ – الگوریتمی بنویسید که دما را برحسب فارنهایت دریافت کرده و معادل آن‌را به سانتی‌گراد نمایش دهد (با توجه به فرمول $C = \frac{۵}{۹}(F - ۳۲)$).

۳ – عدد طبیعی و دلخواه N را دریافت کرده و N! را محاسبه کرده و نمایش دهد.

۴ – الگوریتمی بنویسید که حقوق یک کارمند را دریافت کرده و میزان مالیات حقوق وی را مطابق جدول زیر پس از محاسبه نمایش دهد.

مالیات	حقوق
صفر	کمتر از ۴/۰۰۰/۰۰۰ ریال
۲ درصد حقوق	از ۴/۰۰۰/۰۰۰ ریال تا ۵/۵۰۰/۰۰۰ ریال
۳ درصد حقوق	از ۵/۵۰۰/۰۰۰ ریال تا ۷/۰۰۰/۰۰۰ ریال
۵ درصد حقوق	از ۷/۰۰۰/۰۰۰ ریال به بالا

۵ – الگوریتمی بنویسیدکه نمرات ۱۰ درس یک دانش‌آموز را دریافت کرده و معدل و میانگین نمرات وی را نمایش دهد.

۶ – الگوریتمی بنویسید که عدد طبیعی و دلخواه N را دریافت کرده و ارقام آن‌را معکوس کند (مثلاً عدد ۲۴۸۵ به ۵۸۴۲ تبدیل شود).

۷ – الگوریتمی بنویسید که مقدار زمان را براساس ثانیه دریافت کرده و مقدار ساعت‌ها، دقیقه‌ها و ثانیه‌های آن‌را محاسبه کرده و نمایش دهد.

۸ – الگوریتمی بنویسید که عدد طبیعی و دلخواه M را دریافت کرده و اعداد زوج کوچک‌تر از آن را نمایش دهد.

۹ – الگوریتمی بنویسید که دو عدد طبیعی دلخواه را دریافت کرده و بزرگ‌ترین مقسوم‌علیه مشترک آن‌ها را محاسبه نموده و نمایش دهد.

واحد کاردوم

هدف جزیی

توانایی ترسیم فلوچارت

هدف‌های رفتاری

پس از مطالعه این واحد کار از فراگیر انتظار می‌رود که:

۱ – مفهوم فلوچارت را توضیح دهد.

۲ – علایم ترسیم فلوچارت و نحوه استفاده از آن‌ها را توضیح دهد.

۳ – بتواند انواع فلوچارت را برای مسایل مختلف طراحی کند.

کلیات

در ارایه الگوریتم‌ها به منظور حل مسایل مختلف از جملات فارسی یا سایر زبان‌ها به همراه تعدادی از علایم قراردادی استفاده می‌شود، البته در الگوریتم‌های ساده و کوچک، این روش به خوبی شما را برای تبدیل الگوریتم به برنامه موردنظر کمک می‌کند اما با پیچیده شدن الگوریتم و افزایش دستورالعمل‌ها این کار دشوار خواهد شد، بنابراین لازم است از روشهای مفیدتری استفاده کنید؛ یکی از این روش‌ها، ترسیم و طراحی فلوچارت است. فلوچارت (نمودار گردشی) در واقع مجموعه‌ای از اشکال و ترسیمات قراردادی است که دستورالعمل‌ها و ترتیب اجرای آنها را مطابق با الگوریتم موردنظر نمایش می‌دهد. فلوچارت تمام ویژگیهای الگوریتم را داشته، علاوه بر این امکان درک بهتر از نحوه‌اجرای دستورالعمل‌ها را نیز به وجود می‌آورد.

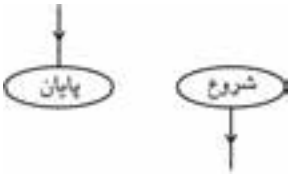
فلوچارت‌ها را می‌توانید پس از طراحی الگوریتم یا به طور مستقل پس از بررسی مسأله و انتخاب روش حل آن ترسیم کنید.

۱–۲ علایم و اشکال در فلوچارت

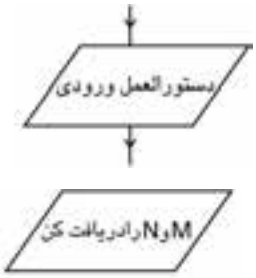
برای تبدیل یک الگوریتم به فلوچارت از علایم قراردادی به جای استفاده از جملات استفاده می‌شود و به جای هر یک از دستورالعمل‌ها، می‌توانید شکل و علامت معادل آن را به‌کار بگیرید و براساس ترتیب اجرای دستورالعمل‌ها این اشکال را با استفاده از خطوط فلش‌دار به یکدیگر متصل نمایید. لازم به ذکر است که تعداد فلش‌هایی که می‌تواند به هر علامت وارد شود، نامحدود است اما در تمام علایم بجز علامت شرط فقط یک فلش می‌تواند خارج شود.

۱–۱–۲ علایم شروع و پایان

برای دستورالعمل‌های شروع کن و پایان از علامت بیضی به صورت زیر استفاده کنید:



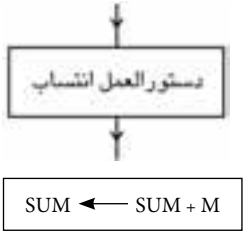
۲-۱-۲ علایم ورودی



برای دستورالعمل‌های ورودی مانند دریافت‌کن، بگیر و نظایر آن می‌توانید از شکل متوازی‌الاضلاع استفاده کنید.
مانند:

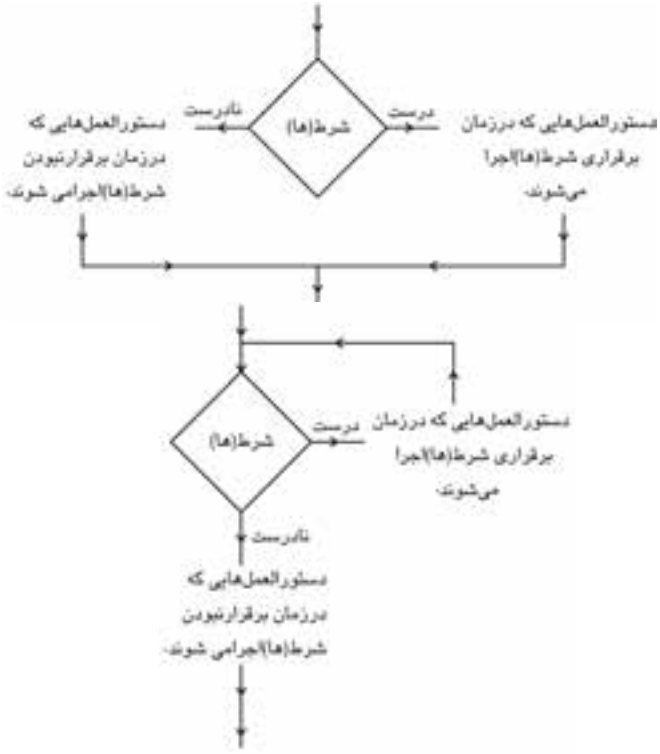
۲-۱-۳ علامت انتساب

برای دستورالعمل‌های محاسباتی از علامت انتساب استفاده می‌شود. این علامت عمل ذخیره‌سازی یک مقدار را در یک متغیر نشان می‌دهد.
به این منظور از شکل مستطیل استفاده می‌شود.
مانند:

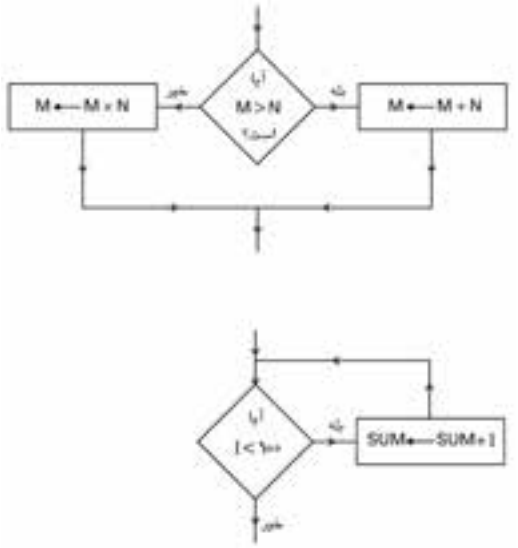


۲-۱-۴ علامت شرط

این علامت می‌تواند جایگزین دستورالعمل‌های شرطی شود و به یکی از صورت‌های بعدی قابل استفاده است.



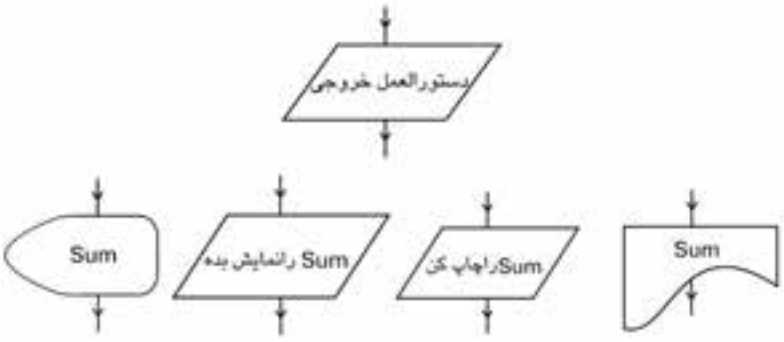
مانند:



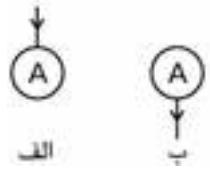
۵–۱–۲ علایم خروجی

با توجه به اینکه اطلاعات خروجی می‌توانند به چاپگر یا صفحه نمایش ارسال شوند از این علایم استفاده می‌شود:

به‌علاوه می‌توانید از علامت متوازی‌الاضلاع به جای دستورالعمل خروجی نیز استفاده کنید.



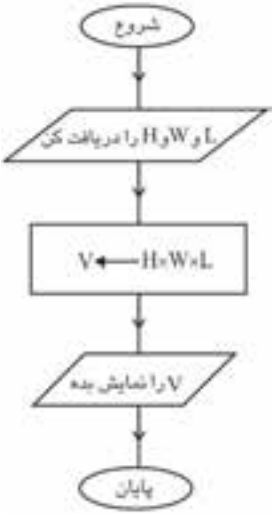
۶-۱-۲ علامت اتصال



گاهی اوقات ممکن است فلوچارت به اندازه‌های بزرگ باشد که در یک صفحه کاغذ قرار نگیرد؛ در چنین شـرایطی می‌توانید از علامت اتصال اسـتفاده کنید (شکل الف) و ادامه فلوچارت را در صفحـه دیگری که آن هم با یک علامت اتصال دیگر شـروع می‌شـود (شـکل ب)، ادامه دهید و در داخل هر دو علامت یک حرف از حروف الفبا یا یک عدد مثبت قرار دهید.

مثـال ۱: الگوریتم و فلوچارتی بنویسـید که وزن و بهای واحد یک کالا را دریافت کرده و بهای کل آن‌را محاسبه کند.

۱ – شروع
۲ – W را دریافت کن
۳ – P را دریافت کن
۴ – P × W را نمایش بده
۵ – پایان



مثـال ۲: الگوریتـم و فلوچارتی بنویسـید که حجم یک مکعب را محاسبه کند.

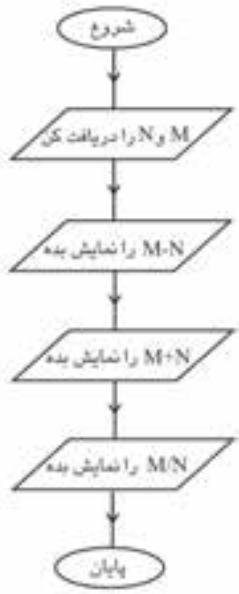
۱ – شروع
۲ ـ L و W و H را دریافت کن
۳ ـ V←H×W×L
۴ –V را نمایش بده
۵ – پایان

فلوچارت این الگوریتم به این صورت خواهد بود:

تمرین:

فلوچارتی رسم کنید که مساحت و محیط یک مربع دلخواه را محاسبه کند.

مثال ۳: الگوریتم و فلوچارتی بنویسید که دو عدد را دریافت کرده و حاصل تفریق، جمع، تقسیم و ضرب آن‌ها را نمایش دهد.



۱ – شروع

۲ – M و N را دریافت کن

۳ – M – N را نمایش بده

۴ – M + N را نمایش بده

۵ – M × N را نمایش بده

۶ – M / N را نمایش بده

۷ – پایان

تمرین:

فلوچارتی رسم کنید که میانگین سه عدد دلخواه را محاسبه کند.

مثال ۴: الگوریتم و فلوچارتی بنویسید که عدد طبیعی و دلخواه M را دریافت کرده و زوج یا فرد بودن آن را معین کند.

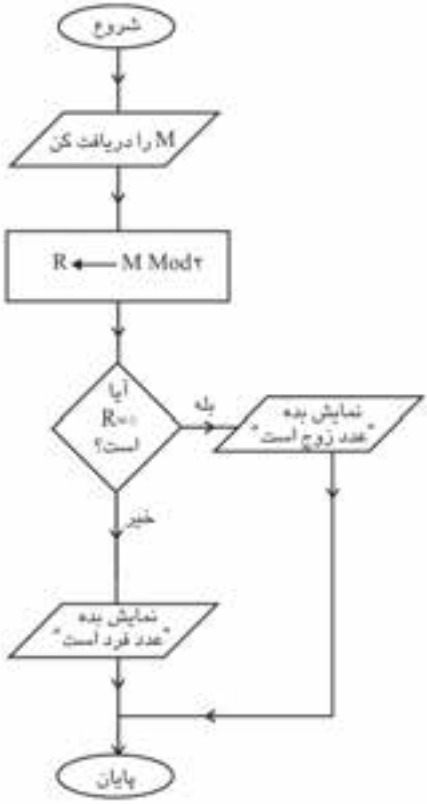
۱ – شروع

۲ – M را دریافت کن

۳– اگر ۰ = ۲ MOD M آن‌گاه «عدد زوج است» را نمایش بده در غیراینصورت «عدد فرد است» را نمایش بده

۴ – پایان

فلوچارت الگوریتم قبل به این صورت خواهد بود:



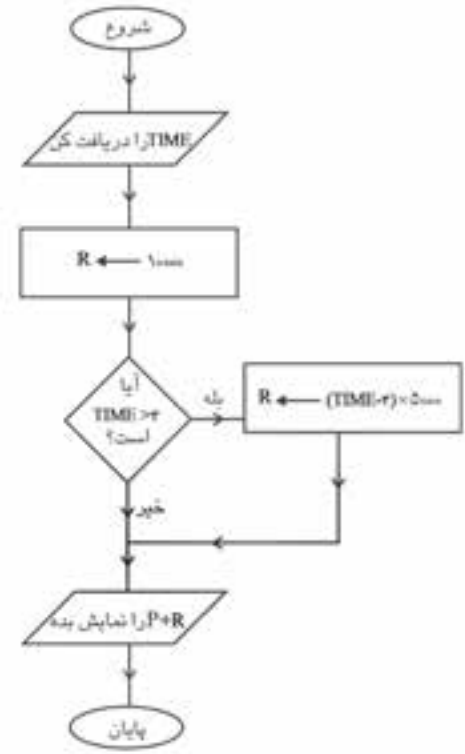
تمرین:

فلوچارتی رسـم کنید که تعدادی عدد دلخواه را دریافت کرده و بزرگ‌ترین مقدار را محاسبه کند و نمایش دهد.

مثال ۵ : فلوچارتی بنویسید که با توجه به شرایط زیر هزینه پارکینگ را برای یک اتومبیل دلخواه محاسبه نموده و نمایش دهد.

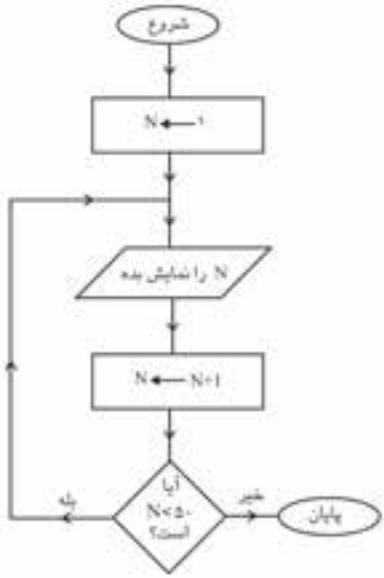
الف– هزینه پارکینگ تا سه ساعت از زمان ورود ۱۰/۰۰۰ریال

ب– هزینه پارکینگ بعد از سه ساعت هر ساعت ۵/۰۰۰ ریال



تمرین:

فلوچارتی رسـم کنید که میانگین دمای یک شهر را در طول یک هفته محاسبه کرده و نمایش دهد.



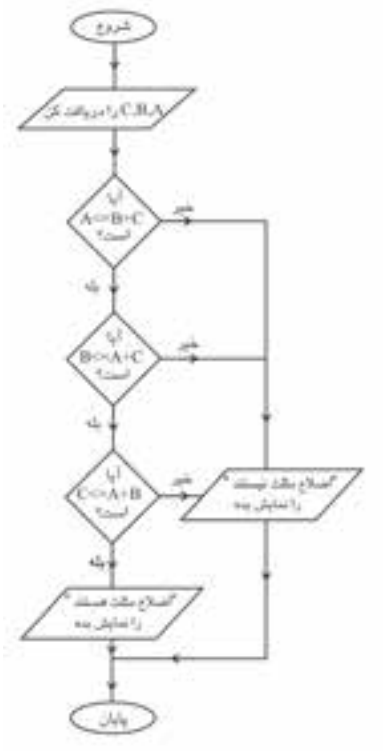
مثال ۶: فلوچارتی رسم کنید که اعداد طبیعی کوچک‌تر از ۵۰ را نمایش دهد.

تمرین:

فلوچارتی رسم کنید که اعداد زوج کوچک‌تر یا مساوی عدد طبیعی و دلخواه N را نمایش دهد.

مثال ۷: فلوچارتی رسم کنید که سه عدد دلخواه را دریافت کرده و معین کند سه عدد می‌توانند اضلاع یک مثلث باشد (شرط مثلث بودن سه عدد این است که:

A<= B+C. یا B<=A+C یا (C<=A+B)



تمرین:

فلوچارتی رسم کنید که دو عدد را دریافت کرده و بخش‌پذیری عدد اول را بر عدد دوم بررسی کند.

مثـال ۸ : الگوریتم و فلوچارتی بنویسـید کـه عدد طبیعـی و دلخـواه m را دریافـت کرده و مقسوم‌علیه‌های آن را نمایش دهد.

۱ – شروع

۲ – M را دریافت کن

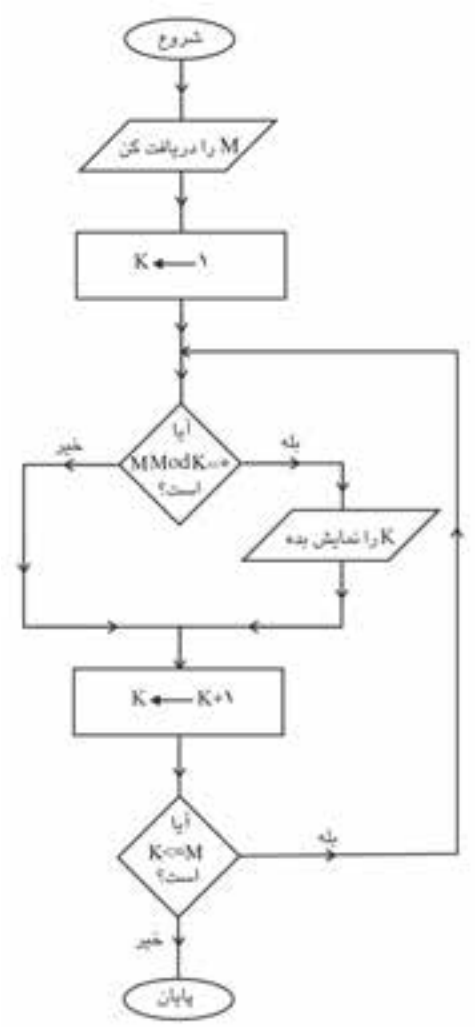
۳ – K ⟵ ۱

۴ – اگـر M Mod K=۰ آن‌گاه K را نمایـش بده

۵ – k ⟵ K + ۱

۶ – اگر K < = M آن‌گاه برو به مرحله ۴

۷ – پایان



تمرین:

فلوچارتی رسم کنید که دو آرایه یک بعدی را دریافت کرده و حاصل‌جمع آن‌ها را در آرایه یک بعدی جداگانه‌ای ذخیره نموده و نمایش دهد.

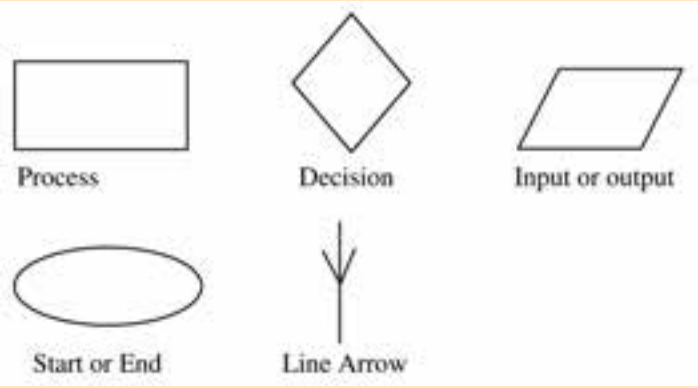
Learn in English

Flowchart Symbols

Flowchrts use special shapes to represent different types of actions or steps in aprocess.

Lines and arrows show the sequence of the steps and therelationships among them.

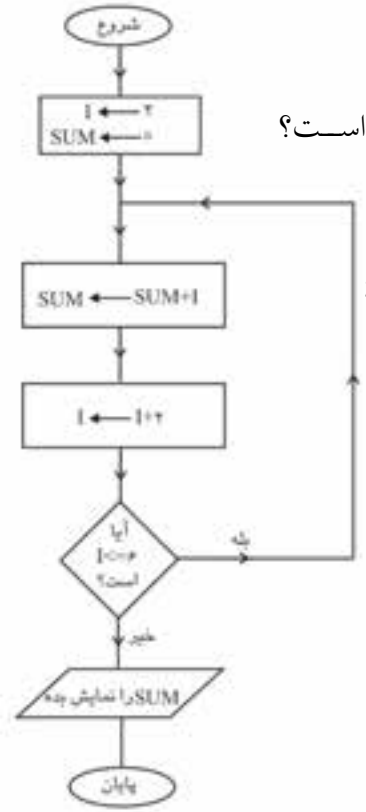
Flowcharts Shapes



واژه‌نامه

Decision	شرطی
Flowchart	مجموعه‌ای از اشـــکال هندســـی که نوع دستورالعمل‌ها و ترتیب اجرای آن‌ها را نمایش می‌دهد.
Lines and arrows	خطوط فلش‌دار
Process	پردازش
Relationship	رابطه
Sequence	ترتیب
Step	مرحله

آزمون نظری



۱ – کدام گزینه در رابطه با اجرای فلوچارت زیر درست است؟

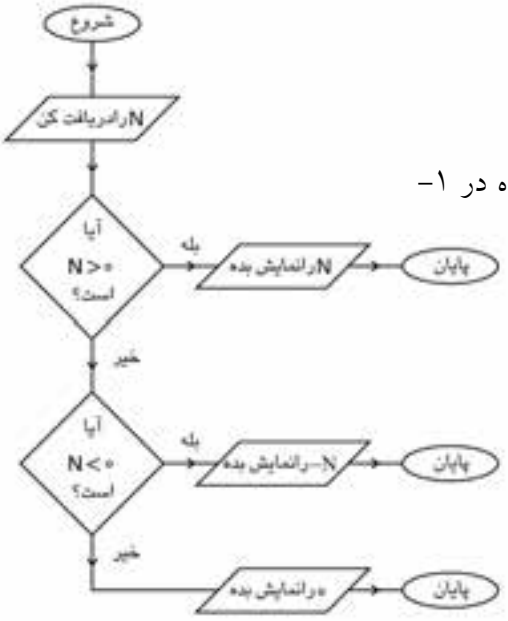
الف– نمایش اعداد زوج کوچک‌تر از ۶

ب– نمایش اعداد زوج مساوی ۶

ج– نمایش مجموع اعداد زوج کوچک‌تر یا مساوی ۶

د– مجموع اعداد زوج کوچک‌تر از ۶

۲ – عملکرد فلوچارت زیر چیست؟



الف– نمایش قرینه یک عدد

ب– نمایش قدرمطلق یک عدد

ج– نمایش عدد به صورت گرد شده

د– نمایش حاصل‌ضرب هر عدد دلخواه در ۱–

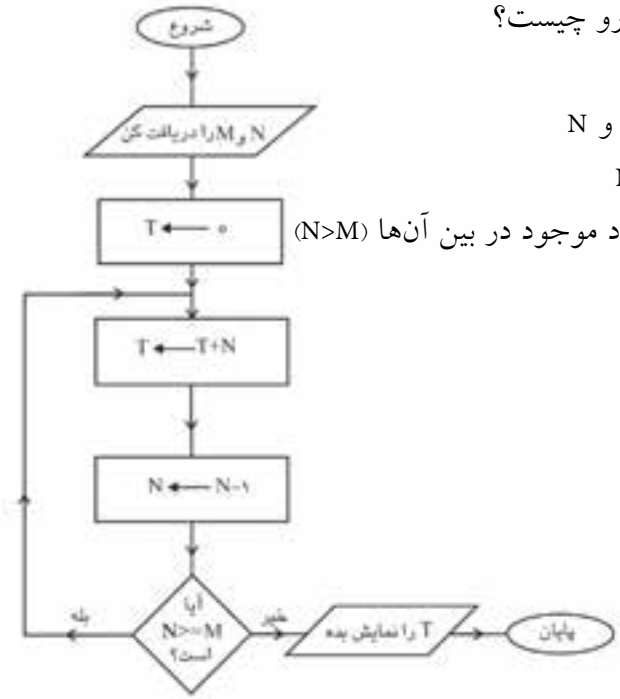
۳ – خروجی فلوچارت روبه‌رو چیست؟

الف– اعداد بین M و N

ب– مجموع اعداد بین M و N

ج– حاصل‌ضرب M در N

د– مجموع M و N و اعداد موجود در بین آن‌ها (N>M)



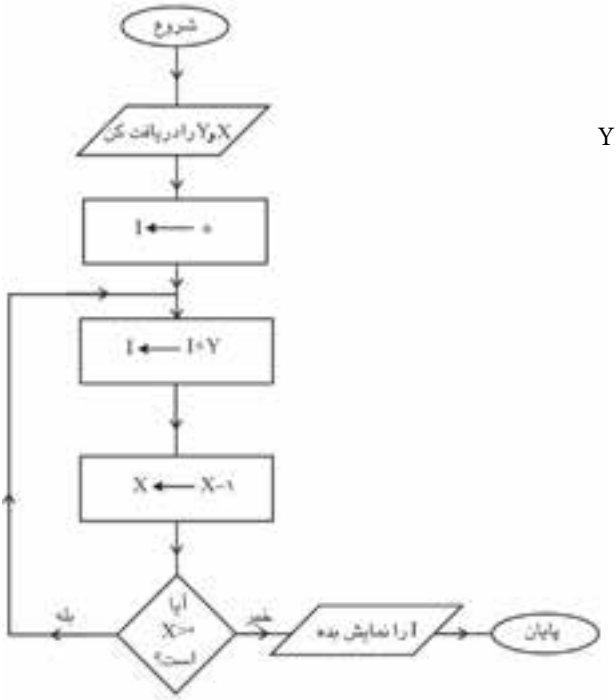
۴ – خروجی این فلوچارت چیست؟

الف– حاصل‌جمع X و Y

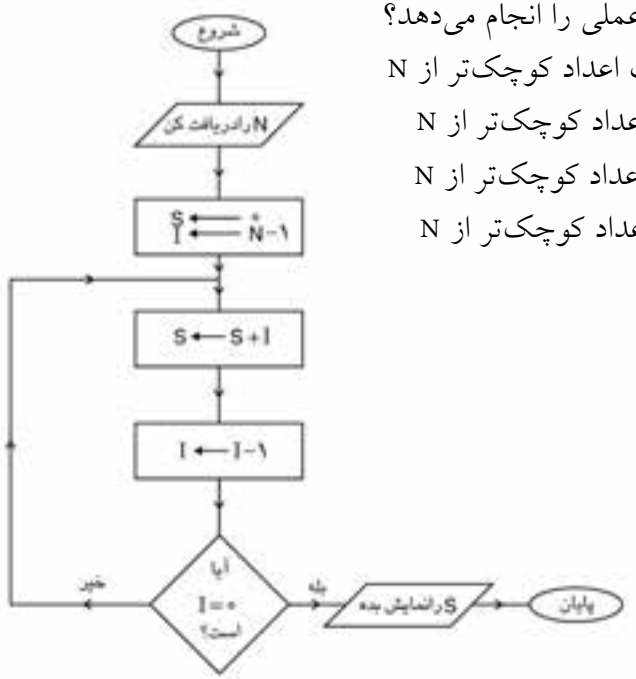
ب– حاصل‌ضرب X در Y

ج– مجموع اعداد بین X و Y

د– باقیمانده تقسیم Y بر X



۵ – فلوچارت روبه‌رو چه عملی را انجام می‌دهد؟



الف– محاسبه حاصل ضرب اعداد کوچک‌تر از N

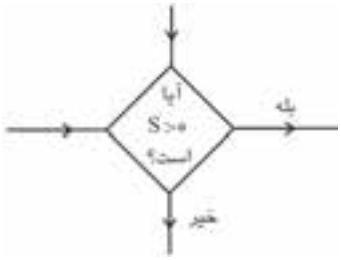
ب– محاسبه حاصل جمع اعداد کوچک‌تر از N

ج– محاسبه حاصل تفریق اعداد کوچک‌تر از N

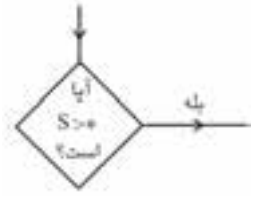
د– محاسبه حاصل تقسیم اعداد کوچک‌تر از N

۶ – کدام گزینه شکل مناسب را برای دستورالعمل شرط در فلوچارت نمایش می‌دهد؟

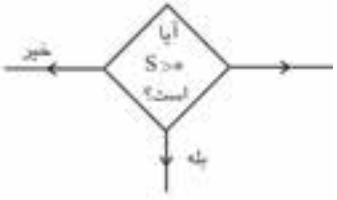
الف –



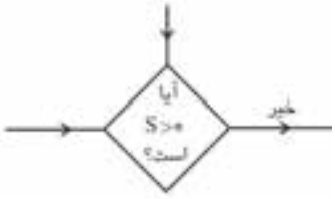
ب –



ج –



د –



۷ – یک فلوچارت می‌تواند حداکثر از یک ..... استفاده کند.

الف– نقطه شروع ب– انتساب ج– نقطه پایانی د– خروجی

۸ – کدام گزینه شکل مناسب برای عمل انتساب را نمایش می‌دهد؟

الف– Z=X+Y ب– X+Y=Z ج– Z←X+Y د– X+Y→Z

۹ – در فلوچارت شکل نشانه چیست؟

الف– پایان　　ب– محاسبه

ج– اتصال　　د– خروجی

10 – The flowcharts use ..... to draw graphical representation of a process.

a– lines　　**b**– shapes

c– rectangles　　**d**– circles & rectangles

۱۱ – مفهوم و کاربرد فلوچارت را توضیح دهید.

۱۲ – شکل کلی نحوه استفاده از حلقه در فلوچارت را توضیح دهید.

۱۳ – شکلی را که برای شروع و پایان فلوچارت استفاده می‌شود، توضیح دهید.

۱۴ – اشکالی را که برای ورودی و خروجی مورد استفاده قرار می‌گیرند، توضیح دهید.

۱۵ – شکلی را که برای استفاده از شرط در فلوچارت استفاده می‌شود، توضیح دهید.

آزمون عملی

۱ – فلوچارتی رسم کنید که مضارب عدد ۵ را که کوچک‌تر از ۵۰۰ هستند، محاسبه کرده و نمایش دهد.

۲ – فلوچارتی رسم کنید که اعداد دو رقمی زوج را نمایش دهد.

۳ – فلوچارتی رسم کنید که میزان حافظه کامپیوتر را براساس مگابایت دریافت کرده و براساس واحدهای دیگر نمایش دهد.

۴ – فلوچارتی رسم کنید که ریشه معادله درجه اول یک مجهولی را محاسبه نموده و نمایش دهد. $ax + b = c$

۵ – فلوچارتی رسم کنید که ساعت کار و دستمزد هر ساعت کار یک کارگر را دریافت کرده و حقوق وی را محاسبه کند.

۶ – فلوچارتی رسم کنید که بتواند با شرایط زیر هزینه استفاده از اینترنت را برای یک مشترک در مدت یک هفته محاسبه کرده و نمایش دهد.

الف– هزینه ۵۰ ساعت اول هر ارتباط ۲۰/۰۰۰ ریال

ب– از ۵۰ ساعت به بالا به ازای هر ساعت ۲/۵۰۰ ریال

۷– فلوچارتی رسم کنید که دو عدد را دریافت کرده و عدد اول را به توان عدد دوم برساند.

واحد کار سوم

هدف جزیی

توانایی درک و شناخت زبان برنامه‌نویسی ویژوال‌بیسیک و ایجاد یک برنامه کاربردی

هدف‌های رفتاری

پس از مطالعه این واحد کار از فراگیر انتظار می‌رود که:

۱ – زبان برنامه‌نویسی را تعریف کند و انواع آن‌را توضیح دهد.

۲ – روش برنامه‌نویسی ساخت یافته و شئ‌گرا را توضیح دهد.

۳– زبان‌های برنامه‌نویسی از نوع مفسر و مترجم را توضیح داده و تفاوت آن‌ها را بیان کند.

۴ – زبان برنامه‌نویسی ویژوال‌بیسیک را توضیح داده و ویژگی‌های آن‌را بیان کند.

۵ – انواع نگارش‌های زبان برنامه‌نویسی ویژوال‌بیسیک را توضیح دهد.

۶ – اجزای تشکیل‌دهنده یک برنامه در ویژوال‌بیسیک را بیان کند.

۷ – وارد محیط ویژوال‌بیسیک شده با اجزای آن کار کند.

۸ – پروژه و فرم‌های مورد نیاز خود را ایجاد و طراحی نماید.

۹ – خصوصیات مربوط به فرم مانند Name و Caption را توضیح داده و با کنترل‌های برچسب و کادر متن کار کند.

۱۰ – با پنجره کدنویسی کار کرده و مفهوم رویداد را بیان کند.

۱۱ – خصوصیات فرم و کنترل‌ها را با استفاده از کدنویسی تغییر دهد.

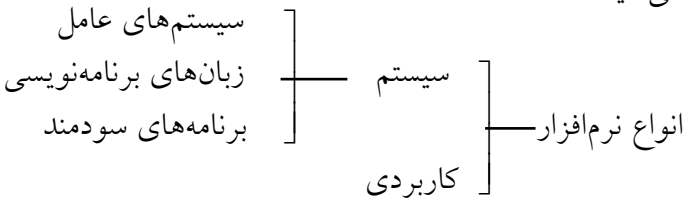
۱۲ – با رویداد Load فرم و رویداد Change کنترل کادر متن کار کند.

۱۳ – فرم و پروژه را ذخیره کرده و برای پروژه نامی تعیین کند.

کلیات

کامپیوتر از دو جزء اصلی سخت‌افزار و نرم‌افزار تشکیل شده است، برای استفاده از هر کامپیوتر لازم است تا داده‌ها و دستورالعمل‌ها برای پردازش به آن داده شود و نتیجه پردازش داده‌ها یعنی اطلاعات ارایه شود یا به عبارت دیگر کاربر بتواند با سخت‌افزار ارتباط برقرار کند، در این‌جاست که نقش نرم‌افزار به عنوان یکی از اجزای اصلی در کامپیوتر کاملاً قابل مشاهده است. در این واحد کار با تعریف و تاریخچه تولید و طراحی زبان‌های برنامه‌نویسی آشنا خواهید شد.

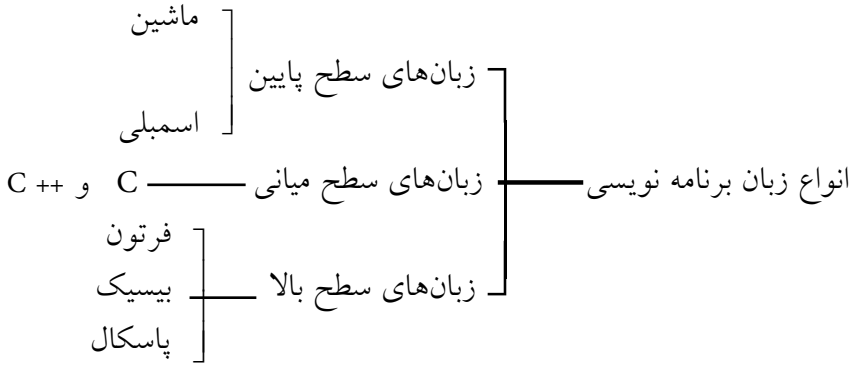
نرم‌افزارها مجموعه‌ای از داده‌ها و دستورالعمل‌ها هستند که به وسیله برنامه‌نویس و براساس قواعد مشخص نوشته می‌شوند و سخت‌افزار را قابل استفاده می‌کنند. نرم‌افزارها به دو دسته کلی سیستمی و کاربردی تقسیم می‌شوند، در شکل زیر انواع نرم‌افزارها را مشاهده می‌کنید:



۱–۳ تقسیم‌بندی زبان‌های برنامه‌نویسی

به مجموعه‌ای از قواعد و دستورالعمل‌های تعریف شده، زبان برنامه‌نویسی می‌گویند.

به طور کلی می‌توان زبان‌های برنامه‌نویسی را به صورت زیر تقسیم‌بندی کرد:



زبان‌های برنامه‌نویسی با توجه به نزدیکی که به زبان ماشین یا همان ۰ و ۱ دارند به سه دسته تقسیم می‌شوند؛ زبان‌های سطح پایین بیشتر به زبان ماشین نزدیک هستند و با ظهور اولین نسل از کامپیوترها این زبان برنامه‌نویسی مورد استفاده قرار گرفت که برنامه‌نویسی با آن نیز کار مشکلی است. با ساخت نسل دوم کامپیوترها، زبان دیگری به نام زبان اسمبلی به وجود آمد که این زبان نیز به زبان ماشین نزدیک بود، ولی استفاده از آن ساده‌تر از زبان ماشین است. پس از نسل دوم و ارایه نسل سوم از کامپیوترها، زبان‌های سطح میانی و سطح بالا به وجود آمدند که به زبان‌های محاوره‌ای و نوشتاری نزدیک‌تر بوده و برنامه‌نویسی با آن‌ها به مراتب راحت‌تر از زبان‌های سطح پایین می‌باشد. از آن زمان تاکنون کیفیت و کمیت زبان‌های برنامه‌نویسی تغییرات زیادی کرده است و برای تهیه برنامه‌ها در محیط‌ها و کاربردهای مختلف، زبان‌های برنامه‌نویسی متفاوتی استفاده می‌شود.

۱-۱-۳ زبان‌های برنامه‌نویسی سطح بالا

زبان برنامه‌نویسی فرترن: یکی از اولین زبان‌های برنامه‌نویسی سطح بالاست و در برنامه‌نویسی‌های علمی که نیاز به دقت بالا در محاسبات است، از آن استفاده می‌شود. از این زبان نسخه‌هایی نیز برای برنامه‌نویسی تحت سیستم عامل ویندوز طراحی شده است.

زبان برنامه‌نویسی پاسکال: این زبان یک زبان برنامه‌نویسی سطح بالا با ویژگی ساخت یافته است و برای برنامه‌نویسی‌های علمی و تجاری مورد استفاده قرار می‌گیرد. نسخه پیشرفته آن با نام دلفی دارای ویژگی‌های برنامه‌نویسی برای محیط ویندوز است.

زبان برنامه‌نویسی C: این زبان نیز از نوع زبان‌های برنامه‌نویسی سطح بالا با ویژگی ساخت یافته است و از توانایی بالایی در سطوح مختلف برنامه‌نویسی برخوردار است. به عبارت دیگر این زبان، زبان برنامه‌نویسی سیستم است و با آن می‌توان هر نوع نرم‌افزاری را طراحی و تولید کرد.

نسخه دیگری از این زبان با نام C++ علاوه بر ویژگی‌های زبان C، دارای قابلیت برنامه‌نویسی شیءگرا نیز می‌باشد. نسخه‌های دیگری از این زبان با نام Visual C++ و C#.Net برای برنامه‌نویسی در محیط ویندوز طراحی و ارایه شده‌اند.

زبان برنامه‌نویسی ویژوال‌بیسیک: زبان ویژوال‌بیسیک در واقع حاصل توسعه و ارتقای زبان بیسیک[1] است. بیسیک اولیه در حدود سال ۱۹۶۴ در کالج دارت موث[2] به وسیله آقایان توماس کورتز[3] و جان کمنی[4] با هدف گسترش برنامه‌نویسی بین دانش‌آموزان و دانشجویان طراحی و ساخته شد. از آن زمان نسخه‌های متعدد و متفاوتی از آن مانند GWBASIC، QUICK BASIC ارایه شد و همواره سعی در افزایش قابلیت‌های آن به عنوان یک زبان سطح بالا شده است. با ارایه سیستم عامل ویندوز ۹۵ و ۹۸، فقدان یک زبان برنامه‌نویسی آسان و قدرتمند برای استفاده در سیستم عامل‌های مذکور کاملاً مشهود بود، از این رو مایکروسافت در سال ۱۹۹۱ نسخه اول ویژوال‌بیسیک را با امکانات یک زبان برنامه‌نویسی قدرتمند و حرفه‌ای برای برنامه‌نویسی در ویندوز ارایه کرد. آخرین نسخه تکامل یافته آن برای استفاده در سیستم عامل جدید مایکروسافت نیز با نام VISUAL BASIC NET طراحی و ارایه شده است. در این پیمانه مهارتی نحوه برنامه‌سازی با زبان برنامه‌نویسی ویژوال‌بیسیک نسخه ۶ را فراخواهید گرفت، اما قبل از هر چیز به بررسی ویژگی‌های این زبان برنامه‌نویسی خواهیم پرداخت. یکی از مهم‌ترین ویژگی‌های زبان برنامه‌نویسی ویژوال‌بیسیک رابط گرافیکی آن است. رابط گرافیکی کاربر (GUI)[5] در ویژوال‌بیسیک یکی از کارآمدترین رابط‌های گرافیکی در زمینه برنامه‌نویسی است که به وسیله آن می‌توان به آسانی برنامه‌های تحت سیستم عامل ویندوز را ایجاد کرده و حتی قبل از اجرا، شکل ظاهری آن را مشاهده کرد یا این‌که برنامه را به صورت یک مفسر یعنی به صورت خط به خط اجرا نموده و عکس‌العمل برنامه را بررسی‌کرد. البته این موارد گوشه‌ای از ویژگی‌های متعدد رابط گرافیکی ویژوال‌بیسیک است. توسعه سریع برنامه (RAD)[6] یکی دیگر از ویژگی‌های این زبان است. منظور از توسعه سریع برنامه یا RAD در ویژوال‌بیسیک این است که طراحی و تولید برنامه‌ها در ویژوال‌بیسیک به دلیل وجود ابزارهای مناسب به سرعت انجام می‌شود، بنابراین هزینه‌های تولید نرم‌افزار به طور قابل توجهی کاهش می‌یابد.

ویژگی دیگر زبان برنامه‌نویسی ویژوال‌بیسیک ویژگی مدیریت رویدادها[7] واتفاقات می‌باشد.ویژوال‌بیسیک یکی از زبان‌های برنامه‌نویسی رویدادگر است.

1 –BASIC (Beginner's All purpose Symbolic Instruction Code)
2 –Darth Mouth
3 –Thomas Kurtz
4 - John Kemeny
5 -Graphical User Interface
6 – Rapid Application Development
۷– Event Handling

مزیتی که این گونه زبان‌ها دارند در این است که برنامه‌نویس می‌تواند از قبل دستورات لازم را برای رویدادهایی که ممکن است در هنگام اجرای برنامه توسط کاربر رخ دهد، برنامه‌سازی کند.

وجود محیط IDE[1] نیز یکی از ویژگی‌های مهم این زبان است. محیط IDE به برنامه‌نویس اجازه می‌دهد تا برنامه‌های خود را به سهولت و سرعت طراحی، تولید، خطایابی و اجرا کند. این امکانات به وسیله ابزارهای متعددی که به صورت یکپارچه در رابط گرافیکی ویژوال‌بیسیک قرار داده شده است، قابل دسترسی است.

علاوه بر مواردی که گفته شد، دسترسی به برنامه‌های کاربردی ویندوز به وسیله توابع یکی دیگر از ویژگی‌های این زبان است. توابع API، توابع داخلی ویندوز هستند که ویژوال‌بیسیک را قادر می‌سازد تا با استفاده از فرامین خاصی بتواند به امکانات داخلی موجود در ویندوز دستیابی پیدا کند و برنامه‌نویس را نیز قادر می‌سازد تا در صورت نیاز با استفاده از این توابع، برنامه‌هایی را با توانایی‌های مورد نظر ایجاد کند.

یکی دیگر از جنبه‌هایی که تفاوت شگرفی بین ویژوال‌بیسیک و سایر نسخه‌های قبلی بیسیک ایجاد می‌کند، امکان استفاده از برنامه‌نویسی به‌روش شئ‌گراست. این ویژگی سبب می‌شود تا ویژوال‌بیسیک بتواند خواسته‌های برنامه‌نویس در رابطه با تعریف و به کارگیری اشیا و کلاس‌های جدید را که سبب راحت‌تر شدن برنامه‌نویسی می‌شود، برطرف کند. در برنامه‌نویسی ساخت‌یافته، برنامه‌ها با استفاده از رویه‌ها به بخش‌های مختلف تقسیم می‌شوند که به صورت مجزا از هم قرار می‌گیرند. در برنامه‌نویسی شئ‌گرا با استفاده از اشیا می‌توان مجموعه‌ای از دستورالعمل‌ها و داده‌ها را در عنصر واحدی به نام شئ قرار داد و در زمان مورد نظر از هر یک از بخش‌های شئ مربوطه استفاده کرد.

ویژگی دیگری که در نحوه کار با یک زبان برنامه‌نویسی مد نظر قرار می‌گیرد نحوه کشف، تصحیح و برخورد با اشتباهات و خطاهایی است که در هنگام طراحی یا اجرای برنامه‌ها رخ می‌دهد. ویژوال‌بیسیک علاوه بر این‌که امکانات بسیار مناسبی در زمینه کشف خطاهای نوشتاری و منطقی برنامه در اختیار برنامه‌نویس می‌گذارد، به وی امکان می‌دهد با استفاده از فرامین مناسب، خطاهای غیرقابل‌پیش‌بینی را نیز در هنگام اجرا تشخیص داده و نحوه ارایه راه‌حل مناسب را برای راهنمایی کاربران در اختیار آنان قرار دهد.

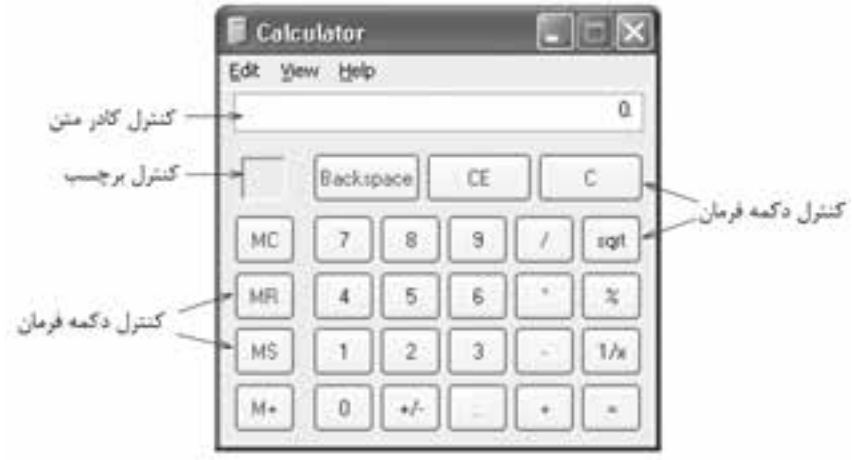
نگارش حرفه‌ای ویژوال‌بیسیک علاوه بر ویژگی‌های نگارش آموزشی، امکان استفاده از کنترل‌های مربوط به بانک‌های اطلاعاتی، طراحی کنترل‌های ActiveX و هم‌چنین

۱ - Integrated Development Environment

به‌کارگیری ویزاردهای مناسب برای تسهیل امر برنامه‌نویسی را نیز در اختیار برنامه‌نویسان قرار می‌دهد و در آن امکان برنامه‌نویسی تحت شبکه برقراری ارتباط با بانک‌های اطلاعاتی را دارد.

۲ – ۳ اجزای تشکیل‌دهنده یک برنامه

آنچه که قبل از طراحی و ایجاد یک برنامه کاربردی لازم است بدانید اجزای تشکیل دهنده یک برنامه است. در واقع اصلی‌ترین جزء تشکیل دهنده یک برنامه کاربردی در ویژوال‌بیسیک، پروژه (Project) است که از اجزای کوچک‌تری نظیر فرم‌ها، ماژول کد و ماژول کلاس تشکیل می‌شوند. فرم‌ها در واقع همان پنجره‌ها هستند که در سیستم عامل ویندوز از اجزای اصلی یک برنامه به شمار می‌روند و اطلاعات مربوط به شکل ظاهری برنامه و کنترل[1]های موجود در آن را نگهداری می‌کنند. در شکل ۱–۳ پنجره برنامه ماشین حساب را به همراه کنترل‌های موجود روی آن مشاهده می‌کنید. از کنترل‌ها به منظور هدایت عملکرد برنامه، انجام عملیات و تنظیمات موردنظر در برنامه‌ها استفاده می‌شود، عناصری مانند دکمه‌های فرمان، کادرهای متن، کادرهای لیست، دکمه‌های انتخاب و کادرهای علامت نمونه‌هایی از کنترل هستند. ماژول‌های کد، قطعاتی شامل کد هستند که دستورالعمل‌های مورد نیاز برنامه در آن‌ها قرار دارد. ماژول‌های کلاس نیز مانند ماژول‌های کد شامل قطعات کد هستند و از آن‌ها در برنامه‌نویسی شئ‌گرا برای تعریف انواع کلاس‌ها استفاده می‌شود.



شکل ۱–۳

1 -Control

نکته یک برنامه کاربردی می‌تواند از چند پروژه تشکیل شود.

۳-۳ نحوه‌اجرای برنامه‌ویژوال‌بیسیک و معرفی‌اجزای‌موجود در آن

قبل از هرگونه اقدام برای طراحی یک پروژه لازم است نحوه اجرای برنامه ویژوال‌بیسیک و اجزای موجود در پنجره آن را فرا بگیرید، به این منظور مراحل زیر را به ترتیب اجرا کنید:

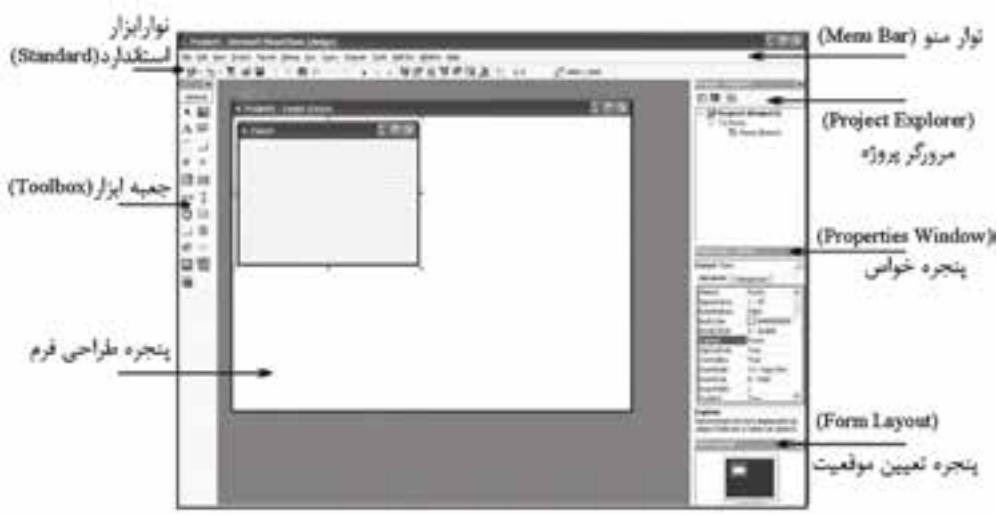
برای اجرای برنامه ویژوال‌بیسیک از طریق منوی Start به گروه برنامه‌ها All Programs بروید و در آنجا گزینه Microsoft Visual Studio 6.0 را انتخاب نموده و سپس از منویی که ظاهر می‌شود روی میانبر Microsoft Visual Basic 6.0 کلیک کنید. پنجره اصلی برنامه به همراه کادر محاوره New Project نمایش داده خواهد شد (شکل ۲-۳).

شکل ۲-۳

در کادرمحاوره New Project سه زبانه به همراه سه دکمه و یک کادر لیست مشاهده می‌شود. در کادر محاوره New Project روی زبانه New کلیک کنید و در کادر لیست

موجود در آن آیکن Standard EXE را انتخاب کنید، سپس روی دکمه Open کلیک کنید (شکل ۲–۳). پروژه‌هایی که از نوع Standard EXE طراحی می‌شوند به شما اجازه می‌دهند که از پروژه ایجاد شده، فایل‌های اجرایی مستقل برای اجرا در محیط ویندوز طراحی کنید که به آن‌ها Application نیز می‌گویند.

پس از انجام مرحله دوم، پنجره ویژوال‌بیسیک مطابق شکل ۳–۳ نمایش داده می‌شود.



شکل۳-۳ پنجره طراحی برنامه در حالت Standard EXE

همان‌طور که در شکل ۳–۳ مشاهده می‌کنید در پنجره ویژوال‌بیسیک اجزا و بخش‌های مختلفی وجود دارد.

پنجره طراحی فرم: به وسیله پنجره طراحی فرم که در قسمت مرکزی پنجره ویژوال‌بیسیک قرار دارد می‌توان تغییرات لازم را روی فرم برنامه که در داخل پنجره طراحی است، اعمال کرد.

پنجره طراحی فرم را می‌توانید با استفاده از دکمه View Object در پنجره پروژه، فشردن کلید ترکیبی Shift+F7 یا انتخاب گزینه Object از منوی View فعال کنید.

پنجره پروژه(Project Explorer): در قسمت بالا و سمت راست پنجره ویژوال‌بیسیک پنجره یا مرورگر پروژه قرار دارد که اجزای تشکیل دهنده برنامه مانند پروژه‌ها و فرم‌ها

را به صورت ساختار درختی نمایش می‌دهد. با استفاده از این بخش برنامه‌نویس علاوه بر مشاهده نمای کلی از اجزای تشکیل دهنده برنامه خود، توانایی دسترسی، ویرایش، ذخیره‌سازی یا حذف هر یک از اجزا را نیز دارد. پنجره پروژه را می‌توانید با استفاده از دکمه Project Explorer در نوار ابزار استاندارد، فشردن کلید ترکیبی Ctrl+R یا انتخاب گزینه Project Explorer از منوی View فعال کنید.

پنجره خصوصیات (Properties Window): در زیر پنجره پروژه، پنجره خصوصیات قرار گرفته است که به وسیله آن می‌توانید خصوصیات و ویژگی‌های مربوط به فرم‌ها و کنترل‌ها را مشاهده و تنظیم کنید.

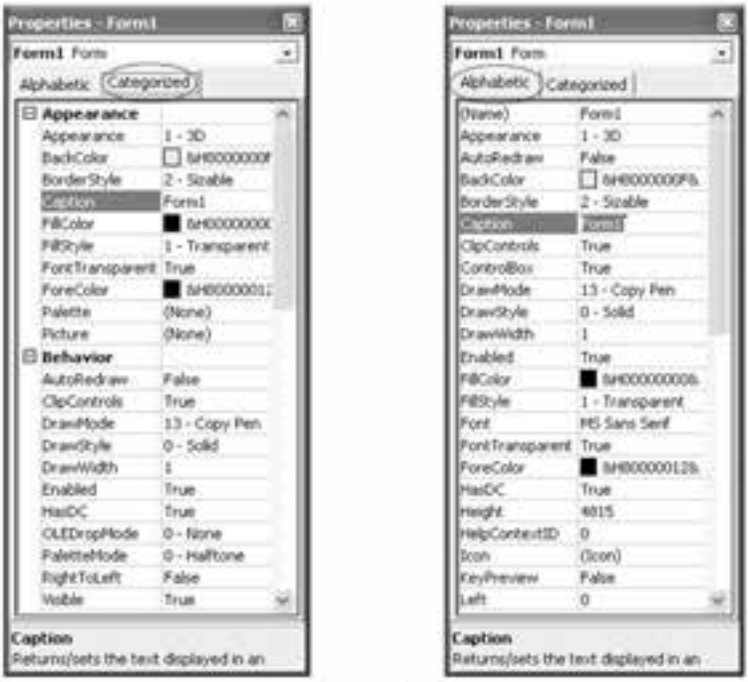
خصوصیت، یک ویژگی از فرم یا کنترل است که می‌توانید با تنظیم این ویژگی، فرم و کنترل را مطابق میل خود طراحی و استفاده کنید. خصوصیات و ویژگی‌های یک فرم یا کنترل می‌توانند انواع مختلفی از داده‌های عددی و غیر عددی را با توجه به وظیفه‌ای که به عهده دارند، کسب کنند.

این خصوصیات می‌توانند شکل ظاهری مثل رنگ عناوین، رنگ زمینه، ابعاد کنترل یا فرم را تنظیم کنند. هم‌چنین می‌توانند روی عملکرد کنترل یا فرم مثل فعال بودن یا مخفی شدن آن‌ها تأثیر بگذارند که به مرور با تعداد قابل توجهی از آن‌ها به طور کامل آشنا خواهید شد.

پنجره خصوصیات را می‌توانید با استفاده از دکمه Properties Window در نوار ابزار استاندارد، فشردن کلید F4 یا انتخاب گزینه Properties Window از منوی View فعال کنید.

در پنجره خصوصیات با استفاده از دکمه‌های Alphabetic و Categorized می‌توانید نحوه نمایش اسامی خصوصیات را به‌صورت الفبایی یا موضوعی تعیین کنید (شکل ۴-۳).

وقتی یک خصوصیت را انتخاب می‌کنید در قسمت پایینی پنجره خصوصیات توضیحات خلاصه‌ای در رابطه با خصوصیت انتخاب شده ارایه می‌شود.



شکل ۴–۳

پنجره تعیین موقعیت (Form Layout Window) : به طور معمول در زیر پنجره خصوصیات قرار دارد و موقعیت نمایش پنجره برنامه را روی دسک‌تاپ و در زمان اجرا نشان می‌دهد؛ به علاوه با استفاده از این پنجره می‌توانید موقعیت پنجره‌های برنامه را روی دسک‌تاپ تنظیم کنید.

پنجره تعیین موقعیت را می‌توانید با استفاده از دکمه Form Layout در نوار ابزار استاندارد یا انتخاب گزینه Form Layout Window از منوی View فعال کنید.

جعبه‌ابزار (Toolbox) : در سمت چپ پنجره ویژوال‌بیسیک و طراحی فرم، جعبه ابزار (Toolbox) حاوی آیکن کنترل‌های مختلف وجود دارد.

جعبه ابزار را می‌توانید با استفاده از دکمه Toolbox در نوار ابزار استاندارد یا انتخاب گزینه Toolbox از منوی View فعال کنید.

مثال ۱: یک پروژه از نوع Standard EXE به همراه یک فرم مطابق شکل ۵–۳ و جدول ۱–۳ طراحی کنید که یک پنجره معمولی را روی دسک‌تاپ نمایش دهد.

شکل ۵-۳

جدول ۱-۳

خصوصیت	مقدار
Height	۳۶۰۰
Width	۵۲۵۰

۱ – برنامه ویژوال‌بیسیک را اجرا کنید و در کادرمحاوره New Project آیکن Standard EXE را انتخاب کنید، سپس روی دکمه Open کلیک کنید.

۲ – در پنجره پروژه روی علامت ⊞ در کنار پوشه Forms کلیک کنید تا آیکن فرم برنامه نمایان شود، سپس روی آیکن فرم در پنجره پروژه دابل کلیک کنید تا پنجره طراحی فرم نمایش داده شود.

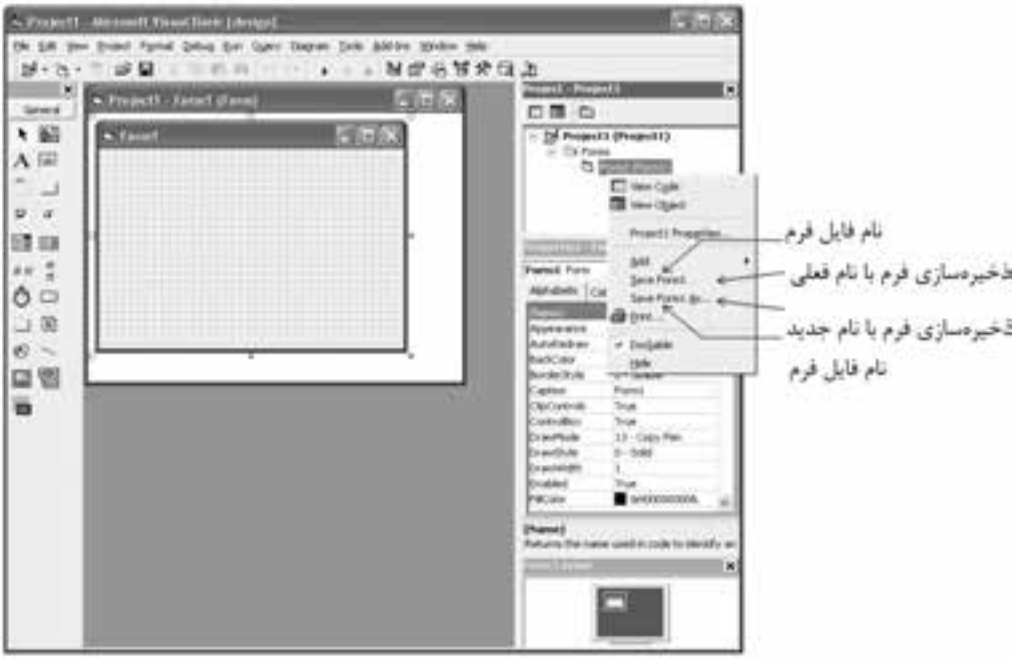
۳ – برای تنظیم عرض و ارتفاع فرم به پنجره خصوصیات بروید و در این پنجره خصوصیات Height را پیدا کنید و پس از کلیک روی نام خصوصیت مقدار ۳۶۰۰ را تایپ کنید، به همین صورت مقدار خصوصیت Width را روی ۵۲۵۰ تنظیم کنید (جدول ۱-۳). البته برای تنظیم عرض و ارتفاع فرم می‌توانید از عمل Drag روی گیره‌های مربعی شکل که در اطراف فرم قرار دارند، استفاده کنید و ابعاد فرم را در انتهای نوار ابزار استاندارد مشاهده کنید (شکل ۶-۳).

شکل ۶-۳

۴ – برای ذخیره‌سازی فرم در نوار منوی پنجره ویژوال‌بیسیک ابتدا روی منوی File و سپس گزینه Save Form1 As... کلیک کنید. کادرمحاوره Save File As... برای ذخیره‌سازی فرم، نمایش داده می‌شود. همان‌طورکه در این کادرمحاوره مشاهده می‌کنید، پسوند فایل فرم، frm است. فرم را با نام Main در پوشه‌ای با نام MyPrograms روی دسک‌تاپ (یا هر مسیر دلخواهی که مایل هستید) ذخیره کنید.

البته می‌توانید فرم‌ها را با یکی از این روش‌ها نیز ذخیره کنید:

الف–در پنجره پروژه، آیکن فرم را انتخاب کرده و روی آن کلیک راست کنید، سپس گزینه Save Form1 As... را برگزینید (شکل ۷–۳).



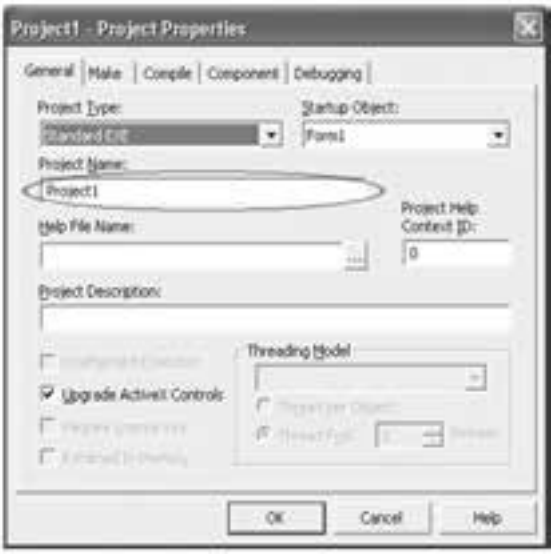
شکل ۷–۳

ب– گزینه Save Form1 را از منوی File انتخاب کنید یا از کلید ترکیبی Ctrl+S استفاده کنید.

۵– در این مرحله نامی برای پروژه تعیین کنید. برای تعیین نام پروژه در پنجره برنامه ویژوال‌بیسیک روی منوی Project کلیک کنید و سپس گزینه Project Properties... را برگزینید تا کادر محاوره Project Properties مطابق شکل ۸–۳ نمایش داده شود. هر پروژه یک خصوصیت Name دارد که به وسیله آن امکان دسترسی و شناسایی پروژه در زمان کدنویسی فراهم می‌شود. برای تعیین یک عبارت برای نام پروژه می‌توانید از حروف

الفبای انگلیسی، ارقام صفر تا ۹ و کاراکتر زیرخط استفاده کنید که باید با یک کاراکتر حرفی آغاز شود زیرا استفاده از کاراکترهای نقطه، فضای خالی و کاراکتر تفریق (–) مجاز نیست.

۶ – در این کادر محاوره روی زبانه General کلیک کنید، سپس در کادر متن Project Name عبارت FirstProgram را تایپ کنید و در پایان روی دکمه OK کلیک کنید.



شکل۸-۳

۷– برای ذخیره‌سازی پروژه، از منوی File گزینه Save Project As... را برگزینید.

۸ – با استفاده از کادر محاوره Save Project As فایل پروژه را با نام MyProject در پوشه MyPrograms روی دسک‌تاپ یا هر مسیر دلخواهی ذخیره کنید. همان‌طورکه در این کادر محاوره می‌بینید، پسوند فایل‌های پروژه vbp است. فایل پروژه اطلاعات اجزای تشکیل‌دهنده پروژه را روی دیسک با نام فایلی که تعیین می‌شود، نگهداری می‌کند. زمانی‌که لازم باشد پروژه مورد استفاده قرار بگیرد با انتخاب و بازکردن فایل پروژه، فایل‌های مربوط به فرم‌ها و ماژول‌های تشکیل‌دهنده پروژه نیز به‌طور خودکار باز می‌شوند.

البته می‌توانید پروژه‌ها را با یکی از این روش‌ها نیز ذخیره کنید:

الف–در پنجره پروژه آیکن پروژه را انتخاب کرده و روی آن کلیک راست کنید، سپس گزینه Save Project را برگزینید.

ب– گزینه Save Project را از منوی File انتخاب کنید.

ج– روی دکمه Save در نوار ابزار استاندارد کلیک کنید.

نکته در زمان ذخیره کردن پروژه و در صورتی که فرم را قبلا ذخیره نکرده باشید، ابتدا کادر محاوره ذخیره‌سازی فرم نمایش داده می‌شود.

۹ – تا این مرحله توانستید یک پروژه را به همراه یک فرم ساده ایجاد کنید. اکنون پنجره ویژوال‌بیسیک را با استفاده از دکمه Close ببندید. البته می‌توانید گزینه Exit را از منوی File نیز انتخاب کنید.

تمرین:
یک پروژه از نوع Standard EXE را که شامل یک فرم مطابق جدول ۲–۳ باشد، طراحی کنید سپس فرم و پروژه را به ترتیب با نام‌های Myform و Test ذخیره کنید. در انتها نام پروژه را First Project بگذارید.

مثال ۲: اکنون عبارت VISUAL BASIC 6.0 را روی فرمی که در مثال قبل ایجاد کردید، مطابق شکل ۹–۳ و جدول ۳–۳ نمایش دهید. به این منظور مراحل بعد را به ترتیب انجام دهید:

جدول ۲–۳ خصوصیات فرم

نام خصوصیت	مقدار
Height	۲۵۰۰
Left	۳۵۰۰
Top	۳۰۰۰
Width	۴۵۰۰
Caption	First Form

شکل ۹–۳

جدول ۳–۳

کنترل / خصوصیت	Label
Name	lblmessage
Height	۶۱۵
Left	۹۶۰
Top	۱۴۴۰
Width	۲۶۵۵
Caption	VISUALBASIC6,0
Alignment	2-Center

۱ – برای نمایش یک پیام روی فرم می‌توانید از کنترل برچسب (Label) استفاده کنید. بـرای قـراردادن کنترل روی فرم، ابتدا روی آیکن A در جعبه ابزار کلیک کنید، سـپس اشـاره‌گر ماوس را روی فرم قرار داده و با استفاده از عمل درگ، کنترل را روی فرم قرار دهید. البته برای قرار دادن یک کنترل روی فرم می‌توانید در جعبه ابزار روی آیکن کنترل موردنظر دابل‌کلیک کنید، سپس در پنجره خصوصیات و با استفاده از خصوصیات Width و Height عرض و ارتفاع کنترل را روی مقادیر ۲۶۵۵ و ۶۱۵ تنظیم کنید (شـکل ۱۰–۳). هنـگام ایجاد کنترل می‌توانید عـرض، ارتفاع و مختصات محـل قرارگیری کنترل را در انتهای نوار ابزار اسـتاندارد مشـاهده کنید. در ضمن می‌توانید عرض و ارتفاع کنترل را با استفاده از عمل Drag روی گیره‌های مربعی اطراف کنترل نیز تنظیم کنید.

نکته بـرای قرار دادن کنترل‌ها روی فـرم می‌توانید روی آیکن کنترل مورد نظر در جعبه ابزار ویژوال‌بیسیک دابل کلیک کنید.

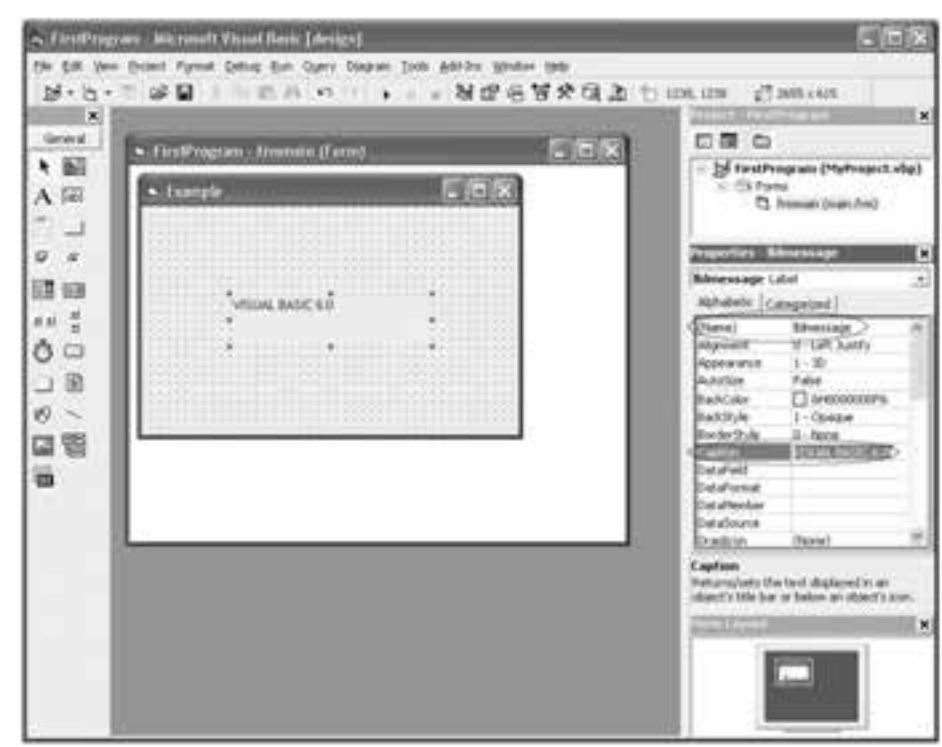
شکل ۱۰–۳

نکته بـرای تنظیم محـل قرارگرفتن یک کنترل می‌توانید از خصوصیات Top و Left در پنجره خصوصیات استفاده کنید.

۲ – در پنجره خصوصیات روی خصوصیت Name کنترل برچسب کلیک کنید و در کادر متن روبه‌روی آن عبارت lblmessage را به عنوان نام کنترل برچسب تایپ کنید و سپس کلید Enter را بفشارید (شکل ۱۱–۳).

نکته توصیه می‌شود در نام‌گذاری کنترل برچسب از پیشوند lbl استفاده کنید.

۳ – در پنجره خصوصیات روی خصوصیت Caption کنترل برچسب کلیک کرده و در کادر متن روبه‌روی آن عبارت VISUAL BASIC 6.0 را تایپ کنید، همان‌طور که می‌بینید خصوصیت Caption این کنترل عبارتی را که نمایش داده می‌شود، نگهداری می‌کند (شکل ۱۱–۳).



شکل ۱۱-۳

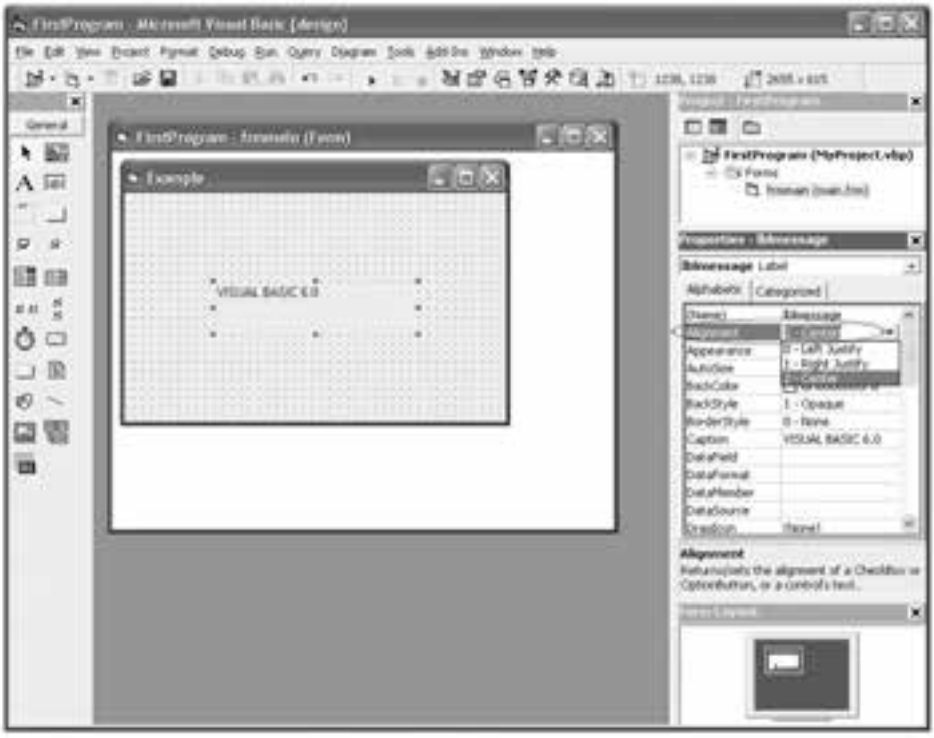
۴ – پس از انجام مرحله قبل همان‌طورکه می‌بینید عبارت به شکل مناسبی در داخل کنترل نمایش داده نمی‌شود، بنابراین برای آن‌که عبارت مربوط در وسط کنترل قرار گیرد از خصوصیت Alignment استفاده کنید. خصوصیت Alignment اجازه می‌دهد تا تراز عبارتی را که در خصوصیت Caption نگهداری می‌شود، تعیین کنید. عبارت نمایشی

می‌تواند از سمت چپ یا راست کنترل یا در وسط آن نمایش داده شود، مقادیری که این خصوصیت می‌تواند کسب کند در جدول ۴-۳ ارایه شده است.

جدول ۴-۳ مقادیری که خصوصیت Alignment کسب می‌کند.

مقدار خصوصیت	توضیح
۰-Left Justify	تراز متن از سمت چپ
۱-Right Justify	تراز متن از سمت راست
۲-Center	تراز متن از مرکز

۵ –در کادر لیست موجود در پنجره خصوصیات نام کنترل برچسب lblmessage را انتخاب کنید و خصوصیت Alignment را پیدا کرده و روی آن کلیک کنید، سپس روی دکمه روبه‌روی این خصوصیت کلیک کنید و گزینه Center را برگزینید (شکل ۱۲-۳).



شکل۱۲-۳

۶ – ارتفاع کنترل را به‌وسیله گیره‌های اطراف آن روی مقدار ۲۵۵ تنظیم کنید.

۷ – با استفاده از دکمه Start در نوار ابزار استاندارد، برنامه را اجرا کرده و نتیجه را بررسی کنید.

برای اجرای برنامه می‌توانید از این روش‌ها نیز استفاده کنید:

- فشردن کلید F5
- گزینه Start را از منوی Run در نوار منوی ویژوال‌بیسیک انتخاب کنید.

۸ – با استفاده از دکمه در پنجره برنامه اجرای برنامه را خاتمه دهید و به محیط طراحی برنامه بازگردید.

برای پایان دادن به اجرای یک برنامه می‌توانید از این روش‌ها نیز استفاده کنید:

- در نوار ابزار استاندارد روی دکمه top کلیک کنید.
- گزینه End را از منوی Run در نوار منوی ویژوال‌بیسیک انتخاب کنید.

۹–تغییرات ایجاد شده در پروژه و فرم را ذخیره کنید و از برنامه ویژوال‌بیسیک خارج شوید.

تمرین:
یک پروژه از نوع Standard EXE را به همراه یک فرم طراحی کنید تا نام مدرسه خود را با تراز راست روی فرم، نمایش دهید.

مثال ۳: پروژه MyProject را که در مثال قبل ایجاد کردید به گونه‌ای تنظیم کنید که در هنگام نمایش فرم، رنگ عبارت نمایشی قرمز شود، به این منظور عملیات زیر را به ترتیب انجام دهید:

۱ – برنامه ویژوال‌بیسیک را اجرا کنید تا کادر محاوره New Project نمایش داده شود.

۲ – در کادر محاوره New Project روی زبانه Recent کلیک کنید. با انتخاب این زبانه لیستی از پروژه‌هایی را که اخیراً طراحی کرده یا تغییر داده‌اید، مشاهده خواهید کرد (شکل ۱۳–۳).

پروژه MyProject را از کادر لیست موجود در کادرمحاوره انتخاب کرده سپس روی دکمه Open کلیک کنید.

۳ – پنجره طراحی فرم را فعال کنید، سپس در پنجره خصوصیات و از کادر لیست موجود در آن گزینه lblmessage Label را برگزینید.

۴ – برای تنظیم رنگ عبارت نمایشی در کنترل برچسب، خصوصیت ForeColor را از لیست اسامی خصوصیات کنترل انتخاب کنید. سپس روی دکمه روبه‌روی نام این خصوصیت کلیک کنید (شکل ۱۴–۳).

شکل۱۳-۳

شکل۱۴-۳

۵ – در کادر رنگی که ظاهر می‌شود دو زبانه با عناوین System و Palette مشاهده می‌کنید که به‌وسیله این دو زبانه می‌توانید رنگ‌های متفاوتی را انتخاب کنید (شکل ۱۴-۳). هم‌چنین می‌توانید برای تعیین رنگ مورد نظر یک عدد مبنای شانزده را در کادر متنی که در روبه‌روی خصوصیت ForeColor قرار دارد، تایپ کنید. البته استفاده از این روش کمی دشوارتر از سایر روش‌هاست. به این دلیل استفاده از روش‌های دیگر متداول‌تر است.

۶ – روی زبانه Palette و سپس روی رنگ قرمز کلیک کنید. همان‌طورکه مشاهده خواهید کرد در پنجره طراحی فرم عبارت VISUAL BASIC 6.0 به رنگ قرمز درمی‌آید.

تمرین:
پروژه‌ای طراحی کنید تا عبارت Microsoft Windows XP Home Edition را با رنگ زرد روی زمینه سیاه و روی یک فرم با عنوان Title نمایش تغییر یابد رنگ زمینه می‌توانید از خصوصیت BackColor استفاده کنید).

۴-۳ برنامه‌نویسی رویدادگرا

بـــه هر یک از اتفاقاتـــی که در محیط ویندوز رخ می‌دهد، یـــک رویداد یا Event گفته می‌شـــود، به عنـــوان نمونه، هر یک از اتفاقات مانند بســـتن و بازکـــردن پنجره‌ها، کلیک، دابل کلیک، کلیک راســـت و فشردن کلیدهای صفحه کلید یک رویداد به شمار می‌روند؛ ویژوال‌بیســـیک یک زبان برنامه‌نویسی رویدادگرا است که امکانات لازم را برای شناسایی و استفاده هر یک از رویدادها فراهم می‌کند.

یک رویداد، بخشـــی از برنامه اســـت که به صورت دســـتورالعمل‌های مختلف توسط برنامه‌نویـــس تهیـــه و آماده می‌شـــود تا در زمانی‌که رویـــداد مربوطه اتفـــاق می‌افتد، این دســـتورات اجرا شـــوند. رویدادهای مربوط به فرم و کنترل‌های آن در بخش ماژول فرم قرار دارند که قســـمتی از اطلاعات فرم به شـــمار می‌روند. در واقع یک فرم از دو بخش عمده تشکیل می‌شود بخش اول بخشی است که در پنجره طراحی فرم مشاهده می‌کنید و بخش دیگر ماژول فرم است و از رویدادهایی تشکیل می‌شود که می‌توانید از آن‌ها برای اجرای دستورالعمل‌های مورد نیاز خود استفاده کنید. فرم‌ها و کنترل‌ها رویدادهای مشابه و متفاوتی دارند که به مرور به توضیح آن‌ها می‌پردازیم.

شکل کلی یک رویداد به‌صورت زیر است:

```
Private Sub   Form        نام رویداد _ نام کنترل یا کلمه ( )
    دستورات
End Sub
```

هر بلاک رویداد از دو بخش ابتدایی و انتهایی تشکیل می‌شود. بخش ابتدایی هر بلاک رویداد از کلمه کلیدی Private Sub و سپس نام کنترل یا کلمه Form (برای فرم‌ها) تشکیل می‌شود و در ادامه این بخش نام رویداد انتخاب شده همراه با یک خط زیر (Underline) و دو پرانتز بازو بسته قرار می‌گیرد. بخش پایانی در تمام رویدادها را کلمات کلیدی End Sub تشکیل می‌دهند و دستوراتی که در زمان رخ دادن رویداد اجرا می‌شوند در بین بخش آغازین و پایانی قرار می‌گیرند. به‌عنوان مثال رویداد کلیک برای یک فرم به این صورت اســـت:

```
Private Sub   Form _Click ( )
    دستورات
End Sub
```

دســـتوراتی که در این رویداد قرار می‌گیرند زمانی اجرا خواهند شد که کاربر در زمان اجرای برنامه روی فرم کلیک کند.

۵-۳ کنترل دکمه فرمان (Command Button)

این کنترل زمانی استفاده می‌شود که لازم باشد با کلیک کردن روی آن (یا فشردن کلید Enter) یک یا مجموعه‌ای از دستورات اجرا شود یا عملیاتی را متوقف کرده یا خاتمه دهد. احتمالاً با این نوع از کنترل‌ها آشنا هستید و از دکمه‌هایی مانند OK، Apply و Cancel در کادرهای محاوره (مانند Display Properties) استفاده کرده‌اید. این کنترل مانند کنترل برچسب دارای خاصیت‌های Name، Caption و Fore Color می‌باشد و رویداد Click آن از پرکاربردترین رویدادهای این کنترل است.

مثال ۴: یک پروژه از نوع Standard EXE به همراه یک فرم و یک دکمه فرمان با عنوان Show ایجاد کنید که با کلیک روی دکمه فرمان نام و نام‌خانوادگی شما نمایش داده شود. به این منظور عملیات زیر را به ترتیب انجام دهید:

۱ – برنامه ویژوال‌بیسیک را اجرا کنید و یک پروژه از نوع Standard EXE شامل یک فرم و یک کنترل دکمه فرمان و دو کنترل برچسب ایجاد کنید.

۲ – اکنون باید رویداد مناسبی را برای این منظور انتخاب کنید، بنابراین در پنجره طراحی فرم روی کنترل دکمه فرمان دابل کلیک کنید تا بخش ماژول فرم فعال شود.

البته می‌توانید با استفاده یکی از این روش‌ها پنجره ماژول فرم را فعال کنید:

الف– برای فعال کردن ماژول فرم می‌توانید گزینه Code را از منوی View انتخاب کنید.

ب– برای فعال کردن ماژول فرم می‌توانید روی دکمه View Code در پنجره پروژه کلیک کنید.

در قسمت بالای این پنجره دو کادر لیست را ملاحظه می‌کنید. در کادر لیست سمت چپ نام کنترل‌ها و فرم و در کادر لیست سمت راست نام رویدادها را مشاهده خواهید کرد. برای آماده‌سازی رویداد موردنظر نام کنترل یا کلمه Form (برای فرم) را از کادر لیست سمت چپ و نام رویداد مورد نظر را از کادر لیست سمت راست انتخاب کنید. در این صورت بلاک‌های کد مانند شکل ۱۵-۳ نمایش داده می‌شود.

۳ – در این مرحله رویداد Click دکمه فرمان در پنجره کد نمایش داده شده است. این رویداد زمانی اجرا می‌شود که عمل کلیک روی دکمه فرمان انجام شود.

۴ – اکنون در رویداد Click دکمه فرمان، دستوری بنویسید که خصوصیت Caption کنترل

برچسب را در هنگام اجرای برنامه تنظیم کند؛ بنابراین دستور بعد را در رویداد مزبور بنویسید (شکل ۱۵–۳).

```
lblmessage.Caption=«Kaveh Behrozi»=vbRed
```

شکل ۱۵–۳

همان‌طورکه ملاحظه می‌کنید برای تغییر خصوصیات فرم‌ها و کنترل‌ها با استفاده از کدنویسی می‌توانید به این شکل عمل کنید:

مقدار خصوصیت = نام خصوصیت . نام

نکته در هنگام تایپ کاراکتر نقطه پس از نام کنترل، ویژوال‌بیسیک به‌طور خودکار لیستی از خصوصیات کنترل را در اختیار شما قرار می‌دهد که می‌توانید خصوصیت موردنظر را انتخاب کرده و با فشردن کلید Spacebar آن را به پنجره کد کپی کنید (شکل ۱۵–۳).

۵ – پروژه را اجرا کرده و روی دکمه فرمان Show کلیک کنید.

۶ – اجرای پروژه را متوقف کنید و به پنجره طراحی فرم بازگردید. سپس پروژه و فرم را با نام Show name ذخیره کنید.

مثـــال ۵: می‌خواهیـــم پروژه Showname را به شـــکلی تنظیم کنیم تـــا بتواند نام و نام‌خانوادگی هر شـــخص دلخواهی را نمایش دهد. بـــرای این کار عملیات زیر را به ترتیب انجام دهید:

۱ – برنامه ویژوال‌بیسیک را اجرا کرده و پروژه Showname را باز کنید.

۲ – یک کنترل کادر متن روی فرم قرار دهید.

۳ – به رویداد Click دکمه فرمان بروید و دستور آن را به این صورت تغییر دهید.

```
lblmessage.Caption=txtname.Text
```

۴ - تغییرات را ذخیره کرده و پروژه را اجرا کنید.

۵ - نـــام دلخواهـــی را در کادر متن تایپ کرده و روی دکمـــه فرمان Show کلیک کنید.

۶ - اجرای پروژه را متوقف کرده و به پنجره ویژوال‌بیسیک بازگردید.

Learn in English

Creating a Project

You begin creating this application by choosing New Project from the File menu, then selecting Standard EXE in the New Project dialog box (when you first start Visual Basic, the New Project dialog box is presented). Visual Basic creates a new project and displays a new form. To draw the interface, you use a label control, a textbox control or some of other controls.

واژه‌نامه

Alphabetic	الفبایی
Caption	عنوان
Categorized	موضوعی
Create	تشکیل دادن، ایجاد کردن
Development	توسعه
Dialog Box	کادرمحاوره
Enterprise Edition	نگارش نهایی
Error Handling	رسیدگی به خطا
Event	رویداد
Event Handling	رسیدگی به اتفاق (رویدادگر)
High Level Languages	زبان‌های سطح بالا
Environment Integrated Development	محیط برنامه‌نویسی مجتمع
Label Control	کنترل برچسب
Learning Edition	نگارش آموزشی
Low Level Languages	زبان‌های سطح پایین
Module	ماژول (به عبارت دیگر قطعه‌ای که بخشی از یک پروژه بوده و فقط حاوی دستورالعمل‌هاست.)
Object Oriented Feature	ویژگی شئ‌گرا
Object Oriented Programming	برنامه‌نویسی شئ‌گرا
Present	ارائه کردن، معرفی نمودن
Private	محلی
Project	پروژه
Property	خصوصیت
Provide	ارائه‌کردن
Rapid	سریع، تند
Rapid Application Development	طراحی سریع برنامه
Separate	جدا کردن، جداگانه، مجزا
Standard EXE	برنامه اجرایی استاندارد
TextBox Control	کنترل کادر متن

خلاصه مطالب

- به مجموعه‌ای از قواعد و دستورالعمل‌های تعریف شده زبان برنامه‌نویسی می‌گویند.
- زبان برنامه‌نویسی ویژوال‌بیسیک از زبان‌های برنامه‌نویسی سطح بالاست که از ویژگی‌های زیر برخوردار می‌باشد:
  - ❋رابط گرافیکی کاربر (GUI) و محیط توسعه یافته مجتمع (IDE)
  - ❋امکان طراحی سریع برنامه
  - ❋دسترسی به برنامه‌های کاربردی ویندوز با استفاده از توابع API
  - ❋توانایی مدیریت رویدادها و خطاها
- در مرورگر پروژه می‌توانید اجزای تشکیل دهنده برنامه مانند پروژه‌ها، فرم‌ها و ماژول‌ها را مشاهده کنید.
- در پنجره خصوصیات می‌توانید خصوصیات فرم‌ها و کنترل‌های موجود در آن‌ها را مشاهده و تنظیم کنید.
- جعبه ابزار ویژوال‌بیسیک حاوی کنترل‌های متعددی است که از آن‌ها در طراحی برنامه استفاده می‌شود.
- از کنترل برچسب (Label) برای نمایش انواع خروجی‌های متنی مانند پیام‌ها یا مقدار خصوصیات اشیا استفاده می‌شود.
- به اتفاقاتی مانند کلیک، دابل کلیک، کلیک راست، بسته شدن یک پنجره، فشرده شدن یک کلید و غیره که در محیط ویندوز و برنامه‌ها رخ می‌دهد، یک رویداد گفته می‌شود.
- هر رویه رویداد یک مجموعه از دستورات است که به‌صورت جداگانه در بخش کد فرم نگهداری می‌شود و زمانی‌که رویداد رخ می‌دهد، این دستورات انجام می‌شوند.
- هر رویه رویداد با کلمات کلیدی Private Sub به همراه نام فرم یا کنترل آغاز شده و با End Sub خاتمه می‌یابد.
- مقدار خصوصیات کنترل‌ها را علاوه بر تنظیم در پنجره خصوصیات می‌توانید با استفاده از کدنویسی به شکل زیر تنظیم کنید:

مقدار خصوصیت = نام خصوصیت . نام

- از کنترل کادر متن (TextBox) برای دریافت داده‌ها از کاربر و ورود آن‌ها به برنامه استفاده می‌شود.

آزمون نظری

۱- کدام گزینه از اهداف طراحی زبان بیسیک بوده است؟

الف- برنامه‌نویسی سیستمی ب- برنامه‌نویسی آموزشی

ج- برنامه‌نویسی تجاری د- برنامه‌نویسی علمی

۲- کدام گزینه زیر یکی از انواع زبان‌های سطح پایین است؟

الف- فرترن ب- اسمبلی ج- C++ د- پاسکال

۳ - کدام ویژگی در ویژوال‌بیسیک برنامه‌نویسی رویدادگر را میسر می‌کند؟

الف- API ب- EVENT HANDLING

ج- ERROR HANDLING د- OOP

۴ - کدام‌یک از زبان‌های برنامه‌نویسی برای برنامه‌نویسی علمی و تجاری مناسب است؟

الف- پاسکال ب- فرترن ج- زبان ماشین د- اسمبلی

۵- ویژگی RAD (Rapid Application Development) عبارت است از:

الف- کشف و تصحیح آسان و سریع اشتباهات

ب- طراحی سریع و آسان برنامه‌ها

ج- طراحی هم‌زمان چند برنامه

د- برنامه نویسی به روش OOP

۶ - در کدام نگارش ویژوال‌بیسیک ۶ امکانات ویژه‌ای به منظور برنامه‌نویسی برای محیط شبکه‌های محلی و اینترنت وجود دارد؟

الف- نگارش آموزشی ب- نگارش اولیه

ج- نگارش نهایی د- نگارش شبکه

۷ - کدام نگارش از زبان Basic برای برنامه‌نویسی در محیط اینترنت ارائه شده است؟

الف- GWBasic ب- QBasic

ج- Visual Basic د- VB Script

۸ - کدام نوع از انواع زبان‌های Basic برای برنامه‌نویسی در برنامه‌های کاربردی ارائه شده است؟

الف- QBasic ب- VB ج- VBA د- Activex

۹ - کدام‌یک از گزینه‌های زیر برای برنامه‌نویسی سیستمی مناسب هستند؟

الف- فرترن ب- ماشین ج- C و C++ د- بیسیک

۱۰ – برای باز کردن پروژه‌هایی که اخیراً مورد استفاده قرار گرفته، انتخاب کدام زبانه در کادر محاوره New Project مناسب است؟

الف– New ب– Recent ج– Existing د– Open

۱۱ – کدام نام برای خصوصیت Name یک فرم مناسب است؟

الف– frm-show ب– frm main ج– ۱form د– frmtest

۱۲ – پسوند فایل‌های پروژه و فرم به ترتیب عبارتند از (راست به چپ):

الف– frm. ، vbp. ب– vbp. ، frm.

ج– vbf. ، vbp. د– frm. ، .vbf

۱۳ – کدام کنترل برای دریافت داده‌ها از کاربر مناسب است؟

الف– برچسب ب– کادر متن ج– کادر علامت د– فرم

۱۴ – با استفاده از کدام خصوصیت عنوان یک فرم تعیین می‌شود؟

الف– Caption ب– Visible ج– Name د– Enabled

۱۵ –از کدام بخش در پنجره ویژوال‌بیسیک برای ایجاد کنترل‌ها روی فرم استفاده می‌شود؟

الف– نوار ابزار ب– جعبه ابزار ج– نوار منو د– پنجره خصوصیات

16 - The VBA language can be used for programming in .....

a- Excel b- Access c- Applications d- all of above

17 - Which of the following options can be used for creating a project in new project dialog box?

a- Recent b- New c- Existing d- Open

۱۸ – زبان برنامه‌نویسی را تعریف کنید سپس انواع آن‌ها را با توجه به نزدیکی آن به زبان ماشین بیان کنید.

۱۹ – ویژگی‌های زبان برنامه‌نویسی ویژوال‌بیسیک را شرح دهید.

۲۰ – اجزای اصلی در یک برنامه کاربردی را نام ببرید.

۲۱ – کاربرد پنجره پروژه، خصوصیات، ابزار و تعیین موقعیت فرم را توضیح دهید.

۲۲ – مفهوم رویداد را توضیح دهید و کاربرد رویه رویداد را بیان کنید.

۲۳ – نحوه تنظیم خصوصیت در کنترل‌ها را در زمان اجرای برنامه توضیح دهید.

۲۴–خصوصیت‌هایی که به‌وسیله آن‌ها می‌توان موقعیت یک کنترل و فرم را در زمان نمایش تعیین کرد، توضیح دهید.

آزمون عملی

۱ – یک برنامه از نوع استاندارد اجرایی ایجاد کنید که حاوی یک فرم، دو کنترل برچسب و دو کنترل کادر متن بوده و بتواند نام و نام‌خانوادگی کاربر را دریافت کند و نام وی را با رنگ قرمز روی زمینه سبز و نام خانوادگی را با رنگ آبی روی زمینه سفید روی کنترل‌های برچسب نمایش دهد. در ضمن درصورت انجام عمل کلیک روی کنترل‌های برچسب رنگ زمینه و قلم جابه‌جا شود.

۲ – برنامه آزمون عملی شماره یک را به گونه‌ای تغییر دهید تا با انجام عمل کلیک روی هر کادرمتن محتویات آن پاک شود.

واحد کارچهارم

هدف جزیی

توانایی تعریف انواع متغیرها، ثابت‌ها و استفاده از عملگرهای ریاضی و رشته‌ای

هدف‌های رفتاری

پس از مطالعه این واحد کار از فراگیر انتظار می‌رود که:

۱- انواع متغیرها و ثابت‌ها را بیان کند و نحوه تعریف و استفاده از آن‌ها را توضیح دهد.

۲- عملگرهای ریاضی و رشـــته‌ای را بیان کرده، نحوه استفاده و اولویت اجرای آن‌ها را نسبت به یکدیگر شرح دهد.

۳- متغیرهای عمومی و محلی را توضیح دهد و نحوه تعریف و کار با آن‌ها را بیان کند.

کلیات

در طراحی پروژه‌های واقعی لازم است تا داده‌ها و اطلاعات جمع‌آوری شده به کامپیوتر و نرم‌افزار داده شود و پس از انجام پردازش‌هایی روی آن‌ها، نتیجه یا نتایج حاصل به کاربر ارایه شود. بنابراین در تمام مراحل ورود، محاسبه و ارایه نتایج نیاز به ذخیره‌سازی داده و اطلاعات در حافظه کامپیوتر احساس می‌شود که در این رابطه می‌توانید از حافظه اصلی و حافظه جانبی استفاده کنید. در این واحد کار نحوه استفاده از حافظه اصلی یا RAM را برای ذخیره‌سازی داده‌ها و اطلاعات فرا می‌گیرید، به علاوه برای انجام عملیات ریاضی، نحوه استفاده از انواع عملگرها را در ویژوال‌بیسیک می‌آموزید.

۱–۴ نحوه تعریف و استفاده از انواع متغیرها در ویژوال‌بیسیک

یکی از عملیاتی که در برنامه‌ها انجام می‌گیرد، ذخیره‌سازی داده و اطلاعات در حافظه کامپیوتر است. ابتدا نحوه معرفی انواع متغیرها را در ویژوال‌بیسیک توضیح می‌دهیم، سپس نحوه انجام این کار را همراه با یک مثال آموزش می‌دهیم.

۱–۱–۴ نحوه معرفی متغیرها در ویژوال‌بیسیک

متغیر، مکانی از حافظه است که برای نگهداری داده‌ها و اطلاعات حاصل از پردازش داده‌ها استفاده می‌شود, به عبارت دیگر متغیر ظرف نگهداری داده‌ها و اطلاعات است و نوع آن با توجه به داده‌ای که نگهداری می‌کنند، تعیین می‌شود. به علاوه هر متغیر یک نام ویژه دارد که به وسیله آن می‌توان داده‌ها را در حافظه ذخیره کرد یا مقدار آن را مورد دستیابی قرار داد. برای تعریف متغیرها می‌توانید از دستور Dim به این صورت استفاده کنید:

```
Dim نام متغیر    As    نوع داده
```

نام متغیر دلخواه بوده و می‌توانید از کاراکترهای حرفی، رقمی یا ترکیبی از آن‌ها استفاده کنید، اما بهتر است نام متغیر متناسب با مقداری که در آن ذخیره می‌شود، تعیین شده و از اسامی طولانی و نامأنوس استفاده نشود. برای نام‌گذاری یک متغیر این موارد را رعایت کنید:

الف– نام متغیر با یک کاراکتر حرفی آغاز شود و حداکثر ۲۵۵ کاراکتر باشد.

ب–استفاده از فاصله خالی, نقطه, کاما و نظایر آن‌ها مجاز نیست، اما استفاده از کاراکتر زیرخط (_) امکان‌پذیر است.

ج- در نام‌گذاری متغیرها از علایم و اسامی رزرو شده در ویژوال‌بیسیک استفاده نکنید. علایم و اسامی رزرو شده در ویژوال‌بیسیک علایم و عباراتی هستند که از آن‌ها برای انجام اعمال و دستورالعمل‌های خاصی استفاده می‌شود مانند علایم + و ـ و عباراتی مانند Control، Form و غیره.

د- ویژوال‌بیسیک تفاوتی بین کاراکترهای حرفی کوچک و بزرگ قائل نمی‌شود.

نوع داده یک متغیر، تعیین‌کننده نوع مقداری است که در آن ذخیره می‌شود و میزان حافظه‌ای را که برای آن اختصاص داده می‌شود و نحوه ذخیره‌شدن آن‌را در حافظه معین می‌کند. میزان حافظه اشغالی برای ذخیره‌سازی یک اسم با میزان حافظه اشغالی توسط یک عدد صحیح یا اعشاری متفاوت است.

به عنوان مثال یک متغیر از نوع عدد صحیح (Integer) ۲ بایت فضا اشغال می‌کند، در صورتی که یک متغیر از نوع عدد اعشاری (Single) ۴ بایت فضا نیاز دارد. بنابراین در زمان تعریف یک متغیر لازم است تا نوع داده‌ای که در آن ذخیره خواهد شد، معین شود تا ویژوال‌بیسیک با توجه به آن، مقدار حافظه لازم را در اختیار متغیر قرار دهد.

انواع داده‌هایی که در ویژوال‌بیسیک ۶ قابل استفاده هستند به همراه میزان فضای مورد نیاز و محدوده مقادیر در جدول ۱-۴ ارایه شده‌اند.

جدول ۱-۴ انواع داده‌ها در ویژوال‌بیسیک

نوع داده	نوع مقدار	میزان حافظه اشغالی	محدوده مقادیر	پیشوند مناسب
Byte	اعداد صحیح	۱ بایت	صفر تا ۲۵۵	byt
Integer	اعداد صحیح	۲ بایت	-۳۲۷۶۸ تا +۳۲۷۶۷	int
Long	اعداد صحیح	۴ بایت	$\pm ۲\times ۱۰^{۹}$	lng
Single	اعداد اعشاری	۴ بایت	$\pm ۱\times ۱۰^{-۴۵}$ تا $\pm ۳\times ۱۰^{۳۸}$	sng
Double	اعداد اعشاری	۸ بایت	$\pm ۵\times ۱^{-۳۲۴}$ تا $\pm ۱/۸\times ۱۰^{۳۰۸}$	dbl
Currency	اعداد اعشاری	۸ بایت	$\pm ۹\times ۱۰^{۱۴}$	cur
(با طول ثابت) String	متن (رشته)	یک بایت برای هر کاراکتر	کاراکتر ۶۵۴۰۰	str
(با طول متغیر) String	متن (رشته)	یک بایت برای هر کاراکتر	۲×۱۰^{۹} کاراکتر	str
Boolean	منطقی	۲ بایت	True یا False	bln
Date	تاریخ و ساعت	۸ بایت	۱/۱/۱۰۰ الی ۱۲/۳۱/۹۹۹۹	dem
Variant	هر یک از انواع فوق	با توجه به نوع داده متغیراست.	ـــــــــ	vnt

ویژوال‌بیسیک از انواع داده‌های مختلفی پشتیبانی می‌کند که می‌توانند نیازهای مختلف شما را در زمان برنامه‌نویسی پوشش دهند و با استفاده از انواع داده‌های ارایه شده در جدول ۱–۵ انواع متفاوتی از متغیرهای مورد نیاز را ایجاد کنند. در این‌جا به توضیح انواع آن‌ها می‌پردازیم.

متغیرهای عددی

ویژوال‌بیسیک از چند نوع داده عددی مختلف پشتیبانی می‌کند؛ داده صحیح، اعشاری و پولی. داده صحیح شامل انواع Integer، Long و Byte می‌باشد، با انواع Integer و Long می‌توانید متغیرهایی ایجاد کنید که در محدوده اعداد صحیح باشند، مقادیری مانند ۲۲۰، ۲۰۰۵, ۷۸– و غیره. اما نوع Byte در مواردی به کار می‌رود که فقط از اعداد صحیح مثبت در محدوده صفر تا ۲۵۵ استفاده می‌کنیم. انواع داده اعشاری نیز شامل انواع Single،Double و Currency هستند و توانایی ایجاد متغیرهایی را دارند که می‌توانند هرگونه عدد اعم از صحیح, اعشاری، مثبت و منفی مانند ۱۴۲– ، ۱۷/۷۵– یا ۰۰۲۵ / ۰ ۱ ۴۸۹۰را در خود جای دهند. در صورتی که نیاز به نگهداری داده‌هایی از نوع پولی داشته باشید می‌توانید از نوع داده Currency استفاده کنید. به عنوان مثال به این تعاریف توجه کنید:

```
Dim     intcounter     As     Integer
Dim     dbltotal       As     Double
Dim     sngdistance    As     Single,intgrade    As    Integer
Dim     bytage         As     Byte,cursalary     As    Currency
```

همان‌طورکه مشاهده می‌کنید متغیر intcounter از نوع اعداد صحیح و dbltotal از نوع اعشاری تعریف شده‌اند. به‌علاوه‌متغیرهای sngdistance و intgrade به‌طور هم‌زمان با استفاده از یک دستور Dim تعریف شده‌اند. همین‌طور متغیرهای bytage و cursalary نیز به ترتیب از نوع Byte و پولی معرفی شده‌اند.

نکته استفاده از پیشوندهای مناسب با نوع داده در نام متغیر, شناسایی آسان متغیرها و در نتیجه خوانایی کدها را درپی خواهد داشت.

متغیرهای منطقی

این نوع متغیرها توانایی دریافت مقادیر درست (True) یا نادرست (False)، بله (Yes) یا خیر (No) را دارا می‌باشند، از این نوع متغیرها معمولاً زمانی استفاده می‌شود که داده‌های دو حالتی مانند زن یا مرد، مردود یا قبول، بیمار یا سالم و نظایر آن‌ها مورد نظر باشند به عنوان مثال برای ذخیره‌سازی وضعیت قبولی یک دانش‌آموز در مدرسه می‌توان متغیر blnresult را به این شکل تعریف کرد:

```
Dim blnresult As Boolean
```

متغیرهای تاریخ و زمان (Date & Time)

با استفاده از این نوع متغیرها می‌توانید مقادیر ساعت و تاریخ یا ترکیبی از این دو را ذخیره کنید. به عنوان مثال برای ذخیره‌سازی تاریخ تولد افراد می‌توانید متغیر dtmb_Date را به این صورت تعریف کنید:

```
Dim dtmb_Date As Date
```

متغیرهای رشته‌ای

اگر بخواهید مقادیر غیرعددی مانند متون و کاراکترها را ذخیره کنید، می‌توانید متغیرهای خود را از نوع داده رشته‌ای تعریف کنید. به عنوان مثال این نوع داده را می‌توان برای ذخیره‌سازی نام و نام خانوادگی اشخاص، نام شهر، آدرس یک محل و غیره به کار برد.

متغیر رشته‌ای با طول ثابت، توانایی ذخیره‌سازی داده‌های رشته‌ای را با توجه به تعداد کاراکترهای تعیین شده دارد و برای تعریف آن از این الگو استفاده می‌شود:

```
Dim نام متغیر رشته‌ای As String*n
```

n عدد صحیح و مثبتی است که تعداد کاراکترها را در متغیر رشته‌ای تعیین می‌کند. به عنوان مثال برای ذخیره‌سازی نام افراد با حداکثر طول نام ۲۵ کاراکتر، از این دستور استفاده می‌شود:

```
Dim strname As String*25
```

متغیر رشته‌ای با طول متغیر، توانایی نگهداری یک تا ۲ میلیارد کاراکتر را دارد و در هنگام ذخیره‌سازی اطلاعات، فضای لازم را با توجه به طول رشته در اختیار متغیر قرار می‌دهد. برای تعریف این‌گونه متغیرها از این الگو استفاده می‌شود:

```
Dim نام متغیر رشته‌ای As String
```

به عنوان مثال برای ذخیره‌سازی اسامی افراد که حداکثر طول نام آن‌ها مشخص نیست، از این دستور استفاده می‌شود:

```
Dim strname As String
```

متغیرهای ترکیبی (Variant)

اگر به متغیری احتیاج دارید که نوع داده‌هایی که در آن ذخیره می‌شوند قابل پیش‌بینی نبوده و انواع متفاوتی را دربرمی‌گیرد، از این نوع متغیر استفاده کنید. این نوع متغیر تمام انواع داده‌های قبلی را دربرمی‌گیرد و می‌تواند مقادیر عددی, رشته‌ای, منطقی و تاریخی را بدون آن‌که نیازی به تبدیل آن‌ها باشد در خود جای دهد.

نکته در صورتی که نوع یک متغیر تعیین نشود، نوع متغیر به طور خودکار از نوع Variant در نظر گرفته خواهد شد.

تمرین:

کدام یک از این تعاریف اشتباه است:

```
Dim 1Age As Integer
Dim snggrade 1 As Single,n-no As Double
Dim sngaverage As Single,intno As Integer
```

علاوه بر روش‌هایی که تاکنون برای تعریف انواع متغیرها گفته شد، می‌توانید با استفاده از بعضی کاراکترها نوع متغیر را به طور ضمنی تعریف کنید. به عبارت دیگر با اضافه کردن کاراکترهای ارایه شده در جدول ۲-۴ به انتهای نام متغیر، نوع آن را معین کنید. از این روش فقط می‌توانید برای داده‌های عددی و رشته‌ای استفاده کنید.

جدول ۲-۴

نوع داده	کاراکتر
Integer	%
Long	&
Single	!
Double	#
Currency	@
String	$

به عنوان مثال به موارد ارایه شده در جدول ۳-۴ توجه کنید:

جدول ۳-۴

تعریف متغیرها با دستور Dim	تعریف متغیرها به صورت ضمنی
Dim intage As Integer	Intage%
Dim lngspeed As Long	Lngspeed&
Dim sngaverage As Single	sngaverage!
Dim dblgrade As Double	dblgrade#
Dim cursalary As Currency	cursalary@
Dim strname As string	strname$

۲-۱-۴ نحوه مقداردهی به متغیرها

پس از تعریف یک متغیر بهتر است مقداری را در آن ذخیره کنیم. به این کار انتساب مقادیر یا مقداردهی به متغیر گفته می‌شود. برای انجام این کار می‌توانید از این الگو استفاده کنید:

Let نام متغیر = expression

البته ذکر کلمه کلیدی Let اختیاری است و معمولاً از آن استفاده نمی‌شود. Expression عبارتی است که می‌تواند یکی از انواع داده‌های گفته شده، فرمول و عبارات محاسباتی، خصوصیت یک شئ مانند فرم یا کنترل و غیره باشد. مشاهده می‌کنید که دستور انتساب شبیه به مقداردهی به خصوصیات اشیا مانند فرم‌ها و کنترل است. به عنوان مثال به مواردی که در جدول ۴-۴ ارایه شده است، توجه کنید.

جدول ۴-۴

نوع عبارت	دستور انتساب
رشته‌ای	Strname="Ali"
عددی	int count=-4
عبارت ریاضی	Sngave=snglength/3
تاریخ	dtmb_Date=#Dec-4-90#
منطقی	blntest=True
عبارت ریاضی	dblsum=inta+intb+intc
ساعت	dtmtime= # 18:05:00 PM#
تاریخ و ساعت	dtmzone= #8/10/2004 5:12:17 AM#

نکته در مقداردهی به متغیرهای منطقی می‌توانید از مقادیر یک و صفر به جای True و False استفاده کنید.

متغیرهای منطقی در هنگام تعریف به طور خودکار مقدار False، متغیرهای عددی مقدار صفر و متغیرهای رشته‌ای به صورت یک رشته خالی ("") مقداردهی اولیه می‌شوند.

مقداردهی متغیرهای تاریخ و ساعت در هنگام تعریف با توجه به تنظیمات تاریخ و ساعت که به‌وسیله برنامه Regional and Language Options در بخش Control Panel ویندوز انجام می‌شود به دو روش صورت می‌گیرد. اگر ساعت روی حالت ۲۴ ساعتی تنظیم شود مقدار اولیه به‌صورت ۰۰: ۰۰: ۰۰ یا ۰۰: ۰۰: ۰ خواهد بود اما وقتی ساعت روی حالت۱۲ساعتی تنظیم شود مقدار اولیه به صورت ۱۲:۰۰:۰۰ AM می‌باشد.

مثال ۱: یک پروژه مطابق شکل ۱-۴ طراحی کنید که اطلاعات مربوط به یک کارمند شامل نام، نام‌خانوادگی و میزان حقوق وی را دریافت کرده, سپس با کلیک روی دکمه Calculate میزان حقوق خالص وی را پس از کسر مالیات و بیمه بر اساس فرمول‌های زیر محاسبه کرده و نمایش دهد؛ هم‌چنین با کلیک روی دکمه New، برنامه برای دریافت اطلاعات کارمند جدید آماده شود.

۳ درصد حقوق= بیمه

۵ درصد حقوق= مالیات

شکل ۱-۴

۱- برنامه ویژوال‌بیسیک را اجرا کنید و یک پروژه از نوع Standard EXE شامل یک فرم و کنترل‌های کادرمتن و برچسب را مطابق شکل ۱-۴ و جدول‌های ۵-۴، ۶-۴ و ۷-۴ طراحی کنید.

جدول ۴-۵ خصوصیات فرم

خصوصیت	مقدار
Name	frmsal
Caption	Salary

جدول ۴-۶ خصوصیات کنترل‌ها

کنترل / خصوصیت	Label	Label	Label	Label	Label
Name	lblname	lblfam	lblsal	lblp	lblpay
Caption	Name:	Family:	Salary:	Payment:	◦

جدول ۴-۷ خصوصیات کنترل‌ها

کنترل / خصوصیت	TextBox	TextBox	TextBox	Command Button	Command Button	Command Button
Name	txtname	txtfam	txtsal	cmdcalculate	cmdnew	cmdexit
Caption	--	--	--	&Calculate	&New	&Exit

۲- در ایـــن مرحله لازم اســـت کنترل‌های دکمه فرمـــان را روی فرم قرار دهید. به این منظـــور در جعبه ابـــزار روی آیکن (Command Button) دابل‌کلیک کنید پس از ایجاد دکمه‌های فرمان خصوصیت‌های آن‌ها را مطابق جدول ۴-۷ تنظیم نمایید.

۳- برای تعیین کلید ایجاد دسترسی سریع (Hot Key) در کنترل‌ها می‌توان در خصوصیت Caption و قبـــل از حرف موردنظر از کاراکتر & اســـتفاده کرد بنابراین لازم اســـت قبل از حروف C، N و E در ســـه دکمه فرمان ایجاد شـــده کاراکتر & را تایپ کنید. پس از تعیین کلید دسترسی سریع، بلافاصله در پنجره طراحی فرم زیر حروف مربوط یک خط مشاهده می‌شود.

۴- بـــه پنجره ماژول فرم برویـــد و رویداد Click دکمه فرمان Calculate، New و Exit را به این صورت تنظیم کنید:

```
Private Sub cmdcalculate_Click()
```

```
Dim sngsal As Single
Dim sngins As Single
Dim sngtax As Single
Dim sngpay As Single
sngsal = Val(txtsal.Text)
sngins=sngsal *0.03
sngtax=sngsal * 0.05
sngpay= sngsal-sngins-sngtax
1b1pay.Caption=sngpay
End Sub
```

در این رویداد ابتدا با استفاده از دستور Dim متغیرهای sngsal برای حقوق، sngins برای بیمه، sngtax برای مالیات و sngpay برای حقوق خالص از نوع اعشاری تعریف شده‌اند سپس مقدار حقوق که در خصوصیت Text کادر متن txtsal ذخیره شده است با استفاده از تابع Val به عدد تبدیل و در متغیر اعشاری sngsal ذخیره می‌شود تا بعداً برای محاسبه بیمه، مالیات و حقوق پرداختی از آن استفاده شود. تابع Val می‌تواند یک عبارت رشته‌ای را که شامل کاراکترهای رقمی باشد دریافت کرده و به نوع داده عددی تبدیل کند. در مرحله آخر نیز حقوق خالص وی در خصوصیت Caption کنترل برچسب lblpay ذخیره می‌شود تا روی فرم نمایش داده شود.

۵ – رویداد Click دکمه فرمان New را به این صورت تنظیم کنید:

```
Private Sub cmdnew_Click()
  txtname.Text=""
  txtfam.Text=""
  txtsal.Text=""
  1b1pay.Caption = 0
  txtname.setFocus
End Sub
```

در این رویداد برای آن‌که بتوان داده‌های قبلی را در کادرهای متن پاک کرده و امکان ورود داده‌های جدید به وجود آید ابتدا محتویات کادرهای متن پاک شده و سپس مقدار

صفر در کنترل برچسب نمایش داده می‌شود سپس با استفاده از متد SetFocus، فوکوس به اولین کادر متن منتقل می‌شـــود تا وقتی کاربر روی دکمه New کلیک کند بلافاصله بتواند نام کارمند را وارد نماید.

متد، یک یا مجموعه عملیاتی اســـت که روی یک فرم یا کنترل انجام می‌شـــود و روی آن تأثیر می‌گذارد. برای استفاده از متدها می‌توانید از شکل کلی زیر استفاده کنید:

نام متد . نام کنترل

نکته: بـــرای دسترســـی به متدهای یک فرم در مـــاژول فرم مربوط به آن می‌توان از نام متد به تنهایی یا همراه با عبارت .me استفاده کرد.

فوکوس امکانی است که با استفاده از ماوس یا صفحه کلید به کاربر اجازه می‌دهد به فرم یا کنترل موردنظر خود دسترسی پیدا کند. در ویندوز چندین برنامه می‌توانند به طور هم‌زمان اجرا شـــوند، اما کاربر فقط می‌تواند با یک برنامه کار کند یعنی برنامه‌ای که نوار عنوان پنجره آن فعال اســـت یا به عبارت دیگر فوکوس را در اختیار دارد. در یک پنجره و فـــرم با چند کنترل نیز کاربـــر می‌تواند در هر لحظه با یک کنترل کار کند در این حالت کنترلی که فوکوس را در اختیار دارد، قابل اســـتفاده است. برای انتقال فوکوس بین فرم‌ها می‌توانیـــد از کلیـــک روی فرم مورد نظر یا کلید ترکیبی Alt+Tab و برای انتقال فوکوس از یک کنترل به کنترل دیگر از عمل کلیک یا کلید Tab استفاده کنید.

متد SetFocus یکی از متدهای مشترک بین کنترل‌ها و فرم‌هاست و اجرای آن فوکوس را به فرم یا کنترل مربوطه منتقل می‌کند.

رویداد Click دکمه فرمان Exit را به این صورت تنظیم کنید:

```
Private Sub cmdexit_Click()
  Unload Me
End Sub
```

در این رویداد با اســـتفاده از دستور Unload Me فرم برنامه از حافظه خارج می‌شود در نتیجه برنامه خاتمه می‌یابد.

۶ – پروژه و فرم را با نام salary ذخیره کنید، ســـپس پروژه را اجرا کرده و اطلاعات

دلخواهی را در کادرهای متن فرم تایپ کنید.

۷ – روی دکمه Calculate کلیک کنید تا نتیجه محاسبات یا به عبارت دیگر میزان حقوق خالص کارمند مربوطه نمایش داده شود.

۸ – روی دکمه New کلیک کنید و این بار مشخصات فرد دیگری را تایپ کرده و میزان حقوق وی را محاسبه نمایید.

۹ – با کلیک روی دکمه Exit به محیط طراحی بازگردید.

تمرین:

پروژه salary را به شکلی تغییر دهید تا هزینه بیمه، مالیات و حقوق دریافتی هر کارمند به صورت جداگانه، روی فرم قابل مشاهده باشد.

۲–۴ نحوه انجام عملیات ریاضی در ویژوال‌بیسیک

در تمام زبان‌های برنامه‌نویسی برای تبدیل انواع فرمول‌ها و انجام محاسبات، امکاناتی در نظر گرفته می‌شود. در زبان برنامه‌نویسی ویژوال‌بیسیک نیز برای این کار، از عملگرهای ریاضی برای انجام اعمالی چون جمع، تفریق، به توان‌رسانی و غیره استفاده می‌شود. با استفاده از این عملگرها انجام بخش عمده‌ای از عملیات محاسباتی امکان‌پذیر می‌شود.

مثال ۲: یک ماشین‌حساب ساده مطابق شکل ۲–۴ و جداول ۸–۴ الی ۱۰–۴ طراحی کنید که توانایی انجام عملیات ریاضی ساده را داشته باشد. به این منظور مراحل بعد را به ترتیب انجام دهید:

جدول ۸-۴ خصوصیات فرم

خصوصیت	مقدار
Name	frmcalc
Height	۳۷۲۰
Left	۳۰۰
Top	۳۰۰
Width	۴۹۰۵
Caption	Calculator

شکل ۲-۴

جدول ۹-۴ خصوصیات کنترل‌ها

کنترل / خصوصیت	Label	Label	Label	Label	TextBox
Name	lblnum۱	lblnum۲	lblr	lblresult	txtnum۱
Caption	Number ۱ :	Number ۲ :	Result :	۰	--

جدول ۱۰-۴ خصوصیات کنترل‌ها

کنترل / خصوصیت	TextBox	Command Button	Command Button	Command Button	Command Button
Name	txtnum۲	cmdadd	cmddiv	cmdminus	Cmdmulti
Caption	--	+	/	-	×

۱ – برنامه ویژوال‌بیسیک را اجرا کرده و کنترل‌های لازم را مطابق شکل ۵-۴ روی فرم ایجاد کنید.

۲ – رویداد Click دکمه جمع را به این صورت تنظیم کنید:

```
Private Sub cmdadd_Click()
    lblresult.Caption=Val(txtnuml.Text) + Val(txtnum2.Text)
End Sub
```

۳ – فرم و پروژه را با نام Calculator ذخیره کنید، سپس پروژه را اجرا کرده و دو عدد در کادرهای متن بنویسید؛ سپس روی دکمه جمع کلیک کنید و نتیجه را بررسی کنید، همان‌طور که می‌بینید حاصل جمع اعداد تایپ شده در بخش Result نمایش داده می‌شود.

همان‌طورکه در رویداد Click دکمه جمع مشاهده کردید برای جمع دو عدد که در کادرهای متن تایپ می‌شوند از کاراکتر + استفاده می‌شود. در زبان ویژوال‌بیسیک برای انجام عملیات جمع از این کاراکتر استفاده می‌شود. به علاوه مقادیری که در کادرهای متن تایپ می‌شوند ماهیت غیرعددی دارند و جمع آن‌ها به صورت عادی امکان‌پذیر نیست؛

بنابراین با استفاده از تابع Val رشته عددی تایپ شده در کادرهای متن را به نوع عددی تبدیل کرده و سپس آن‌ها را با هم جمع و در خصوصیت Caption کنترل برچسب ذخیره می‌کنید تا نتیجه محاسبه نمایش داده شود.

۴ – در این مرحله رویدادهای Click دکمه‌های تفریق، ضرب و تقسیم را به این صورت و مشابه دکمه جمع تنظیم کنید:

```
Private Sub cmdminus_Click()
    lblresult.Caption=Val(txtnum1.Text)-Val(txtnum2.Text)
End Sub

Private Sub cmdmulti_Click()
    lblresult.Caption=Val(txtnum1.Text)*Val(txtnum2.Text)
  End Sub
Private Sub cmddiv_Click()
    lblresult.Caption=Val(txtnum1.Text)/Val(txtnum2.Text)
End Sub
```

۵ – پروژه را اجرا کرده و عملکرد برنامه را در رابطه با چهار عمل اصلی بررسی کنید.

۶ – پنجره برنامه را ببندید و سپس تغییرات اعمال شده را ذخیره کنید.

۳–۴ نحوه تعریف و استفاده از ثابت‌ها در ویژوال‌بیسیک

قبلاً روش‌های مختلفی را برای ذخیره‌سازی اطلاعات فراگرفته‌اید. روش‌های دیگری نیز وجود دارد که یکی از آن‌ها استفاده از ثابت‌هاست (Constant). علاوه بر ثابت‌هایی که به طور آماده در ویژوال‌بیسیک وجود دارند (برای مثال، می‌توان به ثابت‌هایی که برای تعیین رنگ‌ها استفاده می‌شوند، اشاره کرد.) می‌توانید ثابت‌های مورد نیاز خود را نیز در برنامه تعریف کنید. تفاوتی که بین ثابت‌ها و متغیرها وجود دارد این است که مقدار یک ثابت پس از تعریف قابل تغییر نیست، در صورتی‌که مقدار متغیرها را می‌توانید بارها در هنگام اجرای برنامه تغییر دهید.

برای استفاده از ثابت‌ها دلایلی نیز وجود دارد به عنوان نمونه گاهی اوقات مقادیر عددی و

رشـــته‌ای خاصی به طور مکرر در برنامه اســـتفاده می‌شوند که ذکر آن‌ها توأم با صرف زمان و ایجاد اشتباه در سطح برنامه خواهد شد، اما با استفاده از یک ثابت می‌توان مقدار مورد نظر را در آن ذخیره کرد و در هر جایی از برنامه نام ثابت را به جای مقدار مربوطه به کار برد. در این صورت برنامه خواناتر شده و ایجاد تغییرات در برنامه در آینده سریع‌تر و با اشتباه کمتری همراه خواهد بود.

برای تعریف یک ثابت به این صورت عمل کنید:

Const نام ثابت As نوع ثابت = مقدار

تعریف ثابت‌ها و قوانین نام‌گذاری آن‌ها شـــبیه به متغیرهاســـت. نوع ثابت می‌تواند از انواع داده‌ها در ویژوال‌بیســـیک باشـــد که در جدول ۱-۴ ارایه شده‌اند. پس از تعیین نوع داده برای ثابت و قراردادن علامت مســـاوی، مقدار موردنظر برای ذخیره‌ســـازی در ثابت قـــرار می‌گیرد. مقادیر رشـــته‌ای را داخـــل علامت (") و مقادیر تاریخ و ســـاعت را داخل علامت (#) قرار دهید. به عنوان مثال به این نمونه‌ها توجه کنید:

```
Const   Conlength As Integer = 20
Const   Conname  As String = "Ali"
Const   ConcreateDate As  Date = #12/5/98#
```

مثال ۳: یک پروژه طراحی کنید که کاربر توانایی محاسبه حجم یک استوانه دلخواه را داشته باشد.

۱ – یک پروژه از نوع Standard EXE با یک فرم و کنترل‌های مربوطه مطابق شکل ۳-۴ و جداول ۱۱-۴ الی ۱۳-۴ ایجاد کنید.

جدول ۱۱-۴ خصوصیات فرم

خصوصیت	مقدار
Name	frmcyl
Caption	Cylinder

شکل ۳-۴

جدول ۱۲-۴ خصوصیات کنترل‌ها

کنترل / خصوصیت	Label	Label	Label	Label
Name	lblr	lblh	lblc	lblresult
Caption	Radius :	Height :	Capacity :	۰

جدول ۱۳-۴ خصوصیات کنترل‌ها

کنترل / خصوصیت	TextBox	TextBox	Command Button
Name	txtr	txth	cmdcalc
Caption	--	--	Calculate&

۲ – خصوصیت Border Style فرم را روی مقدارFixed Dialog -3 تنظیم کنید. به وسیله ایـن خصوصیت می‌تـوان کادر دور فرم، دکمه‌های کنترلی منوی کنتـرل و قابلیت تغییر اندازه را در فرم تعیین کرد. این خصوصیت می‌تواند یکی از موارد ارائه شـده در جدول ۱۴-۴ را کسب نماید.

جدول ۱۴-۴ خصوصیت Border Style

مقدار خصوصیت	توضیح
0-None	فرم بدون کادر، نوارعنوان، منوی کنترل، دکمه‌های کنترلی و با اندازه ثابت
1- Fixed Single	فـرم با کادر، نوارعنوان، دکمه‌های کنترلی و قابلیت تغییر اندازه با دکمه‌های Minimize و Maximize
2- Sizable	فرم با کادر، نوارعنوان، دکمه‌های کنترلی و قابلیت تغییر اندازه
3- Fixed Dialog	فرم بـا کادر، منوی کنترل، نوارعنوان، بدون دکمه‌های کنترلی Minimize و Maximize و بـا اندازه ثابت. در ضمن در هنـگام اجرای برنامه آیکن فرم در Taskbar نمایش داده نمی‌شود.
4-Fixed Tool Window	فرم با کادر، نوارعنوان و دکمه Close (با اندازه کوچک) و بدون قابلیت تغییر اندازه، به علاوه در هنگام اجرای برنامه آیکن فرم در Taskbar نمایش داده نمی‌شود.
5- Sizable Tool Window	فرم با کادر، نوارعنوان و دکمه Close (با اندازه کوچک) و قابلیت تغییر اندازه، به علاوه در هنگام اجرای برنامه آیکن فرم در Taskbar نمایش داده نمی‌شود.

نکته منوی کنترل در نوارعنوان فرم‌ها در سمت چپ عنوان فرم به صورت یک آیکن کوچک نمایش داده می‌شـوند و شامل گزینه‌هایی مانند Move، Size، Minimize، Maximize، Restore و Close است که برای مدیریت فرم‌ها استفاده می‌شود.

۳ – رویداد Click کنترل دکمه فرمان را به این صورت تنظیم کنید:

```
Private Sub cmdcalc_Click()
    Const pi As Single=3.14159265358979
    lblresult.Caption=pi*Val(txtr.Text)^2*Val(txth.Text)
End Sub
```

۴– پروژه و فرم را با نام cylinder ذخیره کنید، سپس پروژه را اجرا کرده و برنامه را به ازای چند مقدار آزمایش کنید.

۵– به اجرای پروژه خاتمه دهید و به پنجره ویژوال‌بیسیک بازگردید.

برای اسـتفاده از مقدار عدد π از ثابت pi اسـتفاده شده است که موجب دقیق‌تر شدن محاسبات و تایپ راحت‌تر دستورات خواهد شد.

۴–۴ متغیرهای ایستا، محلی و عمومی در ویژوال‌بیسیک

در این‌جا لازم اسـت انواع متغیرها را از نظر میدان دید (Scope) و شناسـایی آن‌ها در بخش‌های مختلف برنامه مورد بررسـی قرار دهید. میدان دید یک متغیر معین می‌کند که بعد از تعریف یک متغیر چه بخش‌هایی از برنامه می‌توانند به آن دسترسـی داشـته باشند و از آن استفاده کنند.

وقتـی متغیری در یک رویه رویداد تعریف می‌شـود فقط دسـتورات داخل آن رویه رویداد می‌توانند به آن دسترسی داشته باشند چنین متغیرهایی را متغیرهای محلی (Local) در سـطح رویه می‌گویند. در این صورت با خروج از رویه رویداد، متغیر در حافظه پاک شده و مقدار ذخیره شده در آن از بین می‌رود. برای تعریف این نوع از متغیرها می‌توانید از دستورDim استفاده کنید.

نوع دیگری از متغیرهای محلی نیز وجود دارند که با استفاده از کلمه کلیدی Static به جای Dim تعریف می‌شوند، این نوع متغیرها تمام ویژگی‌های متغیرهای محلی را دارند

بـــا این تفاوت که پس از خاتمه اجرای رویه‌ای که در آن تعریف شـــده‌اند، مقدار آن‌ها در حافظـــه از بین نمی‌رود و در اجـــرای بعدی رویه مربوطه می‌توانیـــد از مقادیر قبلی متغیر استفاده کنید.

به عنوان مثال به این نمونه‌ها توجه کنید:

```
Dim intno As Integer
Static sngk As Single
```

اما گاهی اوقات لازم است تا یک متغیر در تمام رویه‌های یک ماژول فرم قابل استفاده باشد، برای این کار می‌توانید متغیر مورد نظر را در بخش تعاریف ماژول فرم و با استفاده از کلمـــه کلیدی Private تعریف کنید. در صورت تعریف یک متغیر به صورت فوق تمام رویه‌هـــای موجود در همان ماژول می‌توانند از متغیر مربوطه اســـتفاده کنند و تا زمانی‌که فرم از حافظه خارج نشـــده است، مقدار خود را حفظ نمایند. شکل کلی نحوه استفاده از این نوع متغیرها به این صورت است:

Private نام متغیر As نوع داده

به عنوان مثال در صورتی که متغیرهای زیر در بخش تعاریف ماژول فرم تعریف شوند در همه سطح ماژول فرم قابل شناسایی و استفاده می‌باشد.

```
Private strname  As String
Dim snggrasde  As Single
```

نکته به جای استفاده از کلمه کلیدی Private می‌توانید از کلمه کلیدی Dim نیز استفاده کنید.

۴-۵ عملگرهای رشته‌ای در ویژوال‌بیسیک

علاوه بر عملگرهای ریاضی، مقایسه‌ای و منطقی، عملگرهای دیگری نیز وجود دارند که توانایی انجام عملیات را روی رشته‌ها فراهم می‌کنند.

یکی از این عملگرها، عملگر جمع (+) اســـت ایـــن عملگر می‌تواند دو یا چند مقدار رشـــته‌ای را به یکدیگـــر اضافه کند به عنوان مثال عبارت "Visual" + "Basic"، رشـــته‌ای به صـــورت "VisualBasic" را به‌وجود می‌آورد. البته این عملگر را می‌توانید برای متغیرها و

خصوصیات از نوع رشته‌ای به کار ببرید.

نکته به جای عملگر+ می‌توانید از عملگر & نیز استفاده کنید.

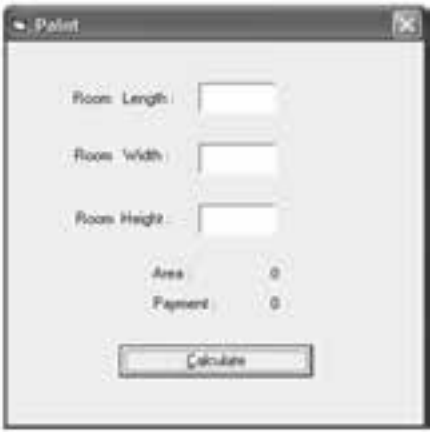
مثال ۴: پروژه‌ای طراحی کنید که هزینه رنگ‌آمیزی یک سالن یا اتاق را با دریافت ابعاد آن‌ها و با فرض این‌که هزینه هر مترمربع ۴۰۰۰۰ریال باشد، محاسبه نماید. برای این کار عملیات زیر را به ترتیب انجام دهید:

یک پروژه از نوع Standard EXE ایجاد کنید که شامل یک فرم و کنترل‌های مربوطه مطابق شکل ۴-۴ و جداول ۴-۱۵ الی ۴-۱۸ باشد؛ سپس خصوصیت BorderStyle فرم را روی مقدار 3-Fixed Dialog تنظیم کنید.

سپس رویداد Click دکمه فرمان Calculate را به این صورت تنظیم کنید:

```
Private Sub cmdcalc_Click()
  Dim dblarea As Double
  Dim curpay As Currency
  dblarea=2*Val(txtrl.Text)*Val(txtrh.Text)+ 2*Val(txtrw.Text)*Val(txtrh.Text)+ _
 Val(txtrl.Text)*Val(txtrw.Text)
  curpay = dblarea * 40000
 lblar.Caption=Str(dblarea)+ "M2"
 lblpay.Caption=Str(curpay)+"Rials"
End Sub
```

در این رویداد پس از محاسبه مساحت سطح و هزینه رنگ‌آمیزی شده، مقادیر آن‌ها به ترتیب در متغیرهای dblarea و curpay ذخیره می‌شوند، سپس با استفاده از کنترل‌های برچسب مساحت سطح رنگ شده و کل هزینه برای پرداخت نمایش داده می‌شود. در دستوراتی که برای نمایش نتیجه محاسبات استفاده شده‌اند، ابتدا با استفاده از تابع Str مقادیر عددی به مقادیر رشته‌ای تبدیل می‌شوند، سپس با استفاده از عملگر + آن‌ها را به هم متصل کرده و در خصوصیت Caption کنترل‌های برچسب ذخیره می‌شوند تا در روی فرم نمایش داده شوند.



شکل۴-۴

جدول ۱۵-۴ خصوصیات فرم

خصوصیت	مقدار
Name	frmpaint
Caption	Paint

جدول ۱۶-۴ خصوصیات کنترل‌ها

کنترل / خصوصیت	Label	Label	Label
Name	lblrl	lblrw	lblrh
Caption	Room Length :	Room Width :	Room Height :

جدول ۱۷-۴ خصوصیات کنترل‌ها

کنترل / خصوصیت	Label	Label	Label	Label	Command Button
Name	lbla	lblar	lblpay	lblpayr	cmdcalc
Caption	Area:	۰	Payment :	۰	&Calculate

جدول ۱۸-۴ خصوصیات کنترل‌ها

کنترل / خصوصیت	TextBox	TextBox	TextBox
Name	txtrl	txtrw	txtrh

نکته در صورتی که خطوط مربوط به دستورات بیش از حد طولانی باشند می‌توانید آن‌ها را در چند خط تایپ کنید و آن‌ها را با کاراکتر زیرخط به هم مرتبط کنید.

۳ – پروژه و فرم را با نام Paint ذخیره و اجرا کنید، سپس اعداد ۴، ۲/۵ و ۳ متر را به ترتیب برای طول، عرض و ارتفاع وارد کنید و روی دکمه Calculate کلیک کرده و نتیجه محاسبات را بررسی کنید.

۴ – برنامه را برای مقادیر دیگری نیز آزمایش کنید، سپس از برنامه خارج شوید.

تمرین:

پروژه‌ای طراحی کنید که نام و نام خانوادگی هر فرد دلخواهی را دریافت کند سپس ترتیب آن دو را در کنار هم با یک کنترل برچسب نمایش دهد.

عـلاوه بر عملگر جمـع، عملگر دیگری به نام Like نیز وجـود دارد که توانایی انجام عملیات روی رشـته‌ها را دارد. این عملگر یک رشـته را با رشته دیگری مقایسه کرده و نتیجه را مشخص می‌کند. می‌توانید از این عملگر به صورت زیر استفاده کنید:

عبارت رشته‌ای دوم Like عبارت رشته‌ای اول

اگر عبارت رشته‌ای اول با عبارت رشته‌ای دوم مشابه باشد، مقدار True و اگر عبارت رشـته‌ای اول با عبارت رشـته‌ای دوم مشابه نباشد، مقدار False به دست می‌آید، به عنوان مثـال مقایسـه "ali" Like "ali" نتیجـه True و مقایسـه "ali" Like "alireza" نتیجه False در پی‌خواهد داشـت. به علاوه با اسـتفاده از کاراکترهای * ، ? ، # و [ ] می‌توانید ترکیب‌های متنوعی از مقایسه‌ها را ایجاد کنید.

کاراکتر* بیانگر گروهی از کاراکترها، کاراکتر ?بیانگر هر کاراکتر دلخواه، کاراکتر # بیانگر هر کاراکتر رقمی و کاراکترهای [ ] بـرای ایجاد بـازه‌ای از کاراکترها در مقایسـه به کار می‌روند. برای مشـخص شـدن بهتر مطلـب به مثال‌هایـی که در جدول ۱۹–۴ ارایه شده‌اند، توجه کنید.

جدول ۱۹–۴

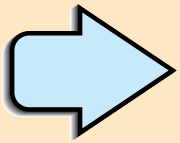
عبارت	
"aBBBa" Like "a*a"	True
"F" Like "[A-Z]"	True
"F" Like "[!A-Z]"	False
"a۲a" Like "a#a"	True
"*BAT۱۲۳khg" Like "B?T"	True
"*CAT۱۲۳khg" Like "B?T"	False
"aM۵b" Like "a[L-P]#[!c-e]"	True

همان‌طورکه ملاحظه می‌کنید اولین مقایسه نتیجه True در بردارد، زیرا * می‌تواند جانشین گروهی از کاراکترها شود. در عبارت دوم حرف F در بازه حروف A تا Z قرار دارد، پس نتیجه True به دست می‌آید؛ اما در عبارت سوم از کاراکتر ! استفاده شده است و به این معنی است که F جزء حروف A تا Z نیست و نادرست است. در عبارت بعدی وجود کاراکتر # در بین حروف، پذیرش کاراکتر رقمی ۲ را در رشته "a۲a" توجیه می‌کند. عبارت‌های بعدی نیز به همین صورت قابل بررسی هستند و ترکیبی از حالت‌های گفته شده در آن‌ها به کار رفته است.

تمرین:

نتیجه این مقایسه‌ها را به‌دست آورید:

```
"Vb6"  Like  "?b#"
"Microsoft"  Like  "M*"
"Windows XP"  Like  "?i[a-z]*"
"Visual C++"  Like  "Visual ?[!A-Z]++"
```



Learn in English

Variables

In Visual Basic, you use variables to temporarily store values during the execution of an application. Variables have a name (the word you use to refer to the value the variable contains) and a data type (which determines the kind of data the variable can store).

Declaring Variables

To declare a variable is to tell the program about it in advance. You declare a variable with the Dim statement, supplying a name for the variable:

Dim variablename [**As** type]

واژه‌نامه

ثابت	Constant
مقادیر پولی	Currency
تعریف کردن	Declare
تعیین کردن	Determine
اجرا	Execution
اولویت	Precedence
خصوصی	Private
عمومی	Public
ذخیره کردن	Store
تأمین کردن، عرضه کردن	Supply
موقت	Temporary
متغیر	Variable
ترکیبی	Variant

خلاصه مطالب

- برای انجام عملیات ریاضی از عملگرهای ریاضی استفاده می‌شود.
- انواع داده‌ها در ویژوال‌بیسیک عبارتند از: داده‌های عددی، رشته‌ای، منطقی، پولی، تاریخ و ساعت و داده‌های ترکیبی.
- برای تعریف متغیر از دستور Dim استفاده می‌شود.
- در صورت استفاده از دستور Option Explicit، باید متغیر را قبل از استفاده از آن تعریف کرد.
- متغیرهایی که در یک رویداد تعریف می‌شوند فقط در همان رویداد قابل شناسایی و استفاده هستند به این متغیرها، متغیرهای محلی می‌گویند.
- متغیرهای خصوصی در تمام رویدادهای ماژول فرم قابل شناسایی و استفاده هستند و برای تعریف آن‌ها از بخش تعاریف استفاده می‌شود.
- برای محاسبه زمان و اجرای دستورات مورد نظر به‌طور خودکار و مکرر در فاصله‌های زمانی معین از کنترل زمان‌سنج استفاده می‌شود.
- عملگرهای رشته‌ای در ویژوال‌بیسیک عبارتند از: + ، & و Like.

شکل کلی نحوه استفاده از متدها به این صورت است:

نام متد .نام شیء

آزمون نظری

۱ – کدام نوع از انواع داده برای تعریف یک متغیر از نوع پولی مناسب‌تر است؟

الف– Single ب– Double ج– Currency د– Boolean

۲ – کدام عملگر ریاضی نسبت به سایرین از اولویت پایین‌تری برخوردار است؟

الف – جمع ب– Mod ج– توان د– ضرب

۳ – کدام گزینه برای تعریف یک متغیر از نوع اعداد صحیح بلند درست است؟

الف– % ب– & ج– ! د– #

۴ – کدام عملگر رشته‌ای مشابه عملگر جمع در رشته‌هاست؟

الف– % ب– & ج– Like د– #

۵ – اگر متغیری در بخش تعاریف ماژول فرم با دستور Dim معرفی شود متغیر ..... نامیده می‌شود.

الف– عمومی ب– محلی ج– خصوصی د– ایستا

۶ – کدام نوع داده برای تعریف اعداد مثبت کوچک‌تر از ۲۵۵ مناسب است؟

الف– String ب– Integer ج– Byte د– Long

۷ – حاصل عبارت"Computer2004" Like "C?m?u*####" چیست؟

الف– True ب– False

ج– عبارت اشتباه است. د–در هنگام اجرا پیام خطا نمایش داده می‌شود.

۸ – متغیرهای خصوصی یا Private در چه محدوده‌ای قابل شناسایی هستند؟

الف– فرم‌ها

ب– ماژول فرمی که در آن تعریف شده‌اند.

ج– تمام پروژه

د– رویدادی که در آن تعریف شده‌اند.

۹ – The variable can be declared with ..... statement in within a procedure.

a– private b– public c– dim d– as

۱۰– مفهوم متغیر و نوع داده را توضیح دهید و نحوه تعریف متغیرها را بیان کنید.

۱۱– تفاوت بین متغیرهای عمومی و محلی را شرح دهید.

۱۲– انواع عملگرهای ریاضی و عملگرهای رشته‌ای را توضیح دهید.

آزمون عملی

۱ – پروژه‌ای طراحی کنید که با توجه به نوع سپرده‌گذاری در یک بانک، میزان بهره سالیانه را مطابق این جدول محاسبه کرده و نمایش دهد.

نوع سپرده	میزان بهره
قرض‌الحسنه	ـــــ
کوتاه مدت	۸ درصد
بلند مدت ۳ ساله	۱۲ درصد
بلند مدت ۶ ساله	۱۵ درصد

۲ – یک پروژه از نوع Standard EXE طراحی کنید که شامل فرم به صورتی که در شکل نشان داده شده و با این شرایط باشد.

الف– نام کاربر با اندازه ۲۴ و با رنگ سیاه و ضخیم در روی فرم نمایش داده شود.

ب– کاربر بتواند با کلیک روی دکمه‌های Red، Green و Blue رنگ قلم را در نام کاربر روی رنگ قرمز، سبز یا آبی تنظیم کند.

ج– در صورت کلیک روی دکمه Hide نام کاربر مخفی شود؛ به علاوه دکمه Hide غیرفعال و دکمه Show فعال شود.

د– در صورت کلیک روی دکمه Show نام کاربر نمایش داده شود؛ به علاوه دکمه Show غیرفعال و دکمه Hide فعال شود.

هـ – فرم و پروژه را با نام Show ذخیره کنید.

۳ – عبارات زیر را به زبان ویژوال‌بیسیک تبدیل کنید.

الف– $\dfrac{\dfrac{a}{b}-\dfrac{c}{d}}{۵-\dfrac{e}{f}}$

ب– $(۳x+y^{۲})\times(۶z-۱۲)$

ج– $۴x^{۲}-۵x+۳$

د– $\dfrac{\left(\dfrac{۱۵}{۷y}\right)}{۹z-۱۸x}$

۴ – عبارات زیر را به صورت ریاضی بنویسید.

الف– ۱/۲*x+۱۴*z^۳

ب– x\y*z/w^u

ج– ۲۴۳–a/b^۲+۳*z

۵ – پس از انجام دستورات زیر حاصل متغیر x چیست؟

```
Dim str1 As String , str2 As String
Dim x As Integer
str 1 ="34"
str2="Ali"
x=Val (str1)+Val (str2)
```

۶ – با توجه به تمرین ۵، حاصل عبارات str1=str1+str2 و str1=str(x)+str2 چیست؟

واحد کارپنجم

هدف جزیی

توانایی استفاده از دستور شرطی IF
و عملگرهای مقایسه‌ای و منطقی

هدف‌های رفتاری

پس از مطالعه این واحد کار از فراگیر انتظار می‌رود که:

۱- بتواند از دستور If با حالت یک دستوری و چند دستوری استفاده کند.

۲- عملگرهای مقایسه‌ای و منطقی و نحوه استفاده از آن‌ها را توضیح دهد.

۳- اولویت اجرای عملگرهای مقایسه‌ای و منطقی را بیان کند.

۴- یک فرم جدید را به پروژه اضافه کرده، از کنترل دکمه فرمان استفاده کند و خصوصیات مربوط به کنترل دکمه فرمان را توضیح دهد.

۵- بتواند با متدهای Show و Hide فرم کار کند.

۶- به کمک کنترل‌های کادر تصویر و Image تصاویر مورد نظر خود را نمایش دهد.

۷- بتواند از کادرهای پیام و کادرهای ورود داده استفاده کند.

کلیات

در طراحی پروژه‌های برنامه‌نویسی معمولاً نیاز به کنترل روند برنامه خواهید داشت، چرا که یک برنامه باید در شرایط مختلف عملکرد مناسبی را از خود نشان دهد و توانایی تصمیم‌گیری را با توجه به حالات و رویدادهای متفاوتی که در هنگام اجرای برنامه رخ می‌دهند، داشته باشد. در چنین شرایطی استفاده از ساختارهای تصمیم اجتناب‌ناپذیر است. زبان برنامه‌نویسی ویژوال‌بیسیک امکانات لازم را برای کنترل روند اجرای برنامه در اختیار برنامه‌نویسان قرار می‌دهد.

۱–۵ نحوه استفاده از ساختار IF و عملگرهای مقایسه‌ای در برنامه‌ها

ساختارهای تصمیم به شما اجازه می‌دهند که نحوه اجرای برنامه را به میل خود تغییر داده و تنظیم کنید. یکی از انواع دستورات کنترلی در ویژوال‌بیسیک دستور If است. این دستور با بررسی شرط یا شرط‌های تعیین شده و با توجه به درست یا نادرست بودن نتیجه بررسی‌ها، روند اجرای برنامه را معین می‌کند.

به‌طورکلی می‌توانید از دستور If به دو صورت یک دستوری و چند دستوری استفاده کنید. شکل کلی دستور If در حالت یک دستوری به این صورت است:

```
If شرط‌ها Then یک دستور Else یک دستور
```

در این دستور، ابتدا شرط و مقایسه مورد نظری که پس از دستور If قرار می‌گیرد، بررسی می‌شود و در صورتی‌که نتیجه بررسی شرط درست باشد دستوری که پس از Then قرار گرفته است، اجرا می‌شود سپس دستوری که پس از If قرار گرفته اجرا می‌شود. اما در صورتی‌که نتیجه بررسی شرط نادرست باشد از اجرای دستوری که بعد از Then قرار گرفته است صرف نظر کرده و دستوری که پس از Else قرار گرفته است، اجرا می‌شود. سپس دستور خطی که پس از If قرار گرفته، اجرا می‌شود. البته در صورت تمایل می‌توانید از بخش Else صرف نظر کرده و از آن استفاده نکنید، در نتیجه دستور پس از Then فقط در زمان درست بودن شرط اجرا می‌شود و در غیر این صورت بدون اجرای آن، کنترل برنامه به دستوری که پس از If قرار گرفته، منتقل می‌شود. با این حال گاهی لازم است از دستور If با حالت چند دستوری استفاده کنید. شکل کلی دستور If در این حالت در

ادامه آمده است:

If (شرط(ها)) **Then**
⋮ **دستور(ات)**
Else
⋮ **دستور(ات)**
End If

در این حالت نیز مانند حالت یک دستوری اگر نتیجه شرطی که پس از دستور If قرار گرفته است، درست باشد دستورات بین Then و Else اجرا می‌شوند و سپس دستوری که پس از End If قرار گرفته است، اجرا می‌شود. اما در صورتی‌که نتیجه بررسی شرط نادرست باشد از اجرای دستورات بین Then و Else صرف نظر می‌شود و دستورات بین بخش Else و End If اجرا می‌شوند.

اکنون دستوری که پس از End If قرار دارد، اجرا می‌شود. البته در این حالت نیز استفاده از بخش Else اختیاری است و می‌توانید از این دستور به صورت زیر نیز استفاده کنید:

If شرط‌ها **Then**
⋮ **دستور(ات)**
End If

در این حالت اگر نتیجه بررسی شرط درست باشد، دستورات بین Then و End If اجرا می‌شوند و سپس دستوری که پس از End If قرار دارد، اجرا می‌شود و اگر نتیجه این بررسی نادرست باشد از اجرای دستورات موجود بین Then و End If صرف‌نظر شده و اجرای برنامه به دستور پس از End If منتقل می‌شود.

مثال ۱: پروژه‌ای طراحی کنید که یک عدد دلخواه را دریافت کرده و زوج یا فرد بودن آن را مشخص کند. برای این منظور عملیات زیر را به ترتیب انجام دهید:
۱- برنامه ویژوال‌بیسیک را اجرا کنید و یک پروژه جدید از نوع Standard EXE به همراه یک فرم و کنترل‌های آن را مطابق شکل ۱-۵ ایجاد کنید و خصوصیت‌های آن‌ها را مطابق جدول‌های ۱-۵ و ۲-۵ تنظیم نمایید:

شکل ۱–۵

جدول ۱–۵ خصوصیات فرم

خصوصیت	مقدار
Name	Frmeven or odd
Caption	Even or Odd

جدول ۲–۵ خصوصیات کنترل‌ها

کنترل / خصوصیت	Label	Label	Text Box	Command Button
Name	lblno	lblresult	txtno	cmdevenodd
Caption	Enter No:	---	---	&Even or Odd

۲ – در این مرحله رویداد Click دکمه فرمان cmdevenodd را به‌صورت زیر تنظیم کنید:

```
Private Sub cmdevenodd_Click()
    Dim intno As Integer
    Dim R As Integer
    Intno = Val (txtno. Text)
    R= intno Mod 2
    If  R= 0  Then
        lblresult. Caption = " Your Number Is Even."
    Else
        lblresult. Caption = "Your Number Is Odd."
End Sub
```

در این رویداد ابتدا رشته عددی که در کنترل کادر متن txtno تایپ می‌شود با استفاده از تابع Val به عدد تبدیل شده و در متغیر intno ذخیره می‌شود سپس با استفاده از عملگر Mod باقیمانده تقسیم متغیر intno را بر عدد ۲ محاسبه می‌کند و در متغیر R قرار می‌دهد.

در ایـن مرحله دسـتور If شـرط R=۰ را بررسـی می‌کند اگر مقـدار R برابر با صفر باشـد عـدد زوج اسـت و نتیجـه بررسـی شـرط درسـت (True) می‌شـود بنابرایـن پیـام"Your Number Is Even." نمایـش داده خواهد شـد اما اگر عدد فرد باشـد مقدار R برابر با ۱ خواهد شـد بنابراین نتیجه بررسـی شـرط نادرسـت (False) می‌شـود و پیام "Your Number Is Odd" نمایش داده می‌شود.

۳ – فرم و پروژه را با نام even or odd ذخیره کنید.

۴ – پروژه را اجرا کنید و برای اعداد زوج و فرد آن‌را آزمایش کنید.

۵ – اجرای پروژه را متوقف کرده و به پنجره ویژوال‌بیسیک بازگردید.

تمرین:

پـروژه مثال ۱ را به‌گونه‌ای تنظیم کنید تا برای اعداد منفی و صفر با نمایش یک پیام خطا کاربر را مطلع نماید.

۲–۵ کنترل کادر تصویر (Picture Box)

یکی دیگر از ویژگی‌های زبان ویژوال‌بیسیک امکان استفاده آسان از تصاویر، فایل‌های گرافیکی و عکس‌ها در برنامه‌هاست. برای نمایش تصاویر می‌توانید از کنترل کادر تصویر اسـتفاده کنید. این کنترل می‌تواند انواع فایل‌های گرافیکی مانند bmp، gif، jpg، ico و cur را نمایش دهد.

مهم‌ترین خصوصیت کنترل کادر تصویر Picture است. این خصوصیت مسیر و نام فایل گرافیکی یا تصویر را مشـخص می‌کند. برای این کار کافی اسـت روی دکمه  در روبـه‌روی خصوصیت مزبور کلیک کنید و به‌وسـیله کادرمحـاوره Load Picture تصویر موردنظرتان را انتخاب کنید.

کنترل کادر تصویر دارای خصوصیتی به نام Align می‌باشـد. این خصوصیت می‌تواند محـل نمایش تصویر را تعیین کند. مقادیری که این خصوصیت می‌تواند کسـب کند در جدول ۳–۵ ارائه شده‌اند.

جدول ۳-۵ مقادیر خصوصیت Align

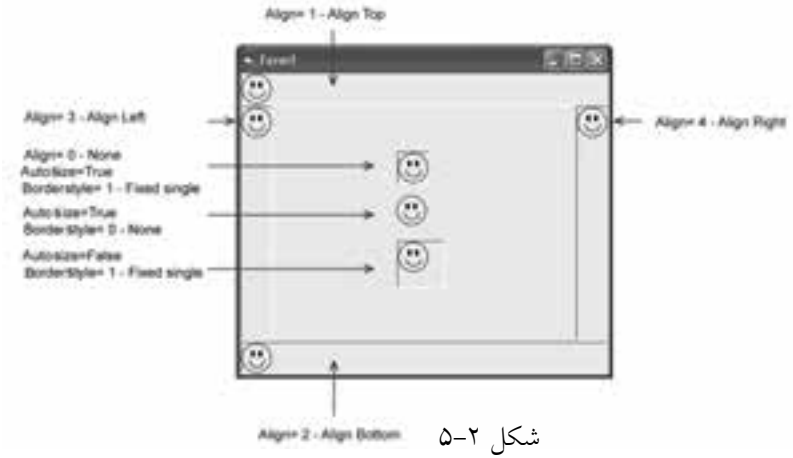
توضیح	مقدار خصوصیت
نمایش کنترل با استفاده از مقادیر خصوصیت‌های Top و Left	0 - None
نمایش کنترل در بالای فرم	1- Align Top
نمایش کنترل در پایین فرم	2-Align Bottom
نمایش کنترل در سمت چپ فرم	3 -Align Left
نمایش کنترل در سمت راست فرم	4 -Align Right

یکی دیگر از خصوصیت‌های کادر تصویر خصوصیت AutoSize است و می‌تواند مقدار True یا False را کسب کند. در صورتی‌که مقدار این خصوصیت را روی True تنظیم کنید اندازه کنترل کادر تصویر با اندازه تصویر به‌صورت خودکار یکسان می‌شود اما اگر مقدار این خصوصیت روی False تنظیم شود اندازه کنترل کادر تصویر با توجه به مقدار خصوصیت‌های Height و Width تنظیم می‌شود. به عبارت دیگر در این حالت تصویر با اندازه خود و کنترل کادر تصویر نیز با اندازه خود نمایش داده می‌شوند.

کنترل کادر تصویر دارای خصوصیت دیگری به نام BorderStyle است. با استفاده از این خصوصیت می‌توانید کنترل کادر تصویر را با یک کادر در اطراف آن یا بدون کادر نمایش دهید. خصوصیت BorderStyle می‌تواند مقادیر ارائه شده در جدول ۴-۵ را کسب کند.

جدول ۴-۵

توضیح	مقدار خصوصیت
کنترل کادر تصویر بدون کادر نمایش داده می‌شود.	0 -None
کنترل کادر تصویر با کادر نمایش داده می‌شود.	1 -Fixed Single



شکل ۲-۵

کنترل کادر تصویر علاوه بر رویدادهای مشترکی که با سایر کنترل‌ها دارد، دارای دو رویداد Change و Resize می‌باشد. رویداد Change وقتی رخ می‌دهد که تصویر نمایش داده شده به وسیله کنترل کادر تصویر تغییر کند و رویداد Resize نیز زمانی اجرا می‌شود که اندازه کنترل تغییر می‌کند.

مثال ۲: پروژه‌ای طراحی کنید که یک تصویر را روی یک فرم نمایش دهد. برای این منظور ابتدا فرم اصلی را طراحی کرده، آن‌را کامل کنید، سپس عملیات بعد را به ترتیب انجام دهید:

۱ – برنامه ویژوال‌بیسیک را اجرا کنید و یک پروژه جدید از نوع Standard EXE به همراه یک فرم مطابق شکل ۳–۵ ایجاد کنید که مشخصات فرم آن مطابق جدول ۵–۵ باشد.

شکل ۳–۵

جدول ۵-۵ خصوصیات فرم

خصوصیت	مقدار
Name	frmshow
Caption	Show Picture

۲ – برای نمایش تصویر می‌توانید از کنترل کادر تصویر (PictureBox) استفاده کنید. برای استفاده از این کنترل ابتدا به پنجره طراحی Form۱ بروید و در جعبه‌ابزار روی‌آیکن PictureBox دابل کلیک کنید تا کنترل کادر تصویر روی فرم قرار گیرد، سپس خصوصیات کنترل کادر تصویر را مطابق جدول ۶–۵ تنظیم کنید.

جدول ۶-۵

خصوصیت	مقدار
Name	picshow
AutoSize	True

۳ – اکنون باید تصویر نمایشی را برای کنترل کادر تصویر انتخاب کنید، بنابراین روی کنتـــرل کادر تصویر کلیک نمایید و در پنجره خصوصیات، خصوصیت Picture این کنترل را پیدا کنید.

۴ – روی دکمه ... در روبه‌روی این خصوصیت کلیک کنید و بااستفاده از کادرمحاوره Load Picture فایل EARTH.ICO را از مسیر زیر انتخاب کنید، همان‌طورکه مشاهده خواهید کرد تصویر انتخاب شده روی فرم قابل مشاهده است.

D:\Program Files\Microsoft Visual Studio\Common\Graphics\Icons\Elements

۵ – یـــک کنترل دکمـــه فرمان روی فرم قرار داده ســـپس ویژگی‌های آن‌را با توجه به جدول ۷–۵ تنظیم‌کنید. از این کنترل برای خروج از برنامه استفاده می‌شود.

جدول ۷-۵

مقدار	خصوصیت
cmdexit	Name
&Exit	Caption

نکته برای آن‌که یک کلید دسترســـی ســـریع برای یک کنترل ایجاد کنید، کافی است که در خصوصیـــت Caption آن و قبل از حرف مورد نظر یک کاراکتر & تایپ کنید، در این‌صورت فشردن کلید ترکیبی Alt به همراه حرفی که کاراکتر & قبل از آن تایپ شده است، معادل کلیک کردن روی کنترل دکمه فرمان خواهد بود.

۶ – اکنون باید رویداد مناســـبی را انتخاب کنید تا با اســـتفاده از آن و در زمان کلیک روی دکمه فرمانExit، برنامه خاتمه یابد. بنابراین در پنجره طراحی فرم روی دکمه فرمان Exit دابل‌کلیک کنید تا پنجره کد فرم فعال شود و رویداد( ) cmdexit_Click در اختیار شما قرار گیرد. سپس دستور زیر را در این رویداد تایپ کنید:

Unload frmshow

در زمان اجرا وقتی کاربر روی دکمه Exit کلیک کند رویداد ( ) cmdexit_Click به دنبال آن دستور Unload اجرا خواهد شد، این دستور اطلاعات فرم frmshow را از حافظه خارج می‌کند، به عبارت دیگر برنامه خاتمه می‌پذیرد.

۷ – برنامه را اجرا کنید و پس از مشاهده فرم و تصویر نمایشی روی دکمه Exit کلیک کنید تا برنامه خاتمه یابد.

۸ – پروژه و فرم را با نام showpicture ذخیره کنید.

مثـال ۳: اکنون پروژه showpicture را کامل کنیـد تا در هنگام اجرای برنامه، ابتدا یک کلمه رمز سـؤال شـود و در صورت صحیح بودن کلمه رمـز، اجازه ورود کاربر به برنامه داده شود و فرم اصلی برنامه با تصویر مورد نظر نمایش داده شود. برای این منظور عملیات زیر را انجام دهید:

۱ – به پنجره ویژوال‌بیسیک و پروژه showpicture بروید.

۲ – بـرای اضافه کردن یک فرم جدید، در مکان خالی از پنجره پروژه کلیک راسـت کنید و گزینه Add را انتخاب کنید، سپس روی گزینه Form کلیک کنید (شکل ۵–۵).

شکل ۴–۵

شکل ۵–۵

کادر محاوره Add Form مطابق شکل ۶–۵ نمایش داده می‌شود که دو زبانه New و Existing در آن دیده می‌شود. برای ایجاد یک فرم جدید می‌توانید از زبانه New و برای اضافه کردن یک فرم که قبلاً ایجاد و در روی دیسک ذخیره شده است، از زبانه Existing استفاده کنید.

شکل ۶-۵

البته می‌توانید فرم‌ها را با یکی از این روش‌ها نیز اضافه کنید:

الف– برای دسترسی به کادر محاوره Add Form می‌توانید گزینه Add Form را از منوی Project در نوار منوی ویژوال‌بیسیک انتخاب کنید.

ب– برای دسترسی به کادر محاوره Add Form می‌توانید روی علامت مثلثی شکل دکمه در نوار ابزار استاندارد کلیک کنید و سپس گزینه Form را انتخاب کنید.

۳ – در کادر محاوره Add Form ابتدا زبانه New را انتخاب کنید، سپس از کادر لیست موجود در زیرزبانه‌ها، روی آیکن Form و بعد روی دکمه Open کلیک کنید. همان‌طور که مشاهده می‌کنید یک پنجره طراحی دیگر در پنجره ویژوال‌بیسیک نمایش داده می‌شود به علاوه در پنجره‌های پروژه، خصوصیات و تعیین موقعیت نیز، فرم اضافه شده قابل مشاهده است.

۴ – ویژگی‌های فرم جدید را تنظیم کنید و سپس آن‌را با نام Log On در روی دسک‌تاپ ذخیره کنید.

۵–اکنون باید یک کنترل کادر متن، برچسب و دکمه فرمان با مشخصات ارائه شده در جدول ۹-۶ به پنجره frmlogon اضافه کنید.

۶ – حال باید عملکرد برنامه را طوری تنظیم کنید تا اگر عبارتی که در کادر متن تایپ می‌شود با کلمه رمز برابر باشد، امکان ادامه اجرای برنامه فراهم شود. برای این‌کار از رویداد کلیک کنترل دکمه فرمان استفاده می‌شود. این رویداد زمانی اجرا می‌شود که کاربر روی این دکمه کلیک کند یا کلید Enter را بفشارد. بنابراین کنترل cmdok را در پنجره طراحی

فرم frmlogon انتخاب کنید، سپس روی این کنترل دابل کلیک کنید. همان‌طورکه مشاهده می‌کنید پنجره کد فرم نمایش داده می‌شـود و رویداد cmdok_Click در دسترس شما قرار می‌گیـرد. با فرض این‌که کلمه رمز BASIC باشـد، رویـداد Click این کنترل را به صورت زیر تنظیم کنید:

```
Private Sub cmdok_Click()
    If txtpass.Text = "BASIC" Then frmshow.Show
End Sub
```

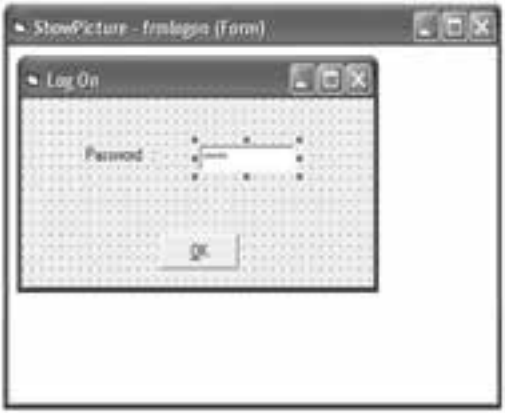
در رویداد cmdok_Click از دستور If استفاده شده است. در این دستور کلمه رمزی که کاربر در کنترل کادر متن تایپ کرده و در خصوصیت Text کادر متن نگهداری می‌شـود با استفاده از علامت تساوی (=) با عبارت "BASIC" که به عنوان کلمه رمز، در نظر گرفته شـده است، مقایسـه می‌شـود. اگر کلمه رمز تایپ شـده با عبارت "BASIC" برابر باشد، دستوری که بعد از کلمه کلیدی Then قرار دارد، اجرا می‌شود و در غیر این‌صورت یعنی اشتباه بودن کلمه رمز، دستوری که بعد از Then قرار گرفته است، اجرا نمی‌شود و دستور خط بعد از If اجرا خواهد شد.

اکنون به بررسـی دستور frmshow.Show که بعد از کلمه کلیدی Then قرارگرفته است، می‌پردازیم. Show متدی است که می‌تواند یک فرم را که اطلاعات آن در حافظه بارگذاری شـده اسـت، نمایش دهد. البته در صورتی‌که فرم در حافظه بارگذاری نشـده باشد آن‌را بارگذاری کرده و سپس نمایش می‌دهد.

نکته متد Hide نیز از انواع متدهای فرم اسـت و عملکرد آن برعکس متد Show می‌باشـد و از نمایش فرم جلوگیری کرده و آن‌را مخفی می‌کند، اما اطلاعات فرم را از حافظه خارج نمی‌کند.

۷- می‌دانیـد که در زمـان تایپ کلمه رمز در مکان مربوطه، برای جلوگیری از مشـاهده کلمه رمز توسـط سـایر کاربران، به جای کاراکترهایی که تایپ می‌شوند کاراکتر دیگری نمایـش داده می‌شـود. به این منظـور می‌توانید از خصوصیـت PasswordChar کادر متن استفاده کنید. این خصوصیت کاراکتر تعیین شده را به جای کاراکترهایی که در کادر متن تایپ می‌شوند، نمایش می‌دهند، اما محتویات تایپ شده در کادر متن به همان صورت در

خصوصیت Text کادر متن نگهداری می‌شود. بنابراین خصوصیت فوق را پیدا کرده و در کادر متن روبه‌روی این خصوصیت کاراکتر ستاره (*) را تایپ کنید. همان‌طورکه مشاهده می‌کنید پنج کاراکتر ستاره در پنجره طراحی فرم و در کادر متن نمایش داده می‌شود. به عبارت دیگر به تعداد کاراکترهای عبارت موجود در خصوصیت Text کادر متن، کاراکتر ستاره نمایش داده می‌شود (شکل ۷–۵).



شکل ۷–۵

۸ – در این مرحله برای تعداد کاراکترهای کلمه رمز، محدودیت ایجاد کنید. خصوصیت MaxLength کادر متن، حداکثر تعداد کاراکترهایی را که در این کنترل می‌توان تایپ کرد، معین می‌کند. مقدار پیش فرض این خصوصیت صفر است که تعداد کاراکترهای ورودی نامحدود را پشتیبانی می‌کند. مقدار این خصوصیت را برای کنترل txtpass روی مقدار ۵ تنظیم کنید. در نتیجه هنگام اجرای برنامه می‌توانید حداکثر ۵ کاراکتر را در کادر متن تایپ کنید.

۹ – دستور زیر را در رویداد Load فرم frmlogon تایپ کنید تا در هنگام نمایش این فرم کاربر مجبور نباشد مقدار پیش فرض Text۱ را در کادر متن پاک کند و کلمه رمز را تایپ نماید.

```
txtpass.Text =""
```

۱۰ – در این مرحله تغییرات ایجاد شده در فرم‌ها و پروژه را ذخیره کرده و پروژه را اجرا کنید.

همان‌طورکه می‌بینید پنجره اصلی برنامه یعنی فرم frmshow نمایش داده شده و فرم frmlogon مشاهده نمی‌شود. در واقع در هر برنامه می‌توان یک فرم را به عنوان فرمی که

در ابتدای اجرای برنامه نمایش داده می‌شـود، معرفی کرد. اگر از چند فرم در پروژه خود اسـتفاده می‌کنید باید فرم مورد نظر را برای نمایش در ابتـدای اجرای برنامه معین کنید.

برای ایـن‌کار باید از کادر محاوره Project Properties استفاده کنید.

۱۱ – اجـرای برنامـه را متوقف کنید و از منوی Project گزینه ShowPicture Properties را انتخاب کنید تا کادر محاوره Project Properties نمایش داده شود. در زبانه General و در کادر لیست Startup Object روی دکمه ▼ کلیک کنید (شـکل ۸–۵). سـپس از کادر لیستی که نمایش داده می‌شود، فرم frmlogon را برگزیده و روی دکمه OK کلیک کنید.

شکل ۸–۵

۱۲ – برنامـه را اجـرا کنید و عبارت BASIC را در کادر متن تایپ کنید، سـپس روی دکمـه OK کلیک کنید. همان‌طورکه می‌بینید فرم frmshow نمایش داده می‌شـود و هر دو فرم قابل مشاهده خواهند بود.

۱۳ – پنجره‌های برنامه را ببندید و بار دیگر برنامه را اجرا کنید و این بار کلمه رمز را به صورت basic تایپ کنید و روی دکمه OK کلیک کنید. همان‌طورکه مشـاهده خواهید کرد فرم برنامه نمایش داده نخواهد شد.

۱۴ – اجرای برنامه را متوقف کنید و به پنجره ویژوال‌بیسیک بازگردید.

نکته در صورتی‌که بخواهید یک متغیر در تمام فرم‌های پروژه قابل دسترسـی و شناسـایی باشد می‌توان آن‌را با استفاده از کلمه کلیدی Public در بخش تعاریف ماژول فرم تعریف نمود. بـه این نوع از متغیرها، متغیر عمومی می‌گویند. شـکل کلی نحوه تعریف متغیرهای عمومی به صورت زیر است:

Public نام متغیر As نوع داده

به عنوان مثال به این نمونه توجه کنید:

```
Public curpayment  As  Currency
```

برای دسترسی به متغیرهای عمومی در سایر فرم‌ها می‌توان نام فرمی را که متغیر در آن تعریف شده است را قبل از نام متغیر قرار داد. به عنوان مثال اگر متغیر curpayment در بخش تعاریف فرمی با نام frmmain تعریف شده باشد برای مقداردهی به آن در فرم frmsalary در همان پروژه می‌توان به این صورت عمل کرد:

```
frmmain. Curpayment = 2000
```

تمرین:

۱ – برنامه را به شکلی تنظیم کنید که با کلیک روی دکمه Exit تمام پنجره‌ها به‌طور هم‌زمان بسته شوند.

۲ – برنامه را به‌گونه‌ای تنظیم کنید که در هنگام خروج از برنامه، پنجره کوچکی به همراه دو دکمه Yes و No مبنی بر تأیید برای خروج از نرم‌افزار نمایش داده شود. در صورتی‌که کاربر روی دکمه Yes کلیک کند، برنامه خاتمه یابد و در صورتی که روی دکمه No کلیک کند، به برنامه بازگردد.

در این‌جا لازم است با انواع عملگرهای مقایسه‌ای آشنا شوید. برای انجام هرگونه مقایسه می‌توانید از این عملگرها مطابق جدول ۸–۵ استفاده کنید.

جدول ۸–۵ عملگرهای مقایسه‌ای ویژوال‌بیسیک

عملگر	مفهوم عملگر	مثال	نتیجه مقایسه
=	تساوی	"Ali"="BASIC"	نادرست
<=	کوچک‌تر یا مساوی	"BASIC"<="basic"	درست
>=	بزرگ‌تر یا مساوی	"BASIC">="Basic"	نادرست
>	کوچک‌تر	۵ > ۱۰	نادرست
<	بزرگ‌تر	۲/۷۵ < ۱۴/۵	درست
< >	عدم تساوی یا نامساوی	-۲ <> -۲	نادرست

نکته در مقایسه کاراکترها و عبارات رشته‌ای، از کد اسکی کاراکترها برای انجام مقایسه استفاده می‌شود.

عملگرهای مقایسه‌ای را می‌توانید برای مقایسه هر نوع داده‌ای استفاده کنید، با توجه به این‌که مقادیری که مقایسه می‌شوند از یک نوع باشند.

گاهی اوقات ممکن است از چند عملگر مقایسه‌ای در یک عبارت استفاده شود. در این‌صورت این سؤال مطرح می‌شود که کدام یک از عملگرها زودتر پردازش می‌شوند. در واقع در یک عبارت مقایسه‌ای با چند عملگر، عملیات با توجه به ترتیب قرار گرفتن عملگرها از چپ به راست عبارت انجام می‌شود و اگر از پرانتز استفاده شود، اولویت و حق تقدم عملیات با عملگرهای موجود در داخل پرانتز خواهد بود. به عنوان مثال این عبارت را در نظر بگیرید: ۲ > ۱۰ = ۰

در عبارت مذکور ابتدا مقایسه ۲ >۱۰ بررسی می‌شود که نتیجه نادرست یا False را دربر خواهد داشت، سپس مقایسه False = ۰ بررسی می‌شود که نتیجه آن درست یا True خواهد بود.

تمرین:

۱ – پروژه showpicture را به شکلی تنظیم کنید تا در صورتی‌که کلمه رمز وارد شده صحیح باشد، فرم frmlogon از روی دسک‌تاپ برداشته شده و سپس فرم frmshow مشاهده شود، هم‌چنین در صورت نادرست بودن کلمه رمز رنگ کاراکترهای تایپ شده قرمز شود و کاربر توانایی ورود کلمه رمز دیگری را نداشته باشد. در ضمن در صورت ورود کلمه رمز به‌صورت اشتباه دکمه OK غیرفعال شود و رنگ زمینه کادر متن نیز به رنگ آبی درآید.

۲ – پروژه‌ای طراحی کنید که کاربر با تایپ اسامی کشورها بتواند سایر مشخصات مانند وسعت، جمعیت، پرچم، مذهب و واحد پول آن‌ها را نمایش دهد و هم‌چنین بتواند برای ورود به نرم‌افزار یک کلمه کاربر و یک کلمه رمز را وارد کند.

۳–۵ نحوه استفاده از عملگرهای منطقی برای ترکیب شرط‌ها

گاهی اوقات ممکن است لازم باشد در دستورات شرطی بیش از یک شرط مورد بررسی قرار گیرد و براساس نتیجه حاصل از بررسی تمام شرط‌ها اجرای برنامه صورت گیرد در این حالت می‌توان از عملگرهای منطقی مانند AND، OR یا NOT استفاده کرد. این عملگرها امکان ترکیب دو یا چند شرط را با یکدیگر فراهم می‌کنند.

مثال ۴: یکی دیگر از موارد قابل توجه در مثال قبل، این است که ویژوال‌بیسیک بین حروف کوچک و بزرگ تفاوت قائل می‌شود، به عنوان مثال عبارت BASIC را با عبارت basic مساوی در نظر نمی‌گیرد. البته اگر بخواهید می‌توانید این حالت را تغییر دهید و به ویژوال‌بیسیک بگویید که بین حروف کوچک و بزرگ تفاوت قائل نشود.

به این منظور باید دستور Option Compare Text را در بخش تعاریف (Declarations Section) در پنجره کد فرم frmlogon بنویسید (شکل ۹–۵).

شکل ۹–۵

برنامه را اجرا کنید و کلمه رمز را به صورت basic تایپ کنید و روی دکمه OK کلیک کنید. همان‌طور که مشاهده خواهید کرد این بار کلمه رمز پذیرفته می‌شود. روی دکمه Exit کلیک کنید تا برنامه خاتمه یابد، سپس تغییرات را ذخیره نمایید. البته این کار را می‌توانید به روش دیگری نیز انجام دهید که در این‌جا به ذکر آن می‌پردازیم. با استفاده از

عملگرهای منطقی در ویژوال‌بیسیک می‌توانید ترکیب‌های مختلفی از چند شرط را ایجاد کنید، به عنوان مثال در پروژه قبل برای آن‌که برنامه هر دو نوع عبارت BASIC یا basic را به عنوان کلمه رمز بپذیرد، می‌توانید خط اول از دستور If موجود در رویداد cmdok_Click را به این صورت تنظیم کنید:

```
If txtpass.Text = "BASIC" Or txtpass.Text = "basic" Then
```

سپس دستور Option Compare Text را از بخش تعاریف حذف کنید و پروژه را با دو حالت مختلف کلمه رمز آزمایش کنید و نتیجه را بررسی نمایید. همان‌طور که ملاحظه کردید در هر دو بار کلمه رمز پذیرفته می‌شود.

عملگر Or ("یا" منطقی) یکی از عملگرهای منطقی در ویژوال‌بیسیک است. از عملگرهای منطقی برای ایجاد ترکیب‌های مورد نظر از چند شرط مختلف، استفاده می‌شود. در صورتی‌که دو یا چند شرط با استفاده از عملگر منطقی Or با هم ترکیب شوند، نتیجه بررسی تمام ترکیب شرطی، زمانی درست است که حداقل یکی از شرط‌ها درست باشد یا به عبارت دیگر زمانی ترکیب شرطی نتیجه نادرست خواهد داشت که تمام شرط‌ها نادرست باشند. نتیجه ترکیب شرطی که با عملگر Or به‌وجود می‌آید در حالت‌های مختلف در جدول ۹-۵ ارایه شده است. منظور از A و B در این جدول عبارت‌های مقایسه‌ای است که با عملگر Or ترکیب شرطی را به وجود آورده و می‌توانند ارزش درست و یا نادرست داشته باشند.

جدول ۹-۵

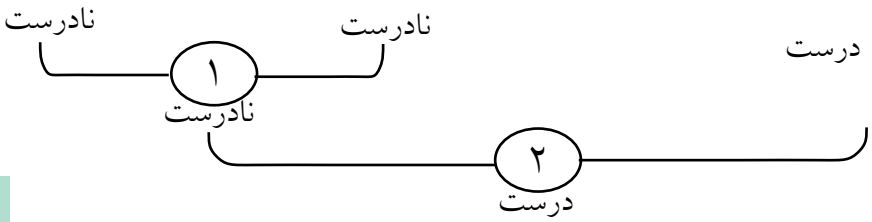
A	B	A Or B
نادرست	نادرست	نادرست
درست	نادرست	درست
نادرست	درست	درست
درست	درست	درست

به عنوان مثال این عبارت را در نظر بگیرید:

```
"ALI" = "ali" Or "d" < "B" Or "Tehran" > = "TABRIZ"
```

در عبارت فوق از دو عبارت مقایسه‌ای استفاده شده است که با عملگر منطقی Or با یکدیگر ترکیب شده‌اند، پس برای به‌دست آوردن نتیجه این ترکیب به صورت زیر عمل می‌کنید:

```
"ALI" ="ali"    Or    "d" < "B"    Or    "Tehran" > = "TABRIZ"
```



عـلاوه بر عملگـر منطقـی Or دو عملگـر منطقی دیگر کـه معمولاً مورد اسـتفاده برنامه‌نویسان قرار می‌گیرند عبارتند از: عملگر منطقی And (" و" منطقی) و عملگر منطقی Not (نقیض).

نتیجه عبارت شـرطی که با And ایجاد می‌شـود در جدول ۱۰–۵ ارایه شـده اسـت. همان‌طور که ملاحظه می‌کنید فقط زمانی ترکیب شـرطی نتیجه درست خواهد داشت که نتیجه هر دو مقایسـه و شـرط به طور هم‌زمان درست باشـد، در غیر این‌صورت ترکیب شرطی، نتیجه نادرست خواهد داشت.

نتیجه ترکیب شرطی که عملگر Not روی آن انجام می‌شود، مطابق جدول ۱۱–۵ است. همان‌طور که مشـاهده می‌کنید عملگر Not می‌تواند ارزش یک شرط یا ترکیب شرطی را معکوس کند.

جدول ۱۱–۵

A	Not A
درست	نادرست
نادرست	درست

جدول ۱۰–۵

A	B	A And B
درست	درست	درست
درست	نادرست	نادرست
نادرست	درست	نادرست
نادرست	نادرست	نادرست

اکنون ارزش درستی این عبارت را به‌دست آورید:

```
Not ("ali" < = "ALI") Or ("Reza" > "R") And ("Computer" < > "computer")
```

قبل از این‌که ارزش درستی عبارت قبل را به‌دست آورید، ذکر این نکته ضروری است که عملگرهای منطقی نیز مانند عملگرهای مقایسه‌ای، نسبت به هم اولویت اجرایی دارند و اگر از عملگرهای مشابه در یک عبارت استفاده شود، حق تقدم و اولویت اجرای آن‌ها از چپ به راسـت خواهد بود، اما اگر از عملگرهای متفاوت اسـتفاده شـود، حق تقدم اجرای آن‌ها با توجه به جدول ۱۲–۵ و از بالا به پایین خواهد بود.

جدول ۱۲–۵

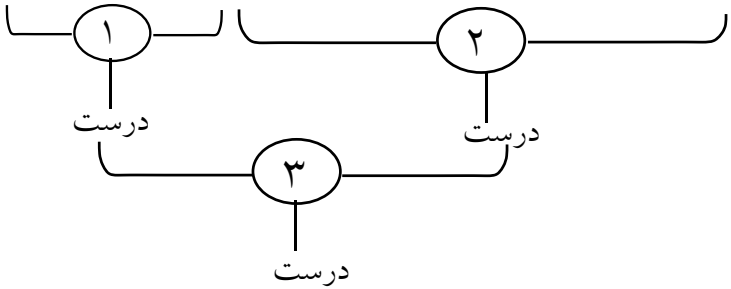
Not
And
Or

نکته اولویت اجرای عملگرهای مقایسه‌ای از عملگرهای منطقی بالاتر است.

با توجه به مطالب گفته شده نحوه ارزیابی عبارت مورد نظر به این صورت خواهد بود:

Not ("ali" < = "ALI") Or ("Reza" > "R") And ("Computer" < > "computer")

Not(نادرست) Or (درست) And (درست)



تمرین:

در صورتی‌که مقدار A=۲ ، B=–۱۴ و C=۰ باشد، ارزش درستی این عبارات را به‌دست آورید:

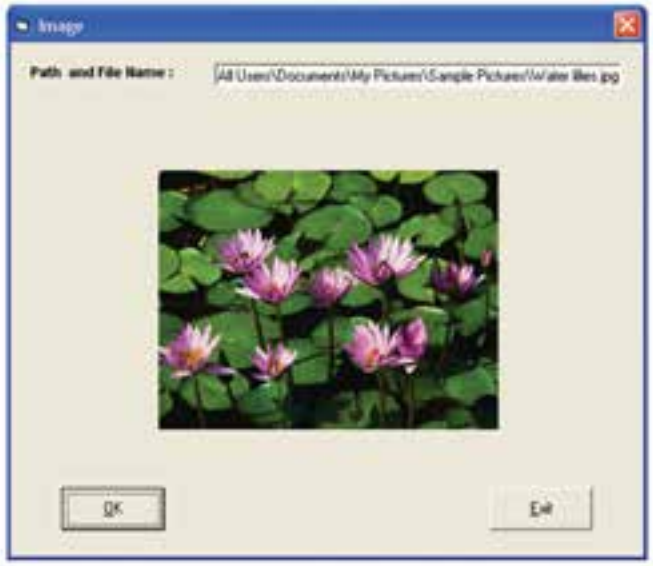
A < C Or Not B< > C AND "A" = "Q" Or C < = ۲

"۶" > = "۹"AND A > B Or Not C > B

۵–۴ کنترل تصویر

ویژوال‌بیسیک کنترل دیگری به‌نام تصویر Image دارد که مانند کادر تصویر برای نمایش تصاویر به‌کار می‌رود. خصوصیات این کنترل مشابه کنترل کادر تصویر است، اما کنترل تصویر (Image) یک خصوصیت به نام Stretch دارد که در کنترل کادر تصویر وجود ندارد. این خصوصیت از نوع منطقی بوده و در صورتی‌که روی مقدار False تنظیم شود، اندازه کنترل با اندازه تصویر هماهنگ می‌شود. اما اگر این خصوصیت روی مقدار True تنظیم شود اندازه تصویر با توجه به اندازه کنترل تنظیم می‌شود. می‌توانید از این خصوصیت برای بزرگ و کوچک کردن تصاویر استفاده کنید.

مثال ۵: پروژه‌ای طراحی کنید که کاربر بتواند هر فایل گرافیکی مورد نظر خود را در یک پنجره مطابق شکل ۱۰–۵ نمایش دهد.



شکل ۵–۱۰

۱– برنامه ویژوال‌بیسیک را اجرا کنید و یک پروژه از نوع Standard EXE ایجاد کنید، سپس کنترل‌های مربوطه را مطابق جدول ۵–۱۳ ایجاد کنید و فرم و پروژه را با نام Image ذخیره نمایید.

جدول ۵–۱۳

کنترل / خصوصیت	Form	TextBox	Label	Command Button	Image	Command Button
Name	frmimage	txtfilename	lblfilename	cmdok	imgshow	cmdexit
Caption	Image	ــــ	Path and File Name:	&OK	ــــ	Exit
Stretch	ــــ	ــــ	ــــ	ــــ	True	ــــ
AutoSize	ــــ	ــــ	True	ــــ	ــــ	ــــ

۲ – به پنجره کد فرم بروید و رویداد Load فرم و Change کادر متن را به صورت زیر تنظیم کنید. این دستورات باعث می‌شوند تا زمانی‌که کاربر، نام و مسیر فایلی را در کادر متن تایپ نکند، دکمه OK فعال و قابل استفاده نباشند.

```
Private Sub Form_Load()
    cmdok.Enabled =False
End Sub
Private Sub txtfilename_change()
    cmdok.Enabled = True
End Sub
```

۳ – این دستور را در رویداد Click دکمه Exit تایپ کنید:

```
Unload frmimage
```

ذکر این نکته ضروری است که برای خروج اطلاعات فرم فعال از حافظه و پایان دادن به برنامه می‌توانید از دستور Unload Me استفاده کنید، کلمه کلیدی Me به نام فرم فعال اشاره می‌کند.

۴ – برای آن‌که پس از تایپ نام، مسیر فایل و کلیک روی دکمه OK، تصویر مورد نظر نمایش داده شود، رویداد Click دکمه فرمان OK را به این صورت تنظیم کنید:

```
Private Sub cmdok_Click()
    imgshow.Picture= LoadPicture (txtfilename.Text)
End Sub
```

با کلیک روی دکمه OK این رویداد اجرا شده و تصویر مورد نظر به‌وسیله تابع Load Picture نمایش داده می‌شود. تابع LoadPicture می‌تواند نام و مسیر یک فایل گرافیکی را دریافت کند و آن‌را در یک کنترل کادر تصویر (PictureBox) یا تصویر (Image) نمایش دهد. شکل کلی نحوه استفاده از این تابع به صورت زیر است:

نام کنترل تصویر یا کادر تصویر.Picture = LoadPicture ("نام و مسیر فایل گرافیکی")

نکته باید نام و مسیر فایل را داخل کاراکتر " قرار دهید. اگر از تابع فوق به صورت ("") Load Picture استفاده کنید تصویر نمایش داده شده در کنترل کادر تصویر یا کنترل تصویر، برداشته شده و حذف می‌شود.

۵ – پروژه را اجرا کرده و نام و مسیر یک فایل گرافیکی را در کادر متن بنویسید و روی دکمه OK کلیک‌کنید.

۶ – اجرای پروژه را خاتمه دهید و به پنجره ویژوال‌بیسیک بازگردید.

۵–۵ اولویت اجرای عملگرها نسبت به یکدیگر

تاکنون اولویت اجرای عملگرها در گروه خود را فرا گرفتید. عملگرهای موجود در گروه‌های مختلف نیز در صورتی‌که در یک عبارت قرار بگیرند، دارای اولویت اجرایی نسبت به هم هستند، اولویت عملگرها در تمام گروه‌ها در جدول ۱۴–۵ ارایه شده‌اند. بالاترین اولویت مربوط به عملگرهای ریاضی و پایین‌ترین اولویت مربوط به عملگرهای منطقی می‌شود.

جدول ۱۴-۵

عملگرهای ریاضی براساس اولویت‌های ذکر شده
عملگر اتصال رشته‌ها (&)
عملگرهای مقایسه‌ای براساس اولویت‌های ذکر شده
عملگرهای منطقی براساس اولویت‌های ذکر شده

۶–۵ نحوه استفاده از کادرهای پیام در ویژوال‌بیسیک

احتمالاً تاکنون با کادرهای پیام در ویندوز برخورد کرده‌اید، کادرهای پیام معمولاً برای تأیید یا لغو عملیات یا مطلع کردن کاربران از یک اتفاق یا خطا استفاده می‌شوند و معمولاً پیامی را به همراه یک یا چند دکمه به کاربر نمایش می‌دهند تا با توجه به نیاز خود تصمیم‌گیری مناسب را انجام دهد. به عنوان مثال می‌توان به کادرهای پیامی که در شکل‌های ۱۱–۵ و ۱۲–۵ نمایش داده شده‌اند، اشاره کرد.

شکل ۱۱–۵

شکل ۱۲–۵

ویژوال‌بیسیک توانایی ایجاد انواع مختلفی از کادرهای پیام را برای شما فراهم می‌کند. بـــرای نمایش کادرهای پیام در ویژوال‌بیســـیک از تابع MsgBox اســـتفاده می‌شـــود. نحوه استفاده از این تابع به یکی از این روش‌ها امکان‌پذیر است:

(عنوان , نوع و تعداددکمه‌ها و آیکن کادرپیام , پیام‌نمایشی) MsgBox = نام‌متغیرازنوع‌صحیح

عنوان , نوع و تعداد دکمه‌ها و آیکن کادر پیغام , پیام نمایشی MsgBox

پیام نمایشـــی و عنوان می‌توانند یک عبارت یا متغیر رشته‌ای باشند که به ترتیب، پیام نمایشی داخل کادر پیام و عنوان کادر پیام را تعیین می‌کنند.

بخش دوم در تابع MsgBox می‌تواند نوع و تعداد دکمه‌ها, نوع آیکن کادر پیام و دکمه پیـــش فرض را در کادر پیام تعیین کند. برای این کار می‌توانید مقادیر عددی یا رشـــته‌ای ارایه شده در جداول ۱۵-۵ الی ۱۷-۵ را با استفاده از کاراکتر + با هم ترکیب کنید.

جدول ۱۵-۵ مقادیر مربوط به نوع و تعداد دکمه‌ها در کادر پیام

توضیح	ثابت عددی	ثابت رشته‌ای
کادر پیام با دکمه OK	۰	vbOKOnly
کادر پیام با دکمه OK و Cancel	۱	vbOKCancel
کادر پیام با دکمه Abort، Retry و Ignore	۲	vbAbortRetryIgnore
کادر پیام با دکمه Yes، No و Cancel	۳	vbYesNoCancel
کادر پیام با دکمه Yes و No	۴	vbYesNo
کادر پیام با دکمه Retry و Cancel	۵	vbRetryCancel

جدول ۱۶-۵ مقادیر مربوط به نوع آیکن در کادر پیام

توضیح	ثابت عددی	ثابت رشته‌ای
آیکن پیام خطای بحرانی	۱۶	vbCritical
آیکن پرسش	۳۲	vbQuestion
آیکن پیام هشدار	۴۸	vbExclamation
آیکن پیام اطلاعات	۶۴	vbInformation

جدول ۱۷-۵ مقادیر مربوط به دکمه پیش‌فرض

توضیح	ثابت عددی	ثابت رشته‌ای
اولین دکمه، دکمه پیش‌فرض است.	صفر	vbDefaultButton1
دومین دکمه، دکمه پیش‌فرض است.	۲۵۶	vbDefaultButton2
سومین دکمه، دکمه پیش‌فرض است.	۵۱۲	vbDefaultButton3

نکته دکمه پیش‌فرض دکمه‌ای است که در زمان نمایش کادر پیام، فوکوس را در اختیار می‌گیرد.

وقتی که کاربر روی دکمه‌ای در کادر پیام کلیک کند، یکی از مقادیر ارائه شده در جدول ۱۸-۵ بازگشت داده می‌شود. با ذخیره‌سازی این مقدار در یک متغیر از نوع Integer می‌توان با توجه به پاسخ کاربر اجرای دستورالعمل‌ها را کنترل و مدیریت کرد.

جدول ۱۸-۵ مقادیر مربوط به دکمه‌ای که کلیک می‌شود.

توضیح	ثابت عددی	ثابت رشته‌ای
اگر روی دکمه OK کلیک شود.	۱	vbOK
اگر روی دکمه Cancel کلیک شود.	۲	vbCancel
اگر روی دکمه Abort کلیک شود.	۳	vbAbort
اگر روی دکمه Retry کلیک شود.	۴	vbRetry
اگر روی دکمه Ignore کلیک شود.	۵	vbIgnore
اگر روی دکمه Yes کلیک شود.	۶	vbYes
اگر روی دکمه No کلیک شود.	۷	vbNo

به عنوان مثال به دستور زیر توجه کنید:

```
Intanswer=MsgBox("Data Entry Is Invalid!" ,
vbRetryCancel+vbCritical+vbDefaultButton2 , "Error")
```

در صورت اجرای این دستور کادر پیامی مطابق شکل ۱۳-۵ نمایش داده می‌شود. همان‌طور که ملاحظه می‌کنید در بخش دوم با استفاده از کاراکتر + مقادیر مربوط به نوع آیکن, تعداد و نوع دکمه‌ها و دکمه پیش‌فرض تعیین شده‌اند.

شکل ۱۳-۵

اگــر کاربــر روی دکمــه Retry کلیک کند مقــدار VbRetry (معادل عــدد ۴) در متغیر Intanswer ذخیره می‌شــود و اگر روی دکمه Cancel کلیــک کند مقدار VbCancel (معادل عدد ۲) در متغیر Intanswer ذخیره می‌شود.

نکته اگر نوع و تعداد دکمه‌ها تعیین نشده باشند کادر پیام با دکمه OK نمایش داده می‌شود. در صورت عدم انتخاب آیکن, هیچ آیکنی نمایش داده نخواهد شد. به علاوه اگر دکمه پیش‌فرض انتخاب نشده باشد، اولین دکمه در کادر پیام به عنوان دکمه پیش‌فرض انتخاب خواهد شد.

به عنوان مثال این دستور یک کادر محاوره مطابق شکل ۱۴-۵ ایجاد می‌کند.

```
Intanswer=MsgBox("Password Is Not Correct!?" , vbOKCancel)
```

شکل ۱۴-۵

مثال ۶: می‌خواهیم پروژه Image را به شکلی تنظیم کنیم تا هنگام خروج از برنامه پیام تأییدی به کاربر نمایش داده شود.

۱ – پروژه Image را باز کنید و به پنجره طراحی فرم بروید.

۲ – روی دکمه فرمان Exit دوبار کلیک کنید تا به رویداد Click دکمه فرمان Exit بروید.

۳ – دستورات رویداد Click دکمه فرمان Exit را به این صورت تغییر دهید.

```
Private Sub cmdexit_Click()

    Dim intanswer As Integerintanswer=MsgBox("DoYou Want To Exit?",

  VbYesNo + vbQuestion,"Exit")

  If intanswer =vbYes Then UnLoad me

End Sub
```

دراین رویداد با استفاده از تابع MsgBox کادر پیامی طراحی شده است که شامل عبارت Do You Want To Exit? و عنــوان Exit بــه همــراه دو دکمه Yes و No می‌باشــد. در صورت اجرای برنامه و کلیک روی دکمه Exit این رویداد اجرا می‌شود و با نمایش کادر پیام سبب

توقف اجرای برنامه می‌شود و تا زمانی که کاربر پاسخی به کادر پیام ندهد اجرای برنامه متوقف خواهد ماند؛ اما در صورتی که کاربر روی یکی از دکمه‌های کادر پیام کلیک کند تابع MsgBox مقداری مناسب با دکمه مربوطه در متغیر intanswer ذخیره می‌کند. اگر کاربر روی دکمه Yes کلیک کند، با بسته شدن فرم مقدار vbYes در متغیر intanswer ذخیره می‌شود و با استفاده از یک دستور If و بررسی مقدار intanswer اجرای برنامه خاتمه می‌یابد، اما در صورتی که کاربر روی دکمه No یا دکمه Close کلیک کند، مقدار vbNo در متغیر intanswer ذخیره می‌شود و برنامه بدون هیچ گونه تغییری در مقدار زمان ادامه می‌یابد.

۴ – تغییرات را ذخیره کرده و پروژه را اجرا نمایید. روی دکمه Exit کلیک کنید همان‌طور که ملاحظه می‌کنید کادر پیامی مطابق شکل ۱۵–۵ نمایش داده می‌شود.

۵ – روی دکمه No کلیک کنید تا به برنامه بازگردید.

۶ – روی دکمه Exit کلیک کنید. این بار روی دکمه Yes در کادر پیام کلیک کنید و تفاوت را با حالت قبل بررسی کنید.

۷ – اجرای برنامه را متوقف کنید و به محیط ویژوال‌بیسیک بازگردید.

شکل ۱۵–۵

۵–۷ نحوه استفاده از کادرهای ورود داده در ویژوال‌بیسیک

تاکنون برای ورود داده‌های خود به برنامه از کنترل کادر متن استفاده کرده‌اید. در ویژوال‌بیسیک روش راحت‌تری برای انجام این کار وجود دارد و آن استفاده از کادر ورود داده یا InputBox است. کادر ورود داده در واقع شبیه به یک کادر پیام است، با این تفاوت که دارای یک کادر متن برای ورود داده و دو دکمه OK و Cancel است (شکل ۱۶–۵). برای ایجاد کادرهای ورود داده در ویژوال‌بیسیک از تابع InputBox استفاده می‌شود.

شکل ۱۶–۵

این تابع می‌تواند یک کادر ورود داده ایجاد کند و در صورتی که کاربر روی OK کلیک کند داده‌های وارد شده در کادر متن را در یک متغیر رشته‌ای ذخیره می‌کند. اما اگر کاربر روی دکمه Cancel کلیک کند، یک رشته خالی ("") را در متغیر رشته‌ای ذخیره می‌کند. شکل کلی نحوه استفاده از این تابع به یکی از روش‌های زیر است:

(x,y ,مقدارپیش‌فرض‌در کادرمتن, عنوان کادر ورود داده
, پیام نمایشی در کادر ورودداده) InputBox = نام متغیر رشته‌ای

InputBox عنوان کادر ورود داده, پیام نمایشی در کادر ورود داده , x , y

مقادیر مربوط به پیام نمایشی و عنوان کادر ورود داده از نوع رشته‌ای است و مقدار پیش‌فرض مقداری است که در زمان نمایش کادر ورود داده در کادر متن نمایش داده می‌شود. x و y مقادیر عددی هستند که به ترتیب فاصله کادر ورود داده را از سمت چپ و بالای دسک‌تاپ تعیین می‌کنند.

نکته در صورتی که عنوان کادر ورود داده و کادر پیام تعیین نشود نام فایل پروژه در نوار عنوان آن‌ها نمایش داده می‌شود

مثال ۷: مطابق شکل ۱۷-۵ برنامه‌ای طراحی کنید که کاربر توانایی محاسبه محیط و مساحت هر دایره به شعاع دلخواه را داشته باشد و برای دریافت داده‌ها از کادر ورود داده که به وسیله دکمه Enter Radius فعال می‌شود، استفاده کند.

۱ - یک پروژه با یک فرم، یک کنترل دکمه فرمان و کنترل‌های برچسب مطابق شکل ۱۷-۵ ایجاد کنید.

شکل ۱۷-۵

۲ – رویداد Click کنترل دکمه فرمان را به این صورت تنظیم کنید:

```
Private  Sub  cmdr_Click()
    Dim  strr As String
    Dim  sngarea As Single, sngper As Single
    strr = InputBox("Enter Radius:","InputData:",0)
    sngper =2*3.14 *Val(strr)
    sngarea =3.14 *Val (strr)^2
    lblper.Caption = sngper
    lblarea.Caption = sngarea
End Sub
```

۳ – پروژه و فرم را با نام Circles و Circle ذخیره کنید، سپس پروژه را اجرا کنید.

۴ – روی دکمه Enter Radius کلیک کنید تا کادر ورود داده نمایش داده شود.

۵ – عدد ۳ را در کادر متن، کادر ورود داده تایپ کنید و روی دکمه OK در کادر ورود داده کلیک کنید. در این صورت مقدار تایپ شده در کادر متن در متغیر رشته‌ای strr ذخیره می‌شود، سپس با استفاده از تابع Val مقدار متغیر strr به عدد تبدیل می‌شود و با استفاده از فرمول‌های مربوطه محیط و مساحت دایره محاسبه می‌شود.

۶ – مجدداً روی دکمه Enter Radius کلیک کنید و عدد ۴ را در کادر متن، کادر ورود داده تایپ کنید و سپس روی دکمه Cancel کلیک کنید. در این‌صورت یک رشته خالی در متغیر strr ذخیره می‌شود و در تبدیل متغیر strr با تابع Val مقدار صفر تولید می‌شود که نتایج محاسبات نیز خواهد بود.

۷ – به اجرای پروژه خاتمه دهید و به پنجره ویژوال‌بیسیک بازگردید.

۸–۵ کنترل کادر علامت CheckBox

این کنترل امکان ایجاد انتخاب‌های دو حالتی را به‌وجود می‌آورد به شکلی که در صورت انتخاب کنترل (قرار دادن علامت چک‌مارک در داخل مربع کنترل) تصمیمات خاصی در برنامه اتخاذ شود و براساس آن پردازش و محاسبات انجام شود یا با برداشتن علامت

چک‌مارک از داخل مربع کنترل کادر علامت از تصمیمات موردنظر صرف‌نظر شود. بدین ترتیب کاربر می‌تواند با استفاده از این ویژگی‌ها شرایط خود را به برنامه اطلاع دهد.

جدول ۱۹-۵

خصوصیت	توضیح
Alignment	در صورتی که مقدار این خصوصیت روی0-Left Justify تنظیم شود، کادر علامت در سمت چپ عنوان کنترل و در صورتی که روی 1-Right Justify تنظیم شود کادر علامت در سمت راست عنوان آن نمایش داده می‌شود.
Style	در صورتی که مقدار این خصوصیت روی 0-Standard تنظیم شود شکل کنترل به صورت یک کادر علامت معمولی و وقتی روی مقدار 1-Graphical تنظیم شود به شکل یک دکمه فرمان دیده می‌شود.
Value	در صورتی که مقدار این خصوصیت برابر 0-Unchecked باشد نشان دهنده این است که کنترل انتخاب نشده است و اگر برابر 1-Checked باشد نشان دهنده این است که کنترل انتخاب شده است و اگر مقدار آن برابر2-Grayed باشد به معنی این است که کنترل دارای علامت چک مارک بوده و زمینه مربع کادر علامت نیز خاکستری رنگ است.

۵-۹ معرفی مهم‌ترین ویژگی‌های فرم‌ها و کنترل‌های کادر متن، دکمه فرمان و غیره

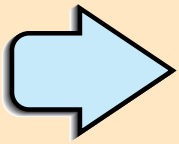
در این واحد کار با اشیایی نظیر فرم و کنترل‌های متعدد آشنا شدید، در پایان لازم است تا شما را با بعضی از خصوصیات و رویدادهای دیگر فرم و کنترل ذکر شده بیشتر آشنا کنیم.

جدول ۲۰-۵ خصوصیات فرم‌ها و کنترل‌ها

خصوصیت	توضیح
Appearance	اگر مقدار این خصوصیت روی ۰-Flat تنظیم شود، کنترل یا فرم به صورت مسطح و اگر روی مقدار ۱-3D تنظیم شود، کنترل یا فرم به صورت سه بعدی نمایش داده می‌شود.
BackColor	رنگ زمینه را در فرم و بعضی از کنترل‌ها مانند کادر متن تعیین می‌کند.
Enabled	اگر مقدار این خصوصیت روی True تنظیم شود، کنترل یا فرم فعال و قابل استفاده بوده و اگر روی False تنظیم شود، کنترل و فرم غیرفعال و غیر قابل استفاده می‌شود.
ForeColor	رنگ قلم را در فرم و بعضی از کنترل‌ها تعیین می‌کند.
Font	تنظیمات قلم را در فرم و بعضی از کنترل‌ها تعیین می‌کند.
Picture	یک تصویر را روی فرم و بعضی از کنترل‌ها تعیین می‌کند.
MousePointer	اشاره‌گر ماوس را برای فرم و بعضی از کنترل‌ها تعیین می‌کند.
MouseIcon	یک آیکن دلخواه را به عنوان اشاره‌گر برای فرم و بعضی کنترل‌ها تعیین می‌کند.
Visible	اگر مقدار این خصوصیت روی True تنظیم شود کنترل یا فرم نمایش داده شده و اگر مقدار این خصوصیت روی False تنظیم شود کنترل یا فرم نمایش داده نخواهد شد.
TabIndex	این خصوصیت مربوط به کنترل‌ها می‌شود و ترتیب حرکت روی کنترل‌ها را با استفاده از کلید Tab تعیین می‌کند و مقادیر مساوی یا بزرگ‌تر از صفر را کسب می‌کند.
TabStop	این خصوصیت از نوع منطقی بوده و مربوط به کنترل‌ها می‌شود و اگر روی مقدار False تنظیم شود در هنگام استفاده از کلید Tab، کنترل مربوط، فوکوس را دریافت نمی‌کند.
BorderStyle	اگر این خصوصیت در کنترل‌های برچسب و کادر متن روی مقدار ۰-None تنظیم شود، کنترل بدون حاشیه و کادر دور و اگر روی مقدار 1-Fixed Single تنظیم شود، کنترل با حاشیه و کادر دور مشاهده می‌شود.
MaxButton	اگر مقدار این خصوصیت روی True تنظیم شود، دکمه Maximize در فرم فعال و در صورتی‌که مقدار آن روی False تنظیم شود، دکمه Maximize غیرفعال می‌شود.
MinButton	اگر مقدار این خصوصیت روی True تنظیم شود، دکمه Minimize در فرم فعال و در صورتی‌که مقدار آن روی False تنظیم شود دکمه Minimize غیرفعال می‌شود.
Icon	یک آیکن دلخواه را به عنوان آیکن منوی کنترل در فرم تعیین می‌کند.

ادامه جدول ۲۰-۵

خصوصیت	توضیح
Alignment	این خصوصیت در کنترل‌هایی مانند برچسب و کادر متن وجود دارد و تراز عبارت را در کنترل تعیین می‌کند، مقدار 0-Left Justify تراز از چپ، 1-Right Justify تراز از راست و 2-Center تراز از وسط را به وجود می‌آورند.
ToolTipText	این خصوصیت در بیشتر کنترل‌ها وجود دارد و زمانی که اشاره‌گر ماوس روی کنترل نگه داشته شود عبارت موجود در این خصوصیت نمایش داده می‌شود. از این خصوصیت برای راهنمایی کاربران استفاده می‌شود.
WordWrap	این خصوصیت مربوط به کنترل برچسب است. اگر محتویات خصوصیت Caption بیش از اندازه کنترل باشد و بخواهید عبارت موجود در خصوصیت Caption در چند خط نمایش داده شود، مقدار این خصوصیت و سپس مقدار خصوصیت AutoSize را روی True تنظیم کنید.
MultiLine	این خصوصیت مربوط به کنترل کادر متن است و در صورتی‌که مقدار آن روی True تنظیم شود، می‌توانید اطلاعات خود را در خطوط جداگانه وارد کنید. البته بهتر است که ارتفاع کنترل را نیز طوری تنظیم نمایید تا بتوان چند خط را به طور هم‌زمان مشاهده کرد.
Moveable	اگر مقدار این خصوصیت روی True تنظیم شود امکان جابه‌جا کردن فرم روی دسک‌تاپ وجود دارد، اما اگر مقدار آن روی False تنظیم شود این امکان وجود ندارد.
WindowState	این خصوصیت مربوط به فرم‌ها بوده و سه مقدار مختلف را کسب می‌کند. در صورتی‌که مقدار آن روی 0-Normal تنظیم شود موقعیت فرم روی دسک‌تاپ به وسیله خصوصیات Top و Left تنظیم می‌شود و اگر مقدار آن روی 1-Minimized تنظیم شود پنجره برنامه در زمان اجرا با اندازه حداقل و در صورتی‌که مقدار آن روی 2-Maximized تنظیم شود، پنجره برنامه با اندازه حداکثر نمایش داده خواهد شد.



Learn in English

If...Then...Else

Use an If...Then...Else block to define several blocks of statements, one of which will execute

If *condition1* **Then**

[*statementblock*]

[Else

[*statementblock*]]

End If

Visual Basic first tests *condition*1. If it's False, Visual Basic executes the corresponding statement Else statement block

For example, your application could perform different actions with depending on the value of grade variable:

```
If  grade < 10 Then
  result ="failed"
Else
 result ="passed"
End If
```

واژه‌نامه

Border	کادر، حاشیه
Compare	مقایسه کردن
Condition	شرط
CorresPonding	متناظر، منتسب
Decision	تصمیم
Execute	اجرا کردن
Image	تصویر
Input Box	کادر ورود داده
Logical	منطقی
Message Box	کادر پیام
Operation	عملگر
Perform	انجام دادن
Structure	ساختار
Style	حالت

خلاصه مطالب

• برای کنترل روند اجرای برنامه در شرایط مختلف از دستورات شرطی مانند If استفاده می‌شود.

شکل کلی دستور If به این صورت است:

```
If  (شرط(ها))  Then
        دستورات
  Else
        دستورات
End  If
```

• برای اضافه کردن یک فرم جدید به پروژه می‌توانید از گزینه Add Form در منوی Project، کلیک راست در پنجره پروژه و انتخاب گزینه Add Form (یا از دکمه Add Form در نوار ابزار استاندارد) استفاده کنید.

• برای مقایسه مقادیر رشته‌ای، عددی و سایر انواع داده‌ها از عملگرهای مقایسه‌ای استفاده می‌شود.

• متد Show می‌تواند یک فرم را در حافظه بارگذاری کرده و نمایش دهد.

• متد Hide می‌تواند یک فرم را بدون آن که از حافظه خارج نماید، مخفی کند.

• برای ایجاد ترکیب چند شرط با یکدیگر از عملگرهای منطقی And، Or و Not استفاده می‌شود.

• با استفاده از دستور Unload می‌توان یک فرم را از حافظه خارج کرد.

• ترکیب شرطی که با استفاده از عملگر And ایجاد می‌شود، وقتی نتیجه درست (True) خواهد داشت که تمام مقایسه‌ها به طور هم‌زمان درست باشند و در سایر حالت‌ها نتیجه نادرست خواهد داشت.

• ترکیب شرطی که با استفاده از عملگر Or ایجاد می‌شود، وقتی نتیجه نادرست (False) خواهد داشت که تمام مقایسه‌ها نادرست باشند و در سایر حالت‌ها نتیجه درست خواهد داشت.

• از کنترل کادر تصویر و کنترل تصویر می‌توان برای نمایش فایل‌های گرافیکی یا تصویر استفاده کرد.

• عملگر Not نتیجه هر عبارت مقایسه‌ای را که درست باشد به نادرست و اگر نادرست باشد به درست تبدیل می‌کند.

• از کادرهای پیام، برای نمایش پیام‌های مناسب در مواقع نیاز و ایجاد حالت‌های مختلف انتخاب برای کاربر استفاده می‌شود.

• از کادرهای ورود داده برای دریافت داده‌ها از کاربر استفاده می‌شود.

• کادرهای پیام با استفاده از تابع MsgBox و کادرهای ورود داده با استفاده از تابع InputBox ایجاد می‌شوند.

• تابع RTrim کاراکترهای فاصله در انتهای یک عبارت متنی (رشته‌ای) را حذف می‌کند.

• تابع LTrim کاراکترهای فاصله در ابتدای یک عبارت متنی (رشته‌ای) را حذف می‌کند.

• تابع Trim کاراکترهای فاصله در ابتدا و انتهای یک عبارت متنی (رشته‌ای) را حذف می‌کند.

• با استفاده از تابع Load Picture می‌توان یک فایل گرافیکی یا تصویر را در کنترل کادر تصویر یا کنترل تصویر نمایش داد.

آزمون نظری

۱ – کدام خصوصیت در کادر متن حداکثر تعداد کاراکترهای ورودی را تعیین می‌کند؟

الف– BorderStyle ب– PasswordChar ج– MaxLength د– TabIndex

۲ –کدام عملگر منطقی از اولویت پایین‌تری نسبت به سایر عملگرهای منطقی برخوردار است؟

الف– And ب– Or ج– Not د– And و Or

۳ – کدام تابع برای ایجاد یک کادر ورود داده مناسب است؟

الف– MsgBox ب– InputBox ج– InBox د– msg

٤ – در صورتی‌که نوع و تعداد دکمه‌ها در کادر پیام تعیین نشود، به‌طور پیش‌فرض از چه دکمه‌هایی استفاده می‌شود؟

الف– OK ب– OK و Cancel ج– Yes و No د– Cancel

۵ – کدام گزینه در رابطه با خصوصیت TabIndex درست است؟

الف– ترتیب نمایش کنترل‌ها را روی فرم تعیین می‌کند.

ب– ترتیب حرکت بین کنترل‌ها را روی یک فرم معین می‌کند.

ج– ترتیب بارگذاری کنترل‌ها را در حافظه تعیین می‌کند.

د– وضعیت کلید Tab را در صفحه کلید بررسی می‌کند.

٦ – حاصل عبارت 5*3 < >12- چیست؟

الف– True ب– False ج– ۲ د– ۲

۷ – کدام خصوصیت مشترک بین فرم و کنترل‌ها وجود دارد که توانایی مخفی کردن یا نمایش آن‌ها را دارد؟

الف– Caption ب– Enabled ج– Visible د– Appearance

۸ – حاصل ترکیب شرطی زیر چیست؟

(با توجه به عدم استفاده از دستور Option Compare Text)

```
"Cpu" = "Cpu"  Or  50 < = -70  And Not True
```

الف– True

ب– False

ج– استفاده از عملگر Not و And در کنار هم اشتباه است.

د– استفاده از عملگر مقادیر منطقی، عددی و رشته‌ای در یک ترکیب اشتباه است.

۹–درصورت عدم انتخاب عنوان یک کادر پیام به‌طور پیش‌فرض از.....استفاده می‌شود.

۱۰–الف– نام فرم ب– نام پروژه ج– نام فایل فرم د– نام فایل پروژه

با استفاده از کدام خصوصیت در کادر متن می‌توان از ویرایش داده‌ها توسط کاربر جلوگیری کرد؟

الف– Wordwrap ب– Lock ج– Locked د– MultiLine

۱۱ –In the If statement while condition is false, Visual Basic executes the statements that exist within the ..... part.

a– then b– else c– if else d– end if

۱۲– عملگرهای مقایسه‌ای را بیان کنید سپس کاربرد هر یک را توضیح دهید.

۱۳– دستور شرطی If را توضیح دهید و حالت‌های مختلف نحوه استفاده از آن‌ها را بیان کنید.

۱۴– اصطلاحات زیر را توضیح دهید:

الف– متد Show

ب– رویداد Unload

ج– متد Hide

د– رویداد Change

۱۵– تفاوت بین کادر پیغام و کادر ورود داده را بیان کنید.

۱۶– عملگرهای منطقی را نام ببرید و کاربرد هر یک را بیان کنید.

آزمون عملی

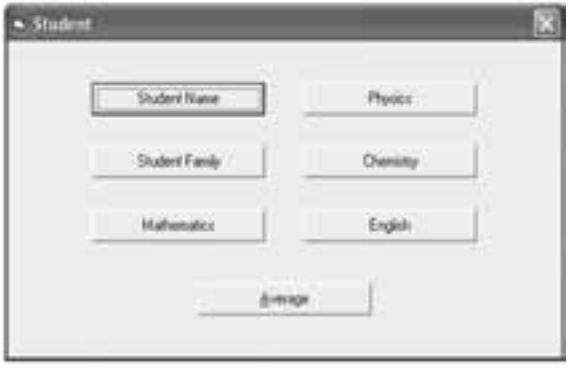
۱ – حاصل عبارت‌های زیر به ازای مقادیر a = ۱۵ ، b = ۲۰ و c =۱۰۰ چه خواهد بود؟

الف– a >۴۰ AND b > =۱۰ OR c < ۰

ب– b< =۱۰ OR c <>۱۰۰

ج– a < =۳۵ OR c > ۱۰۰ AND b > =۲۰

۲ – پروژه‌ای طراحی کنید که مطابق شکل ۱ نام، نام خانوادگی و نمره ۴ درس یک دانش‌آموز را دریافت کند، سپس معدل وی را محاسبه نماید. در صورتی‌که معدل وی کوچک‌تر از ۱۲ باشد یک کادر پیام مطابق شکل ۲ و در صورتی‌که معدل نمره وی بزرگ‌تر یا مساوی ۱۲ باشد کادر پیامی مطابق شکل ۳ نمایش داده شود، به علاوه برای دریافت داده از کادر ورود داده استفاده شود. لازم به ذکر است که کلیک روی دکمه‌های فرمان، کادر ورود داده‌ای را در اختیار کاربر قرار می‌دهد.



شکل ۱

شکل۲

شکل ۳

واحد کارششم

هدف جزیی

توانایی استفاده از انواع حلقه‌ها و ساختار Case Select و کنترل دکمه انتخاب

هدف‌های رفتاری

پس از مطالعه این واحد کار از فراگیر انتظار می‌رود که:

۱- بتواند از دستور Select Case استفاده کند.

۲- بتواند از کنترل دکمه انتخاب استفاده کرده، رویدادها و ویژگی‌های آن را به کاربرد.

۳-نحوه کار با حلقه‌های For...Next، While...Wend، Loop...While و Do While و غیره را توضیح دهد.

۴- بتواند از دستورات خروج از حلقه Exit Do و Exit For استفاده کند.

کلیات

یکی از مسایلی که در پروژه‌های واقعی با آن برخورد خواهید کرد تکراری بودن بعضی از دستورات و عملیات است. این دستورات تکراری ممکن است به تعداد دفعات معین تا رسیدن به شرایط خاصی انجام شوند. ساختارهای تکرار یا به عبارت دیگر حلقه‌ها، در این زمینه به شما کمک می‌کنند تا این‌گونه عملیات را راحت‌تر و مناسب‌تر انجام دهید.

به علاوه معمولاً هنگام طراحی برنامه‌ها لازم است تا روند اجرای دستورات را تعیین کنید. ساختارهای تصمیم، امکانات اجرایی دستورات را با توجه به شرایط مورد نظر فراهم می‌کنند. از این گروه نحوه استفاده از دستورIf را فراگرفته‌اید. نوع دیگری از ساختارهای تصمیم نیز به نام Select Case وجود دارند که در این واحدکار به توضیح آن می‌پردازیم.

۱–۶ دستور Select Case

این دستور مانند دستور If به برنامه‌نویس اجازه می‌دهد تا در شرایط مختلف دستورات مورد نظر خود را اجرا کند و عملیات مناسب را با داده‌ها و اطلاعات موجود انجام دهد. استفاده از دستور If زمانی که نیاز به بررسی شرط‌ها و انجام مقایسه‌های متعدد باشد سبب شلوغ شدن برنامه می‌شود. دستورSelect Case زمانی استفاده می‌شود که یک متغیر یا حاصل یک عبارت، دارای مقادیر مختلفی باشد و بخواهیم براساس مقدار خاصی عملیات ویژه‌ای انجام دهیم. به عبارت دیگر Select Case جایگزین Ifهایی خواهد بود که می‌توانند مقادیر مختلف یک متغیر یا عبارت را بررسی کنند. در صورت به کارگیری دستور Select Case نوشتن کدها آسان‌تر و برنامه از خوانایی و دقت بیشتری برخوردار می‌شود. شکل کلی این دستور به این صورت است:

```
Select Case    عبارت مورد مقایسه
    Case    مقدار اول      دستور(ات)
    Case    مقدار دوم      دستور(ات)
    Case    مقدار سوم      دستور(ات)
    Case Else              دستور(ات)
End Select                 دستور(ات)
```

در این دستور ابتدا مقدار عبارت مورد مقایسه با مقادیری که در مقابل هر Case قرار داده شده است، مقایسه می‌شود. اگر مقدار ذخیره شده در عبارت مورد نظر با مقدار ذکر شده در مقابل اولین Case برابر باشد، دستورات موجود در این Case اجرا می‌شوند و از بررسی سایر Caseها صرف‌نظر خواهد شد؛ اما اگر اولین مقایسه، نتیجه نادرست در پی داشته باشد، Case دوم بررسی می‌شود و در صورت درست بودن نتیجه دستورات، این Case اجرا می‌شود و در غیر این صورت Case بعدی بررسی می‌شود و به همین شکل تمام Caseها به ترتیب بررسی می‌شوند و اگر نتیجه بررسی تمام آن‌ها نادرست باشد، دستورات موجود در بخش Case Else اجرا خواهند شد و سپس برنامه ادامه می‌یابد.

نکته استفاده از بخش Case Else اختیاری است.

برای ایجاد یک بازه از مقادیر عددی در Caseها، می‌توانید از کلمه کلیدی To استفاده کنید. به عنوان مثال برای مقادیر عددی بین ۱۲ تا ۱۴ می‌توانید از Case 12 To 14 استفاده کنید.

برای استفاده از چند مقدار مختلف در یک Case می‌توانید از کاراکتر کاما (,)، کلمه کلیدی To، Is یا ترکیبی از آن‌ها استفاده کنید.

مثال ۱: پروژه‌ای مطابق شکل ۱-۶ و جداول ۱-۶ و ۲-۶ طراحی کنید که سه نمره یک دانش‌آموز را دریافت کند و رتبه وی را مشخص کند. اگر معدل وی بین ۱۸ تا ۲۰ باشد، رتبه A و اگر بین ۱۶ الی ۱۸ باشد رتبه B و بین ۱۴ الی ۱۶ رتبه C و کوچک‌تر از ۱۴ رتبه D برای او نمایش داده شود. برای این کار عملیات زیر را به ترتیب انجام دهید:

۱ – یک پروژه جدید به همراه یک فرم و کنترل‌های آن مطابق شکل ۱-۶ ایجاد کنید.

جدول ۱-۶ خصوصیات فرم

خصوصیت	مقدار
Name	frmave
Caption	Average

شکل ۱-۶

جدول ۲-۶ خصوصیات کنترل‌ها

کنترل / خصوصیت	Label	Label	Label	TextBox	TextBox	TextBox	Command Button
Name	lblit	lblwin	lblvb	txtit	txtwin	txtvb	cmdave
Caption	IT :	WINDOWS :	VB6:	—	—	—	&Average

۲ – سپس رویداد Click را به این صورت تنظیم کنید:

```
Private Sub cmdave_Click()
    Dim sngit  As Single
    Dim sngwin As Single
    Dim sngvb  As Single
    Dim sngav As Single ,strrank As String*1
    sngit = Val(txtit.Text)
    sngwin = Val(txtwin.Text)
    sngvb = Val(txtvb.Text)
    sngav = (sngit+sngwin + sngvb)/3
    Select Case sngav
        Case 18 To 20
        strrank = "A"
        Case 16 To 18
        strrank = "B"
        Case 14 To 16
        strrank = "C"
        Case Is < 14
        strrank = "D"
End Select
  MsgBox «RANK IS : «+ strrank,,»RANKING»
End Sub
```

در ایـن رویـداد پس از تعریف متغیرهـا ابتدا مقادیر تایپ شـده در کادرهای متن با اسـتفاده از تابع Val به عدد تبدیل شـده و در متغیرهای مربوطه ذخیره می‌شـوند سپس متوسط سه نمره محاسبه شده و در ادامه با استفاده از یک دستور Select Case مقدار معدل (sngav) مورد ارزیابی قرار می‌گیرد.

اگر مقدار آن بین ۱۸ تا ۲۰ باشد دستور موجود در شرط Case 18 To 20 را اجرا می‌شود و رتبه A را برای وی در متغیر رشـته‌ای strrank ذخیره می‌کند، سـپس بدون آن که سایر شـرط‌ها بررسی شوند دسـتور Select Case خاتمه می‌یابد؛ اما اگر مقدار متغیر sngav بین ۱۶ تا ۱۸ باشـد شـرط موجود در اولین Case نادرست خواهد بود و شرط Case دوم یعنی Case 16 To 18 بررسـی می‌شـود و در نتیجه دسـتور مربوط به این Case اجرا می‌شـود و بدون بررسی سایر Caseها دستور Select Case خاتمه می‌یابد و به همین شکل در صورت نادرست بودن شرط در Case دوم، سایر Caseها یکی یکی بررسی می‌شوند و در Case آخر اگر مقدار متغیر sngav کوچک‌تر از ۱۴ باشـد شرط Case Is<14 درست خواهد بود و رتبه مناسب برای وی در نظر گرفته می‌شود و پس از خاتمه دستور Select Case رتبه دانش‌آموز به وسیله یک کادر پیام نمایش داده می‌شود.

۳ – پروژه و فرم را با نام Average ذخیره کنید سپس آن‌را اجرا نمایید.

۴ – اعـداد ۱۷، ۱۹ و ۲۰ را در کادرهـای متن موجود تایپ کنید، سـپس روی دکمه Average کلیک کرده و نتیجه را بررسی کنید.

۵ – مجـدداً اعـداد ۱۴، ۵ و ۸ را در کادرهای متن موجـود تایپ کرده و روی دکمه Average کلیک و نتیجه را بررسی کنید.

۶ – اجرای برنامه را متوقف کرده و به پنجره ویژوال‌بیسیک بازگردید.

مثال ۲: پروژه‌ای طراحی کنید که هزینه حمل کالا به وسـیله وسـایل نقلیه مختلف را محاسـبه کند. به این منظور کاربر وزن کالا و مسـافت حمل را به همراه روش حمل آن معین می‌کند و سـپس هزینه حمل کالا محاسـبه شـده، در اختیار وی قرار می‌گیرد. روش‌های حمل کالا و هزینه هر یک به ازای هر کیلومتر جابه‌جایی در جدول ۳–۶ ارایه شده است و برای محاسبه هزینه حمل کالا از فرمول زیر استفاده می‌شود.

هزینه به ازای یک کیلومتر جابه‌جایی مسافت جابه‌جایی وزن کالا = هزینه حمل کالا

جدول ۳-۶

هزینه حمل به ریال	روش حمل کالا
۱۰۰۰۰	اتوبوس
۱۵۰۰۰	قطار
۱۴۰۰۰	کشتی
۲۰۰۰۰	هواپیما

شکل ۲-۶

اکنون این عملیات را به ترتیب انجام دهید:

۱ – یک پروژه از نوع Standard EXE به همراه یک فرم، دو کادر متن و برچسب و یک دکمه فرمان مطابق شکل ۲-۶ ایجاد کنید.

کادر متن اول برای ورود وزن کالا و کادر متن دوم برای ورود مسافت حمل کالا استفاده می‌شود.

۲ – برای آن‌که کاربر توانایی انتخاب نوع روش حمل کالا را داشته باشد، از کنترل دکمه انتخاب (OptionButton) استفاده کنید. این کنترل اجازه می‌دهد تا امکان انتخاب یک گزینه از بین چند انتخاب برای کاربران فراهم شود. این کنترل معمولاً به صورت گروه‌هایی دوتایی یا بیشتر استفاده می‌شود و هر لحظه می‌توان یکی از کنترل‌ها را از مجموعه آن‌ها انتخاب کرد. برای استفاده از این کنترل در جعبه ابزار روی کنترل دکمه انتخاب کلیک کنید و چهار کنترل از آن را مطابق شکل ۲-۶ روی فرم قرار دهید و خصوصیات آن‌ها را تنظیم کنید.

۳ – در پنجره خصوصیات، optbus را انتخاب کنید و خصوصیت Value آن‌را روی مقدار True تنظیم کنید. کنترل‌های دکمه انتخاب دارای خصوصیتی به نام Value هستند که یک خصوصیت منطقی است و می‌تواند True یا False باشد. اگر مقدار این خصوصیت روی True تنظیم شود نشان دهنده این است که دکمه انتخاب مربوطه انتخاب شده است و در غیر این صورت نشان‌دهنده عدم انتخاب کنترل است. همواره در یک گروه از دکمه‌های انتخاب، خصوصیت Value یکی از کنترل‌های دکمه انتخاب، True و برای سایر کنترل‌ها False می‌باشد.

۴ –اکنون باید دستورات مورد نظر را در رویداد Click دکمه فرمان Compute بنویسید.

بنابراین رویداد مزبور را به این صورت تنظیم کنید:

```
Private Sub cmdcom_Click()
   Dim inttrans As Integer, sngprice As Currency
   Dim sngw As Single, sngl As Single
   If  optbus.Value = True Then inttrans=0
   If  opttrain.Value = True Then inttrans =1
   If  optship.Value = True Then inttrans =2
   If  optplane.Value = True Then inttrans =3
   sngw = Val(txtw.Text)
   sngl = Val(txtl.Text)
Select Case inttrans
     Case0
        sngprice =sngw * sngl *10000
     Case  1
        sngprice=sngw*sngl*15000
    Case  2
        sngprice = sngw * sngl *14000
    Case  3
       sngprice = sngw * sngl *20000
   End Select
   MsgBox "Total Price =" + Str(sngprice)+ "Rials", , " Payment"
End Sub
```

در این رویداد با استفاده از دستور If مقدارخصوصیت Value هر یک از کنترل‌های دکمه انتخاب بررسی می‌شود و درصورت انتخاب هر یک از آن‌ها متغیر inttrans، متناسب با روش حمل انتخاب شده مقداردهی می‌شود؛ بنابراین در صورت انتخاب روش حمل با اتوبوس مقدار صفر، حمل با قطار مقدار ۱، حمل با کشتی مقدار ۲ و حمل با هواپیما مقدار ۳ در آن ذخیره می‌شود. سپس با استفاده از دستور Select Case مقدار متغیر inttrans بررسی می‌شود تا با توجه به روش حمل انتخاب شده، هزینه حمل کالا محاسبه و در متغیر sngprice ذخیره شود، در نهایت این مقدار به وسیله یک کادر پیام نمایش داده خواهد شد.

۵ – برای آن که ترتیب دریافت فوکوس در کنترل به صورت مناسب انجام شود، خصوصیت TabIndex دکمه فرمان Compute را روی عدد ۶ تنظیم کنید. در این صورت

دکمه فرمان Compute آخرین کنترلی است که در زمان فشردن کلید Tab، فوکوس را به دست می‌آورد.

۶ – پروژه و فرم را با نام transport ذخیره کنید.

۷ – پروژه را اجرا کنید و وزن کالا را ۲۰۰ کیلوگرم و مسافت جابه‌جایی را ۱۰۰ کیلومتر تایپ کنید، سپس روی دکمه انتخاب Plane کلیک کنید تا روش حمل کالا نیز انتخاب شود.

۸ – روی دکمه Compute کلیک کنید و نتیجه محاسبه را بررسی کنید.

۹ – مجدداً مقادیر ۳۵۰و ۲۰۰ را برای وزن کالا و مسافت جابه‌جایی تایپ کنید و این بار روش حمل با قطار را انتخاب کرده، روی دکمه Compute کلیک کنید و نتیجه را بررسی نمایید.

۱۰ – اجرای برنامه را خاتمه داده و به پنجره ویژوال‌بیسیک بازگردید.

نکته: اگر بخواهید از کنترل‌های دکمه انتخاب به گونه‌ای استفاده کنید که به صورت گروه‌های جداگانه عمل کنند باید از کنترل دیگری به‌نام کنترل قاب یا Frame استفاده کنید، در غیر این صورت تمام کنترل‌های دکمه انتخاب که در یک فرم قرار دارند به صورت یک گروه واحد در نظر گرفته خواهند شد. برای ایجاد گروه‌های مستقل از کنترل‌های دکمه انتخاب، باید ابتدا کنترل قاب را روی فرم قرار دهید و سپس با روش درگ، کنترل‌های دکمه انتخاب را روی کنترل قاب بگذارید. کنترل قاب، رویداد و خصوصیت ویژه‌ای ندارد و مهم‌ترین خصوصیت آن خصوصیت Caption است که عنوان کنترل قاب را تعیین می‌کند.

۲–۶ ساختارهای تکرار در ویژوال‌بیسیک

بعضی اوقات لازم است تا عملیاتی را به صورت تکراری انجام دهید. در ویژوال‌بیسیک دستورات متعددی برای انجام عملیات تکراری قرار داده شده‌اند که بعضی از آن‌ها توانایی اجرای عملیات تکراری را با تعداد دفعات معین دارند و بعضی دیگر عملیات موردنظر را تا رسیدن به شرایط خاصی فراهم می‌کنند. در این واحد کار به توضیح بعضی از آن‌ها می‌پردازیم.

۶-۲-۱ حلقه For ... Next

از این حلقه زمانی استفاده می‌شود که لازم باشد دستورات را به تعداد دفعات معین و محدودی اجرا کنید. نحوه استفاده از این حلقه به این صورت است:

```
For    شمارنده حلقه = مقدار شروع  To  مقدار خاتمه
        دستورات
Next   شمارنده حلقه
```

برای استفاده از این گونه حلقه‌ها از یک متغیر به عنوان شمارنده حلقه استفاده می‌شود که به وسیله آن تعداد دفعات تکرار حلقه کنترل می‌شود. مقادیر شروع و خاتمه با توجه به نیاز برنامه‌نویس تنظیم می‌شود و معمولاً از مقادیر عددی صحیح استفاده می‌شود. اما می‌توان از هر نوع متغیر عددی یا عبارات ریاضی نیز استفاده کرد. این حلقه با کلمه کلیدی For آغاز شده و با کلمه کلیدی Next که به همراه نام متغیر شمارنده ذکر می‌شود، خاتمه می‌یابد و دستورات مورد نظر برای اجرا در بین آن‌ها قرار می‌گیرند. زمانی که اجرای برنامه به اولین خط از حلقه یعنی For می‌رسد، مقدار شروع در متغیر شمارنده ذخیره می‌شود و سپس این مقدار با مقدار خاتمه مقایسه می‌شود و در صورتی که کوچک‌تر یا مساوی با مقدار خاتمه باشد، دستورات بین For و Next اجرا می‌شود. با رسیدن به کلمه کلیدی Next، اجرای برنامه مجدداً به بخش For منتقل می‌شود، سپس یک واحد به مقدار قبلی متغیر شمارنده اضافه شده و در صورتی که این مقدار کوچک‌تر یا مساوی با مقدار خاتمه باشد، دستورات موجود در حلقه اجرا می‌شود و این عملیات تا زمانی که مقدار متغیر شمارنده از مقدار خاتمه بزرگ‌تر شود، ادامه می‌یابد. پس از خاتمه اجرای حلقه، اجرای دستورات بعد از کلمه کلیدی Next انجام می‌شود.

مثال ۳: پروژه‌ای طراحی کنید که مجموع اعداد ۱ تا ۱۰ را محاسبه کرده و روی فرم نمایش دهد. به این منظور عملیات زیر را به ترتیب انجام دهید:

۱ – برنامه ویژوال‌بیسیک را اجرا کنید سپس یک پروژه از نوع Standard EXE به همراه یک فرم و یک کنترل دکمه فرمان و یک کنترل برچسب مطابق شکل ۳-۶ ایجاد کنید.

شکل ۳-۶

۲ – رویداد Click دکمه فرمان cmdshow را به صورت زیر تنظیم کنید:

```
Private Sub cmdshow_Click()
    Dim inti As Integer
    Dim intsum As Integer
    For inti = 1 To 10
        intsum =intsum + inti
    Next inti
    lblsum. Caption = lblsum. Caption + Str (intsum)
End Sub
```

در این رویداد ابتدا متغیرهای مورد نیاز تعریف می‌شـوند. به منظور محاسـبه مجموع اعداد ۱ تا ۱۰ از یک حلقه For استفاده شده است. این حلقه از عدد ۱ شروع و تا رسیدن حلقه به مقدار ۱۰ تکرار می‌شـود و در هر مرحله از اجرای حلقه نیز با اسـتفاده از متغیر intsum مجموع اعداد محاسبه می‌شود و با پایان یافتن حلقه مقدار مجموع intsum پس از تبدیل به نوع داده رشته‌ای در کنترل برچسب نمایش داده می‌شود.

۳ – پروژه و فرم را با نام sumnumbers ذخیره کنید سپس آن‌را اجرا کرده و آزمایش نمایید.

۴ – به اجرای پروژه خاتمه داده و به پنجره ویژوال‌بیسیک بازگردید.

مثـال ۴: پـروژه‌ای طراحی کنید که یک عدد طبیعـی را از ورودی دریافت کند و مجموع اعداد زوج کوچک‌تر یا مساوی آن را نمایش دهد.

۱ – برنامـه ویژوال‌بیسـیک را اجرا کرده، یـک پروژه به همراه یک فرم، سـه کنترل برچسـب و یک کنتـرل کادر متن به همراه یک کنترل دکمه فرمان مطابق شـکل ۴–۶ و جدول ۴–۶ ایجاد کنید.

جدول ۴-۶ خصوصیات فرم

خصوصیت	مقدار
Name	frmsn
Caption	Show Number

شکل ۴–۶

۲ – سپس رویداد Click دکمه Show Number را به این صورت تنظیم کنید:

```
Private Sub cmdshow_Click()
    Dim inti As Integer
    Dim intno As Integer
    Dim lngsum As Long
    intno = Val(txtno.Text)
    For inti = 1 To intno
        If intno mod inti=0
          lngsum=lngsum+inti
            End If
    Next inti
    lblsum.Caption=lblsum.Caption+Str(lngsum)
    End Sub
```

۳ – پروژه و فرم را با نام shownumber ذخیره کنید سپس آن را اجرا کرده و آزمایش کنید.

۴ – اجرای پروژه را متوقف کرده و به پنجره ویژوال‌بیسیک باز گردید.

اکنون شکل کلی حلقه For به شما معرفی می‌شود. همان‌طور که گفته شد متغیر شمارنده حلقه در هر بار اجرای حلقه به میزان یک واحد افزایش می‌یابد، اما گاهی اوقات لازم است تا میزان افزایش شمارنده به وسیله برنامه‌نویس تعیین شود یا از حلقه‌های کاهشی استفاده شود که در آن‌ها مقدار شروع از مقدار خاتمه بزرگ‌تر است و در هر بار اجرای حلقه مقدار شمارنده کم می‌شود. شکل کلی دستور For به این صورت است:

```
For شمارنده حلقه = مقدار شروع To مقدار خاتمه Step میزان افزایش یا کاهش شمارنده
        دستورات
Next شمارنده حلقه
```

در واقع بخش Step امکان تنظیم میزان افزایش یا کاهش مقدار شمارنده حلقه را فراهم می‌کند. مقدار پیش فرض برای این بخش مقدار یک است. می‌توانید برای حلقه‌های افزایشی از مقادیر مثبت و برای حلقه‌های کاهشی از مقادیر منفی استفاده کنید. در این صورت حلقه زمانی متوقف می‌شود که مقدار شروع از مقدار خاتمه کوچک‌تر شود.

نکته درصورتی‌که مقدار شروع از مقدار خاتمه بزرگ‌تر باشد و مقدار Step مثبت باشد، حلقه اجرا نخواهد شد.

در صورت عدم استفاده از بخش Step مقدار افزایش شمارنده حلقه یک واحد خواهد بود

دستور Print

دستور Print می‌تواند انواع مقادیر ثابت، متغیر و مقدار خصوصیت اشیا را روی فرم نمایش دهد. در صورتی‌که از این دستور به تنهایی استفاده شود فقط یک خط خالی نمایش داده می‌شود و اگر از کاراکتر سمی‌کالن (;) استفاده شود، خروجی‌ها پشت سرهم و بدون فاصله قرار می‌گیرند و اگر از کاراکتر کاما (,) استفاده شود، هر خط نمایشی در روی فرم، توانایی نمایش ۱۰ کاراکتر را خواهد داشت و اگر خروجی بزرگ‌تر از ۱۰ کاراکتر باشد، خروجی بعدی در یک ناحیه جلوتر نمایش داده خواهد شد. به عنوان مثال به این دستورات توجه کنید:

```
Print  "Ali" ; "Reza"
Print  "Visual", "Basic"
Print  "Student" ;
Print  "Information"
```

در صورت اجرای دستورات فوق خروجی حاصل به این شکل خواهد بود:

```
AliReza
Visual      Basic
StudentInformation
```

عبارات موجود در دستور اول به دلیل استفاده از کاراکتر سمی‌کالن پشت سرهم و عبارات موجود در دستور دوم به دلیل استفاده از کاراکتر کاما با فاصله از هم قرار گرفته‌اند و در دو دستور آخر نیز چون Print سوم به کاراکتر سمی‌کالن ختم می‌شود، خروجی Print چهارم پشت سر آن نمایش داده می‌شود.

به عنوان مثال به این دستورات توجه کنید و نتیجه اجرای آن‌ها را در شکل ۵-۶ مشاهده کنید.

```
Print  "1234567890" , "1234567890"
Print  "12345678901234567890"
Print  "12345678901" , "1234567890"
```

شکل ۶-۵

در دستور Print اول هر عبارت نمایشی از ده کاراکتر تشکیل شده است؛ بنابراین دو عبارت نمایشی به ترتیب در نواحی اول و دوم در کنار هم قرار می‌گیرند. اما در دستور Print آخر چون عبارت نمایشی اول از ۱۱ کاراکتر تشکیل شده است، عبارت نمایشی دوم در ناحیه ۳ نمایش داده می‌شود. دستور Print دوم برای تشخیص بهتر اندازه نواحی استفاده شده است.

تمرین:

پروژه‌ای را طراحی‌کنید تا مطابق شکل ۶-۶ توانایی نمایش اعداد فرد و زوج کوچک‌تر از هرعدد طبیعی دلخواه موردنظر کاربر را داشته باشد. برای دریافت داده‌ها از کادرورود داده استفاده کنید.

شکل ۶-۶

حلقه‌های For را می‌توان در داخل یکدیگر قرار داد و در این حالت با هر بار اجرای حلقه For اول، حلقه For داخل آن یک دور کامل اجرا می‌شود. شکل کلی نحوه استفاده از حلقه‌های متداخل به‌صورت زیر است:

```
For شمارنده حلقه = مقدار شروع To مقدار خاتمه Step میزان افزایش یا کاهش شمارنده
    For شمارنده حلقه = مقدار شروع To مقدار خاتمه Step میزان افزایش یا کاهش شمارنده
        دستور(ات)
    Next شمارنده حلقه
Next شمارنده حلقه
```

مثال ۵: پروژه‌ای طراحی کنید که با استفاده از حلقه For کاراکترهای ستاره را مطابق شکل ۷–۶ نمایش دهد. برای این کار عملیات بعد را انجام دهید.

شکل ۷–۶

۱ – یک پروژه از نوع Standard EXE به همراه یک فرم و یک کنترل دکمه فرمان مطابق شکل ۷–۶ ایجاد کنید.

۲ – رویداد Click دکمه فرمان را به این صورت تنظیم کنید:

```
Private Sub cmdsshow_Click()
    Dim inti As Integer
    Dim intj As Integer
    For inti = 1 To 4
        For intj= 1 To 5
            Print "*";
        Next intj
        Print
    Next inti
End Sub
```

۳ – پروژه و فرم را با نام showstart ذخیره کنید سپس برنامه را اجرا کرده و روی دکمه show کلیک کنید.

۴ – به اجرای برنامه خاتمه داده و به پنجره ویژوال‌بیسیک بازگردید.

تمرین:

پروژه‌ای طراحی کنید که شکل زیر را رسم نماید.

```
*
**
***
****
*****
```

۶-۲-۲ حلقه While ... Wend و Do While ... Loop

این حلقه‌ها، دستورات را تا زمانی که شرط یا شرط‌های تعیین شده درست باشند، اجرا می‌کنند. از این نوع حلقه‌ها زمانی استفاده می‌شود که دفعات تکرار دستورات معین نباشد. شکل کلی این دستورات به صورت زیر است:

Do While (شرط یا شرط‌ها)

دستور(ات)

Loop

While (شرط یا شرط‌ها)

دستور(ات)

Wend

زمانی که اجرای برنامه به حلقه While برسد، ابتدا شرط یا شرط‌های موجود در جلوی کلمه کلیدی While ارزیابی می‌شوند و اگر نتیجه این ارزیابی درست باشد، دستورات بین While و Wend اجرا می‌شوند. با رسیدن به انتهای حلقه یعنی کلمه کلیدی Wend اجرای برنامه مجدداً به بخش While منتقل می‌شود و بررسی شرط یا شرط‌ها انجام می‌گیرد و به همین شکل اجرای حلقه تا زمانی که نتیجه ارزیابی شرط موجود در حلقه نادرست شود، ادامه می‌یابد. نحوه اجرای حلقه Do While مانند حلقه While است.

مثال ۶: پروژه‌ای طراحی کنید که هر بار مسافت بین دو شهر را براساس کیلومتر دریافت کند سپس مسافت دو شهر را به متر تبدیل کرده و نمایش دهد و این‌کار را تا دریافت مقدار صفر انجام دهد.

۱ – یک پروژه از نوع Standard EXE به همراه یک فرم مطابق با شکل ۸-۶ ایجاد کنید.

شکل ۸-۶

۲ – یک کنترل برچسب و دو کنترل دکمه فرمان روی فرم قرار داده و خصوصیت‌های آن‌ها را تنظیم کنید.

۳ – رویداد دکمه Compute را به این صورت تنظیم کنید:

```
Private Sub cmdcompute_Click ()
    Dim intno As Integer
    Dim strdata As String
    intno = 1
    While (intno > ۰)
       strdata = InputBox ("Enter Your Number:" , "Enter Data" , ۰)
       intno = Val (strdata)
       lblresult. Caption = intno*1000
    Wend
End Sub
```

در ایـــن رویداد پس از تعریـــف متغیرها ابتدا مقدار متغیـــر intnoروی مقدار ۱ تنظیم می‌شـــود تا حلقه While بتواند اجرا شـــود. زیرا شـــرطی که برای حلقه در نظر گرفته شده اســـت intno > ۰ می‌باشـــد اگر intno مقداردهی نشود هیچ وقت امکان ورود داده و انجام محاســـبات وجود نخواهد داشت. در مرحله بعد حلقه While اجرا می‌شود و با استفاده از یک کادر ورود مســـافت دو شـــهر دلخواه دریافت شده و در متغیر strdata ذخیره می‌شود ســـپس با اســـتفاده از تابع Val محتویات متغیر strdata به عدد تبدیل شده و در متغیر intno قرار می‌گیرد ســـپس با تبدیل این مقدار به واحد متر، مســـافت در کنترل برچسب نمایش داده می‌شود. در این مرحله با رسیدن به دستور Wend اجرای برنامه به ابتدای حلقه یعنی دســـتور While منتقل می‌شـــود در نتیجه شـــرط intno > ۰ بررسی شده و اگر نتیجه بررسی شـــرط درست باشـــد عملیات تکرار خواهد شد اما اگر نتیجه بررسی شرط نادرست باشد (در صورت ورود عدد صفر) اجرای حلقه خاتمه می‌یابد.

۴ – پروژه و فرم را با نام Distance ذخیره کنید.

۵ – پروژه را اجرا کرده و روی دکمه Compute کلیک کنید و پس از ورود چند مقدار مختلف عدد صفر را وارد کنید تا مجدداً به پنجره برنامه بازگردید.

۶ – روی دکمـــه Exit کلیک کنید و از برنامه خارج شـــده و به پنجره ویژوال‌بیســـیک بازگردید.

تمرین:
پـروژه‌ای طراحی کنید تـا مضرب‌های کوچک‌تر از ۱۰۰ عدد ۹ را روی فرم نمایش دهد. به این منظور عملیات زیر را به ترتیب انجام دهید:

۳-۲-۶ حلقه Do ... Loop While

این حلقه مشابه حلقه Do While ... Loop است؛ با این تفاوت که دستورات داخل حلقه یک بار اجرا شـده، سپس شرط حلقه بررسی می‌شود. بنابراین در صورت نادرست بودن شـرط، دسـتورات حداقل یک بار اجرا می‌شوند. شـکل کلی این دستور به صورت زیر است:

Do
⋮ **دستور(ات)**
Loop While (شرط یا شرط‌ها)

مثال ۷: پروژه‌ای طراحی کنید که یک عدد طبیعی را دریافت کند و مجموع ارقام آن را نمایش دهد. برای این کار عملیات زیر را به ترتیب انجام دهید:

۱- یک پروژه از نوع Standard EXE به همراه یک فرم با مشخصات جدول ۱۹-۷ ایجاد کنید.

۲ - در این پروژه به منظور اجرای دستورات از رویداد Activate فرم استفاده می‌شود. این رویداد بعد از رویداد Load فرم و زمانی‌که فرم نمایش داده شده و فوکوس را کسب می‌کند اجرا می‌شود. بنابراین رویداد Activate فرم را به این صورت تنظیم کنید:

```
Private Sub Form_Activate()
    Dim lngi As Long
    Dim lngno As Long
    Dim strinput As String
    strinput = InputBox("Enter Your Number :"," input DATA",0)
    lngno =Val(strinput)
    Do
        lngi = lngno Mod 10
        Sum=Sum+lngi
```

```
    lngno = lngno \ 10
  Loop While (lngno>0)
End Sub
```

۳ – اجرای پروژه را متوقف کرده و به پنجره ویژوال‌بیسیک بازگردید.

تمرین:

پروژه طراحی کنید که مضارب کوچکتر از ۱۰۰ عدد ۷ را نمایش دهد.

۶–۲–۴ دستورات خروج از حلقه Exit Do و Exit For

دستورات Exit Do و Exit For به برنامه‌نویس امکان می‌دهند که قبل از آن‌که حلقه به طور عادی خاتمه یابد به اجرای حلقه خاتمه دهد و از آن خارج شود. از دستور Exit For برای خروج از حلقه For...Next و از دستور Exit Do برای خروج از حلقه‌های Do While...Loop، Do..Loop While، Do Until...Loop و Do...Loop Until استفاده می‌شود.

مثال ۸: پروژه digit را به‌گونه‌ای تغییر دهید تا با استفاده از دستور Exit Do اجرای حلقه خاتمه یابد. به این منظور عملیات زیر را به ترتیب انجام دهید:

۱ – پروژه digit را باز کنید و رویداد Activate فرم را به این صورت تغییر دهید:

```
Private Sub Form_Activate()
  Dim lngi As Long
  Dim lngno As Long
  Dim strinput As String
  strinput = InputBox("Enter Your Number :","Input Data",0)
  lngno =Val(strinput)
  Do
     lngi = lngno Mod 10
     sum=sum+lngi
     lngno = lngno \ 10
     If   lngno < 0  Then   Exit Do
  Loop While (True)
End Sub
```

در این رویداد نیز مانند راه حل قبلی حلقه Do...Loop While به کار گرفته شـــده است. با این تفاوت که به‌جای شـــرط lngno >< ۰ از مقدار True استفاده شده است بنابراین حلقه می‌تواند بدون محدودیت بارها اجرا شـــود اما با توجه به حالت پروژه باید اجرای حلقه با توجه به تعداد ارقام عدد وارد شده تنظیم شود. به این منظور از یک دستور If دیگر در انتهای حلقه استفاده شده است که شرط lngno < ۰ را بررسی می‌کند در صورتی‌که نتیجه این بررســـی درست باشد (یعنی دیگر رقمی در عدد باقی نمانده باشد) با دستور Exit Do از اجرای مجدد حلقه جلوگیری به عمل می‌آورد.

۲ – تغییرات را ذخیره کرده و پروژه را اجرا و آزمایش کنید.

۳ – به اجرای پروژه خاتمه داده و به پنجره ویژوال‌بیسیک بازگردید.

مثال ۹: پروژه‌ای طراحی کنید که بزرگ‌ترین عدد را بین n عدد دلخواه جستجو کرده و نمایش دهد. برای این کار عملیات زیر را انجام دهید:

۱–یک پروژه جدید مطابق شکل ۹–۶ ایجاد کنید.

شکل ۹–۶

۲ – مطابق شکل ۹–۶ ابتدا کاربر تعداد اعداد خود را در کادر متن وارد می‌کند؛ سپس با کلیک روی دکمه Input و با استفاده از کادرهای ورود داده اعداد خود را وارد می‌کند و با پایان ورود اعداد، مقدار ماکزیمم روی فرم نمایش داده می‌شود. به علاوه در صورتی که کاربر کاراکتر Q را در کادر ورود داده تایپ کند، برنامه از دریافت اعداد بعدی خودداری می‌نماید. بنابراین رویداد Click دکمه Input را به این صورت تنظیم کنید:

```
Private Sub cmdinput_Click()
    Dim inti As Integer, strno As String
    Dim sngmax As Single
    For inti = 1 To Val(txtno.Text)
        strno = InputBox("Enter Your Number:", "InputBox", 0)
        If strno ="Q" or strno ="q" Then Exit For
        If inti = 1 Then
          sngmax = Val(strno)
       Else
```

```
        If sngmax < Val (strno)Then sngmax =Val(strno)
    End If
Next inti
MsgBox "Maximum Is :"+Str(sngmax)
End Sub
```

در این رویداد از یک حلقه For...Next برای دریافت اعداد استفاده خواهد شد. مقدار خاتمه حلقه با توجه به مقداری که کاربر در کادر متن مربوطه وارد می‌کند، تنظیم می‌شود. بخش عمده‌ای از دستورات موجود در حلقه برای محاسبه بزرگ‌ترین عدد به کار می‌روند، اما پس از آن‌که این رویداد اجرا شد، ابتدا با استفاده از یک کادر ورود داده مقدار اولین عدد را از کاربر دریافت می‌کند و بلافاصله مقدار تایپ شده در کادر ورود داده با یک دستور If بررسی می‌شود و اگر کاراکتر Q یا q تایپ شده باشد نتیجه شرط موجود درست بوده و در نتیجه دستور Exit For اجرا می‌شود که سبب خروج از حلقه و خاتمه دریافت اعداد خواهد شد؛ اما در غیر این صورت دستور If دوم بررسی می‌شود که در این صورت اگر اولین عدد داده شده باشد (یعنی شمارنده حلقه برابر با یک باشد) اولین عدد به عنوان بزرگ‌ترین عدد در نظر گرفته می‌شود و در متغیر sngmax ذخیره می‌شود، اما برای اعداد بعدی ابتدا مقدار تایپ شده در کادر ورود داده با مقدار ذخیره شده در متغیر sngmax مقایسه می‌شود و در صورتی که از آن بزرگ‌تر باشد، مقدار جدید به عنوان بزرگ‌ترین مقدار، در متغیر sngmax ذخیره خواهد شد. به این ترتیب بزرگ‌ترین مقدار، محاسبه و در یک کادر پیام نمایش داده می‌شود.

۳ – پروژه و فرم را با نام maximum ذخیره کنید، سپس آن‌را اجرا نمایید.

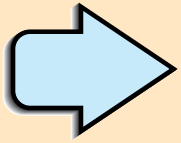
۴ – مقدار ۱۰ را در کادر متن تایپ کرده و روی دکمه Input کلیک کنید.

۵ – اعداد ۷، ۲ و ۱۸ را به ترتیب وارد کنید، سپس کاراکتر Q را در کادر ورود داده چهارم تایپ کنید و روی دکمه OK کلیک کنید. همان‌طور که مشاهده می‌کنید از ورود داده‌های بعدی جلوگیری به عمل می‌آید و بزرگ‌ترین عدد بین سه عدد وارد شده (۱۸) نمایش داده می‌شود.

۶ – پروژه را برای مقادیر دیگری آزمایش کنید، سپس اجرای پروژه را پایان داده و به پنجره ویژوال‌بیسیک بازگردید.

تمرین:

پروژه قبل را به گونه‌ای تنظیم کنید که توانایی محاسبه کوچک‌ترین عدد را نیز داشته باشد.



Learn in English

For...Next Statement

.Repeats a group of statements a specified number of times

Syntax

For counter = start **To** end [*step*]
[*statements*]
Next [*counter*]

The **For…Next** statement syntax has these parts:

Part	Description
counter	Required. Numeric variable used as a loop counter. The variable can>t be a Boolean or an array element.
start	Required. Initial value of counter.
end	Required. Final value of counter.
step	Optional. Amount counter is changed each time through the loop. If not specified, step defaults to one.
statements	Optional. One or more statements between **For** and **Next** that are executed the specified number of times.

واژه‌نامه

Compute	محاسبه کردن
Counter	شمارنده
Even	زوج
Exit	خروج
Final	نهایی، آخری
Information	اطلاعات
Initial	ابتدایی، اولین، اصلی
Odd	فرد
Optional	اختیاری
Payment	مبلغ، بهای پرداختی
Price	بها، قیمت
Rank	رتبه
Required	لازم داشتن
Specify	تعیین کردن
Statement	دستور
Step	مرحله
Transport	حمل کردن

خلاصه مطالب

• از دستور Select Case برای ارزیابی چند مقایسه به صورت یکجا استفاده می‌شود.
از کنترل دکمه انتخـــاب (Option Button) برای ایجاد گزینه‌های متعدد، برای انتخاب کاربر استفاده می‌شود.

• برای اجرای دستورات با دفعات تکرار معین از دستور Next ... For استفاده می‌شود.

• حلقه‌های While... Wend، Do While ... Loop و Do...Loop While دستورات را تا زمانی که نتیجه شرط یا شرط‌های ذکر شده در آن‌ها درست باشند، اجرا می‌کنند.

• حلقه‌های Do Until...Loop و Do...Loop Until دستورات را تا زمانی که نتیجه شرط یا شرط‌های ذکر شده در آن‌ها نادرست باشند، اجرا می‌کنند.

• دستور Print می‌تواند انواع پیام‌ها، عبارات رشته‌ای، مقدار متغیرها و خصوصیات فرم‌ها و کنترل‌ها را روی فرم نمایش دهد.

• از دستور Exit Do برای خروج از حلقه Do While...Loop، Do..Loop While، Do Until...Loop و Do...Loop Until استفاده می‌شود.

• از دستور Exit For برای خروج از حلقه For ... Next استفاده می‌شود.

• از کنترل قاب (Frame) برای دسته‌بندی کنترل‌ها استفاده می‌شود.

• برای اضافه کردن یک پروژه جدید از نوع Standard EXE می‌توان از گزینه New Project در منوی File ویژوال‌بیسیک یا از دکمه Add Standard EXE Project در نوار ابزار استاندارد استفاده کرد.

• به منظور دسترسی به خاصیت‌ها و متدهای کنترل یک فرم در فرم دیگر می‌توان به‌صورت زیر عمل نمود:

نام متد یا خاصیت.نام کنترل.نام فرم

آزمون نظری

۱ – کدام حلقه شبیه به حلقه Do While ... Loop است؟

الف– For...Next ب– While ...Wend ج– Select Case د– While...Loop

۲ – پس از اجرای برنامه زیر مقدار متغیر Sum چقدر است؟

```
Sum=0
I=5
Do While (I>3)
    For k=1  to  2
        Sum = Sum+k*I
    Next k
  I=I-1
 Loop
```

الف– ۲۷ ب– ۲۸ ج– ۲۹ د– ۳۰

۳ – حلقه زیر چند بار تکرار می‌شود؟

```
For i=7  To  -2  Step  -3
  ⋮
Next i
```

الف– ۲ ب– ۳ ج– ۴ د– ۵

۴ – حلقه زیر چند بار تکرار می‌شود؟

```
While (True)
  ⋮
Wend
```

الف– ۳۰ بار ب– ۴۰ بار ج– بی‌نهایت د– تکرار نمی‌شود.

۵ – خروجی برنامه پس از اجرای حلقه زیر چیست؟

```
i=6
While (i>4)
j=6
```

```
    i=i - 1
    Do While (j>3)
        Print i,j
    j=j - 1
    Loop
Wend
```

الف-		ب-		ج-		د-	
۶	۵	۵	۵	۵	۴	۶	۴
۵	۵	۶	۵	۶	۴	۵	۴
۴	۵	۴	۵	۴	۴	۴	۴
۶	۴	۵	۴	۵	۵	۶	۵
۵	۴	۶	۴	۶	۵	۵	۵
۴	۴	۴	۴	۴	۵	۴	۵

۶- خروجی برنامه پس از اجرای حلقه زیر چیست؟

```
For   i=10  To   1  Step  2
Print  "i"  ;i
Next  i
```

الف- اعداد زوج بین ۱ و ۱۰ ب- اعداد فرد بین ۱ و ۱۰

ج- اعداد ۱۰ تا ۱ د- هیچ‌گونه خروجی ندارد.

۷ - درصورتی‌که مقدار متغیر intx برابر ۲ باشد، خروجی دستورات زیر چیست؟

```
Select   Case    intx
Case Is > 10 :  Print    intx*10
Case Is <= 4 :  Print    intx*4
Case Is = 7 :    Print    intx*2
End  Select
```

الف- ۱۰ ب- ۸ ج- ۴ د- ۲

دستور Exit Do سبب خروج از کدام حلقه می‌شود؟

الف- For ...Next ب- While ... Wend

ج- Do While ... Loop د- گزینه‌های ب و ج صحیح هستند.

۹ – کدام حلقه مشابه حلقه زیر است؟

```
For   i=4   to 1   Step –1
  ⋮
Next   i
```

الف–

```
i=4
While (i>=1)
  ⋮
i=i+ 1
Wend
```

ب)

```
i=4
Do    While  (i>=1)
  ⋮
i = i- 1
Loop
```

ج–

```
i =4
While  (i<=1)
  ⋮
i=i - 1
Wend
```

د–

```
i = 4
Do    While   (i >=1)
  ⋮
i = i + 1
Loop
```

۱۰ – کدام گزینه در رابطه با دستورات زیر صحیح است؟

```
For    i=1   To 15
   If   i    Mod   3=○   Then  Print i
Next   i
```

الف- اعداد ۱ تا ۱۵ را نمایش می‌دهد.
ب- مضارب عدد ۳ از ۱ تا ۱۵ را نمایش می‌دهد.
ج- اعداد ۱ تا ۱۵ را که بر ۳ بخش‌پذیر هستند نمایش می‌دهد.
د- باقیمانده تقسیم اعداد ۱ تا ۱۵ را بر ۳ نمایش می‌دهد.

۱۱ - برای دسته‌بندی کنترل‌ها از کنترل ..... استفاده می‌شود.

الف- CheckBox ب- Frame ج- GroupBox د- Label

۱۲- با استفاده از کدام خصوصیت می‌توان کنترل OptionButton را در حالت انتخاب قرار داد؟

الف- Check ب- Index ج- Value د- Tab Index

۱۳ - حلقه زیر چند بار تکرار می‌شود؟

```
k=2
Do
  ⋮
k=k*2
Loop Until (k>10)
```

الف- ۵ ب- ۴ ج- ۳ د- ۲

۱۴-Inthe For Statement, the step argument can be.....number.

a- positive b- negative c- integer d- positive or negative

۱۵- تفاوت بین حلقه‌های While با حلقه‌های Until چیست؟

۱۶- کاربرد دستور Exit Do و Exit For را بیان کنید.

۱۷- نحوه عملکرد دستور For را همراه با بخش Step توضیح دهید.

۱۸- تفاوت دستور Select Case را با IF بیان کنید.

۱۹- کنترل دکمه انتخاب (Option Button) و قاب (Frame) را توضیح داده و کاربرد هر یک از آن‌ها را بیان کنید.

۲۰- نحوه اضافه کردن یک پروژه جدید به پروژه موجود را توضیح دهید.

آزمون عملی

۱ – پروژه‌ای طراحی کنید که مجموع اعداد سه رقمی را محاسبه کرده و نمایش دهد.

۲ – پروژه‌ای طراحی کنید که عدد طبیعی و دلخواهی را از کاربر دریافت کند و مقسوم علیه‌های زوج آن‌را محاسبه کرده و نمایش دهد.

۳ – پروژه‌ای طراحی کنید که یک جدول ضرب ۱۰×۱۰ را چاپ کند.

۴ – پـروژه‌ای طراحی کنید که دو عدد طبیعی را دریافت کند و کوچک‌ترین مضرب مشترک آن‌ها را نمایش دهد.

۵ – خروجی حاصل از اجرای حلقه‌های زیر چه خواهد بود؟

الف–

```
J=10
Do While  (True)
    If j>5  Then  Exit  Do
    For i=1 to j
Print i;
    Next i
    Print
    j=j+ 1
Loop
```

ب–

```
For  I= 1 to  4
    For  j=1  to  3
      Print  "*";
    Next  j
    Print
Next I
```

۶ – پروژه‌ای طراحی کنید که دو عدد را دریافت کرده و بزرگ‌ترین مقسوم‌علیه آن‌ها را نمایش دهد.

واحد کارهفتم

هدف جزیی

توانایی ایجاد و استفاده از انواع رویه‌ها در ویژوال‌بیسیک

هدف‌های رفتاری

پس از مطالعه این واحد کار از فراگیر انتظار می‌رود که:

۱- انواع رویه‌ها را در زبان برنامه‌نویسی ویژوال‌بیسیک توضیح دهد.

۲- رویه‌های رویداد را تعریف کند.

۳- رویه‌های فرعی و رویه‌های تابعی را تعریف کند و تفاوت بین آن‌ها را توضیح دهد.

۴- بتواند با استفاده از دستورات Exit Sub و Exit Function از رویه‌های فرعی و تابعی خارج شود.

۵- انواع روش‌های ارسال متغیرها به رویه‌ها و تفاوت بین آن‌ها را بیان کند.

۶- تفاوت رویه‌های عمومی و محلی را بیان کند و بتواند انواع رویه‌ها را به‌صورت عمومی و محلی تعریف کند.

معمولاً در طراحی پروژه‌های واقعی نیاز به دستورالعمل‌های فراوانی خواهید داشت و گاهی پیش می‌آید که مجموعه‌ای از دستورالعمل‌ها در بخش‌های مختلفی از پروژه یا حتی سایر پروژه‌ها به صورت تکراری استفاده می‌شوند. در این صورت می‌توان به جای تکرار, این مجموعه دستورات را یک‌بار در بخشی از پروژه نوشته و بارها مورد استفاده قرار داد. به این منظور در پروژه‌های بزرگ، برنامه به بخش‌های کوچک‌تر تقسیم می‌شود و در نتیجه زمان لازم برای خطایابی و خطازدایی و نوشتن دستورالعمل‌ها به شکل قابل توجهی کاهش می‌یابد. به علاوه ایجاد تغییرات در برنامه برای ارتقا و توسعه آن در آینده راحت‌تر و با خطای کمتر توأم خواهد بود. به این روش, برنامه‌نویسی ساخت یافته (Structural) می‌گویند. همان‌طور که می‌دانید این ویژگی در زبان ویژوال‌بیسیک نیز وجود دارد و با استفاده از انواع رویه‌ها (Procedure) می‌توانید به روش ساخت یافته نیز برنامه‌نویسی کنید. در ویژوال‌بیسیک، چهار نوع رویه وجود دارد که عبارتند از: رویه‌های فرعی (Sub Procedure), رویه‌های تابعی (Function Procedure), رویه‌های رویداد (Event Procedure) و رویه‌های آماده ویژوال‌بیسیک (Visual Basic Procedure). در این‌جا به توضیح سه گروه اول خواهیم پرداخت و در واحدکار بعد نحوه استفاده از رویه‌های آماده ویژوال‌بیسیک را فراخواهید گرفت.

۱–۷ رویه‌های فرعی (Sub Procedure)

گاهی اوقات لازم است تا برنامه‌نویس دستورات مورد نظر خود را به صورت یک بخش مستقل از سایر دستورات در پروژه قرار دهد و در هر زمان که لازم باشد در هر جایی از پروژه (مانند سایر رویه‌ها) از آن استفاده کند. در این صورت استفاده از رویه‌های فرعی می‌تواند یکی از انتخاب‌های مناسب برای وی باشد.

به عنوان مثال رویه‌های رویداد از نوع رویه‌های فرعی هستند که بارها از آن استفاده کرده‌اید. رویه‌های فرعی دیگر را نیز می‌توانید در ماژول فرم، تعریف و استفاده کنید. اما همان‌طور که می‌دانید رویه‌های رویداد به‌طور خودکار درزمان لازم اجرا می‌شوند، اما برای اجرای رویه‌های فرعی که برنامه‌نویس تعریف می‌کند باید از دستورات خاصی استفاده شود تا بتواند رویه را در زمان مناسب اجرا کند.

در این‌جا لازم اســت تا نحوه تعریف یک رویه را به طور کلی بیاموزید، برای تعریف یک رویه فرعی به این شکل عمل کنید:

Private Sub نام رویه فرعی (نام آرگومان اول As نوع آرگومان , . . .)

دستور(ات)

End Sub

تعریف یک رویه فرعی با کلمه کلیدی Private آغاز می‌شود. استفاده از این عبارت در تعریف یک رویه فرعی سبب می‌شود تا رویه مربوطه به عنوان یک رویه محلی تعریف شود؛ بنابراین فقط در سایر رویه‌های همان ماژول فرمی که در آن تعریف شده، قابل شناسایی و فراخوانی است. در ادامه تعریف رویه، کلمه کلیدی Sub و به دنبال آن نام رویه فرعی قرار می‌گیرد. برای نام‌گذاری رویه‌ها می‌توانید مانند متغیرها عمل کنید.

پــس از آن‌که نام رویــه را تعیین کردید باید نام و نوع آرگومان‌های رویه را مشــخص کنید. در واقع آرگومان‌ها متغیرهایی هستند که به برنامه‌نویس اجازه می‌دهند تا از این طریق مقادیر موردنظر را به رویه فرعی ارسال کنند. قوانین تعریف و نام‌گذاری آرگومان‌ها دقیقاً مشابه متغیرهاست.

البتــه اســتفاده از آرگومان‌ها اجباری نبوده و یک رویه فرعــی می‌تواند یک, دو یا چند آرگومان داشــته باشد یا این‌که اصلاً هیچ آرگومانی نداشته باشد. اما در صورت استفاده از آن‌ها باید در هنگام فراخوانی رویه، دقت لازم را برای انتقال مقادیر مورد نظر به آرگومان‌ها به عمل آورید.

پس از خط اول تعریف رویه، دستورات مورد نظر که در رویه اجرا می‌شوند، قرار داده شده و خاتمه رویه نیز با عبارت End Sub تعیین می‌شود. بدین صورت رویه شما در ماژول فرم تعریف می‌شود.

مثال ۱: پروژه‌ای طراحی کنید که طول و عرض یک مستطیل را از کاربر دریافت کند و با اســتفاده از یک رویه فرعی، محیط و مساحت آن‌را محاسبه کرده و نمایش دهد. برای انجام این کار مراحل بعد را انجام دهید:

۱ - برنامه ویژوال‌بیسیک را اجرا کنید و یک پروژه از نوع Standard EXE به همراه یک فرم مطابق شکل ۱-۷ ایجاد کنید.

شکل ۱–۷

می‌بینید که دو کادر متن برای دریافت طول و عرض مستطیل در نظر گرفته شده است و پس از ورود داده‌ها با کلیک روی دکمه Calculate محیط و مساحت مستطیل محاسبه خواهد شد.

۲ – برای محاسبه محیط و مساحت مستطیل با استفاده از یک رویه فرعی، ابتدا باید آن را تعریف کنید بنابراین به پنجره ماژول فرم رفته و تعریف رویه را به این صورت تایپ کنید، سپس کلید Enter را بفشارید.

```
Private Sub rectangle( )
```

عبارت End Sub در پایین دستوری که تایپ کرده‌اید به طور خودکار ظاهر می‌شود. در واقع این دو دستور محدوده رویه فرعی را تعیین می‌کنند و دستورات مربوط به رویه در بین این دو خط قرار می‌گیرند. رویه‌های فرعی نیز مانند رویه‌های رویداد دارای یک نام هستند. نام‌گذاری رویه‌های فرعی مانند نام‌گذاری متغیرهاست و از همان قوانین پیروی می‌کنند.

۳ – در این مرحله باید دستورات مربوط به محاسبه محیط و مساحت مستطیل را در رویه فرعی rectangle تایپ کنید؛ بنابراین رویه را به صورت بعد تنظیم کنید:

```
Private Sub rectangle()
  Dim sngl As Single, sngw As Single
  Dim sngarea As Single, sngperi As Single
  sngl = Val(txtl.Text)
  sngw = Val(txtw.Text)
  sngarea = sngl * sngw
  sngperi = 2 * (sngl + sngw)
  lblarea.Caption = “Area Is = “ + Str(sngarea)
  lblperi.Caption = “Perimeter Is = “ + Str(sngperi)
End Sub
```

در ایـــن رویه ابتـــدا داده‌هـــای ورودی در کادرهای متن به عدد تبدیل می‌شـــوند و در متغیرهای sngl و sngw ذخیره می‌شـــوند، سپس عملیات محاسبه مساحت و محیط انجام و در نهایت با استفاده از کنترل‌های برچسب نمایش داده می‌شوند.

پس از تعریف یک رویه فرعی لازم است نحوه استفاده از دستورات موجود در رویه یا به عبارت بهتر اجرای رویه فرعی را بدانید به این عمل فراخوانی رویه نیز می‌گویند. برای انجام این کار می‌توانید با توجه به نیاز خود یکی از روش‌های زیر را به کار گیرید:

Call نام رویه (مقدار مربوط به آرگومان اول , . . .)

نام رویه مقدار مربوط به آرگومان اول , . . .

فراخوانی می‌تواند با دســـتور Call یا بدون آن انجام شـــود سپس نام رویه و در صورت استفاده از آرگومان‌ها, مقادیر یا متغیرهای متناسب با هر یک را در داخل پرانتز تایپ کرده و با کاراکتر کاما (,) از یکدیگر جدا کنید.

۴ – برای آن‌که بتوانید دستورات موجود در رویه rectangle را اجرا کنید باید در رویداد Click دکمه Calculate، این دستور را تایپ کنید: Call rectangle

دستور Call در هنگام اجرا، رویه rectangle را فراخوانی می‌کند، به عبارت دیگر اجرای برنامه به رویه rectangle منتقل می‌شـــود و پس از رســـیدن به عبارت End Sub در این رویه اجرای برنامه مجدداً به محل فراخوانی باز می‌گردد و دستور بعد از Call اجرا می‌شود.

۵ – پروژه و فرم را با نام rectangle ذخیره نمایید.

۶ – پروژه را اجرا نمایید. اعداد ۴ و ۶ را به عنوان طول و عرض مستطیل وارد کنید و روی دکمه Calculate کلیک کنید و نتیجه محاسبه را بررسی نمایید.

۷ – پروژه را برای مقادیر دیگر نیز بررسی کنید، سپس به پنجره ویژوال‌بیسیک بازگردید.

تمرین:

پروژه‌ای طراحی کنید که با استفاده از یک رویه فرعی، محیط و مساحت هر دایره به شعاع دلخواه R را محاسبه کند.

مثـال ۲: پـروژه‌ای طراحی کنید که عدد دلخواهی را از کاربر دریافت کرده و زوج یا فرد بودن آن را با اسـتفاده از یک رویه فرعی مشـخص کند. برای انجام این کار مراحل بعد را به ترتیب انجام دهید:

۱ – یک پروژه از نوع Standard EXE و یک فرم مطابق شکل ۲–۷ ایجاد کنید.

۲ – روی منوی Tools در پنجره ویژوال‌بیسـیک کلیک کنید و سپس گزینه Add Procedure ... را برگزینید تا کادر محاوره Add Procedure نمایش داده شود.

۳ – در کادر متـن Name نام رویه را تایپ کنید. برای رویه این مثال، از عبارت oddoreven استفاده کنید (شکل ۳–۷).

۴ – در بخش Type گزینه Sub را برای ایجاد رویه فرعی انتخاب کنید.

۵ – در بخش سوم از تنظیمات کادر محاوره Add Procedure برای میدان دید و شناسایی رویه (Scope) دکمه انتخاب Private را برگزینید.

شکل ۲–۷

شکل ۳–۷

۶ – در پایـان روی دکمه OK در کادرمحاوره Add Procedure کلیک کنید تا رویه فرعی در ماژول فرم اضافه شود.

۷ – اکنـون به پنجره ماژول فرم بروید و رویه oddoreven را به صورت زیر تنظیم کنید:

```
Private Sub oddoreven(sngno As Single)
  If (sngno Mod  2) =  0 Then
    MsgBox “your number is even .» , ,  “Message»
  Else
    MsgBox “your number is odd .» , ,  “Message»
  End If
End  Sub
```

رویه فرعی oddoreven دارای یک آرگومان با نام sngno می‌باشد که از نوع Single تعریف شده است، از این آرگومان برای انتقال عددی که می‌خواهید زوج یا فرد بودن آن را داخل رویه معین کنید، استفاده می‌شود. داخل رویه نیز با استفاده از عملگر Mod و یک If باقی‌مانده تقسیم عدد مورد نظر که در آرگومان sngno ذخیره شده است بر عدد دو بررسی می‌شود. اگر این مقدار برابر صفر باشد کادر پیام، عدد را به عنوان یک عدد زوج و در غیر این صورت عدد را به عنوان یک عدد فرد معرفی خواهد کرد.

۸ – در این مرحله باید رویداد Click دکمه فرمان Compute به شکلی تنظیم شود تا در صورت کلیک روی دکمه Compute رویه فرعی oddoreven فراخوانی شده و زوج یا فرد بودن عدد مربوطه را معین کند، بنابراین رویه رویداد Click دکمه Compute را به این صورت تنظیم کنید:

```
Private Sub cmdcom_Click()
  Dim snginput As Single
  snginput = Val(txtno.Text)
  Call oddoreven(snginput)
End Sub
```

در این رویداد پس از تعریف متغیر snginput، با استفاده از تابع Val عبارت تایپ شده در کادر متن به مقدار عددی تبدیل و در متغیر snginput ذخیره می‌شود. در مرحله بعد باید رویه فرعی oddoreven را فراخوانی کرد. برای این کار با استفاده از دستور Call رویه فراخوانی می‌شود و برای آن‌که عدد مورد نظر کاربر که در متغیر snginput ذخیره شده است در اختیار رویه قرار گیرد، نام متغیر snginput پس از نام رویه و در داخل پرانتز ذکر می‌شود، در این صورت در هنگام اجرای دستور Call بین متغیر snginput و متغیر sngno ارتباط برقرار می‌شود و در واقع هنگام اجرای رویه، مراجعه به متغیر sngno به منزله مراجعه، خواندن و ذخیره‌سازی اطلاعات در متغیر snginput است.

نکته در صورتی‌که یک رویه دارای آرگومان باشد، اگر بعد از تعریف رویه، نام رویه را در ماژول فرم تایپ کنید؛ ویژوال‌بیسیک یک راهنمایی به صورت ToolTip را که شامل نام رویه و نام و نوع آرگومان‌های آن می‌باشد، نمایش می‌دهد (شکل ۴-۷).

شکل ۴-۷

۹ – پروژه و فرم را با نام oddeven ذخیره کرده، سپس آن را اجرا کنید.

۱۰ – در کادر متن فرم برنامه، عدد ۴ را تایپ کنید و روی دکمه Compute کلیک کنید. در این صورت رویه فراخوانی می‌شود و عدد ۴ که در متغیر snginput ذخیره شده است، در آرگومان sngno کپی می‌شود و سپس نوع عدد تعیین شده و در کادر پیام، پیغام «عدد زوج است» نمایش داده می‌شود.

۱۱ – پروژه را برای چند عدد دیگر بررسی کنید و به محیط برنامه‌نویسی ویژوال‌بیسیک باز گردید.

۷-۲ رویه‌های تابعی (Function Procedure)

نوع دیگری از رویه‌ها، رویه‌های تابعی هستند. رویه‌های تابعی شبیه به رویه‌های فرعی هستند با این تفاوت که از آن‌ها در مواردی که لازم باشد، استفاده می‌شود و پس از انجام مجموعه‌ای از محاسبات، مقداری به محل فراخوانی رویه، بازگشت داده می‌شود. قبلاً از بعضی رویه‌های تابعی آماده در ویژوال‌بیسیک استفاده کرده‌اید، برای مثال می‌توان به رویه تابعی Val، Str و نظایر آن‌ها اشاره کرد. تابع Val با دریافت یک رشته از کاراکترهای رقمی آن را به مقدار عددی تبدیل می‌کند و تابع Str با دریافت یک مقدار عددی آن را به رشته تبدیل می‌نماید. در ویژوال‌بیسیک امکان تعریف رویه‌های تابعی مورد نیاز توسط برنامه‌نویس نیز وجود دارد.

تعریف و فراخوانی رویه‌های تابعی نیز شبیه به رویه‌های فرعی است. برای تعریف رویه‌های تابعی می‌توانید رویه را در ماژول فرم تایپ کنید یا از کادر محاوره Add Proc-dure استفاده کنید. شکل کلی تعریف رویه‌های تابعی به این صورت است:

Private Function نام تابع (نام آرگومان As نوع آرگومان , . . .) As نوع مقدار بازگشتی

دستور(ات)
نام تابع=مقداربازگشتی

End Function

تعریف یک رویه تابعی مانند رویه فرعی بوده و با کلمه کلیدی Private آغاز شده و باعث می‌شود رویه تابعی به صورت محلی تعریف شود. در ادامه تعریف رویه تابعی کلمه کلیدی Function و به دنبال آن نام رویه تابعی قرار می‌گیرد. قوانین نام‌گذاری رویه‌های تابعی مانند رویه‌های فرعی و متغیرهاست.

پس از تعیین نام رویه تابعی باید نام و نوع آرگومان‌های رویه را مشخص کنید تا با استفاده از آن‌ها مقادیر مورد نیاز به رویه ارسال شوند. نحوه تعریف و استفاده از آرگومان‌ها در رویه‌های تابعی مانند رویه‌های فرعی می‌باشد و استفاده از آن‌ها و تعداد و نوع آرگومان‌ها اختیاری است. پس از تعریف رویه تابعی باید دستورات مورد نظر خود را در رویه قرار داده و خاتمه رویه را با عبارت End Function تعیین کنید.

اما نکته مهمی که در رابطه با رویه‌های تابعی وجود دارد و در رویه‌های فرعی موجود نیست، مربوط به نحوه بازگشت یک مقدار به محل فراخوانی می‌شود. همان‌طور که در شکل کلی تعریف رویه تابعی می‌بینید، باید مقداری را که می‌خواهید بازگشت دهید به نام تابع انتساب دهید. با رسیدن به این دستور و اجرای آن مقدار تعیین شده به محل فراخوانی بازگشت داده می‌شود. علاوه بر این باید نوع مقدار بازگشتی در انتهای خط اول تعریف رویه و پس از معرفی آرگومان‌ها معین شود. برای فراخوانی یک رویه تابعی می‌توانید به این شکل عمل کنید:

```
نام‌رویه‌تابعی=نام‌یک‌متغیرمتناسب‌بانوع‌داده‌بازگشتی(مقدارمربوط‌به‌آرگومان‌اول, . . . )
```

هم‌چنین می‌توانید رویه تابعی را بدون آن‌که مقداری را به عنوان نتیجه محاسبات بازگشت دهد، تعریف کنید. در این صورت فراخوانی رویه تابعی می‌تواند به صورت فراخوانی یک رویه فرعی باشد.

Call نام رویه (مقدار مربوط به آرگومان اول , . . .)

نام رویه مقدار مربوط به آرگومان اول , . . .

نکته: مقدار بازگشتی یک تابع می‌تواند به عنوان مقدار ارسالی برای تابع دیگری استفاده شود.
نوع مقدار بازگشتی در تعریف تابع با توجه به نوع داده‌ای که به محل فراخوانی بازگشت می‌دهد تعیین می‌شود.
اگر نوع مقدار بازگشتی در تعریف تابع تعیین نشود به طور پیش‌فرض از نوع داده Integer استفاده می‌شود.

مثال ۳: پروژه‌ای طراحی کنید که با استفاده از یک تابع، باقیمانده تقسیم دو عدد دلخواه را محاسبه کرده و نمایش دهد. برای این کار مراحل بعد را انجام دهید:

۱ – برنامه ویژوال‌بیسیک را اجرا کنید و یک پروژه از نوع Standard EXE به همراه یک فرم مطابق شکل ۵–۷ ایجاد کنید.
دو کادر متن روی فرم قرار دارند که اعداد موردنظر در آن‌ها وارد خواهند شد و کاربر با کلیک روی دکمه Remainder باقی‌مانده تقسیم دو عدد را مشاهده خواهد کرد.

شکل ۵–۷

۲ – به پنجره ماژول فرم بروید و تابع remainder را به این صورت برای محاسبه باقیمانده تقسیم دو عدد تعریف کنید:

```
Private Function remainder (intno1 As Integer, intno2 As _ Integer)As Integer
    Dim intresult  As Integer
    intresult = intno1 Mod intno2
    remainder = intresult
End Function
```

تابع remainder دارای دو آرگومان intno۱ و intno۲ است که برای دریافت مقدار مقسوم و مقسوم علیه به‌کار می‌روند؛ سپس در بخش دستورات این رویه ابتدا یک متغیر محلی دیگر برای ذخیره‌سازی باقی‌مانده تقسیم، تعریف می‌شود و به دنبال آن با استفاده از عملگر Mod باقی‌مانده تقسیم دو عدد در این متغیر (intresult) ذخیره می‌شود. اما برای آن‌که نتیجه محاسبه به محل فراخوانی بازگشت داده شود از دستور remainder = intresult استفاده می‌شود و نتیجه محاسبه که در متغیر intresult ذخیره شده است به نام تابع نسبت داده می‌شود. به علاوه چون متغیر intresult از نوع عدد صحیح (Integer) معرفی شده است، بنابراین در انتهای خط اول تعریف تابع نوع مقدار بازگشتی نیز از نوع Integer معرفی شده است.

۳ – در این مرحله رویداد Click دکمه فرمان Remainder را به صورت زیر تنظیم کنید:

```
Private Sub cmdrem _Click()
  Dim intm  As Integer, intn As Integer
  Dim intmod  As Integer
  intm = Val (txtno1.Text)
  intn = Val (txtno2.Text)
  intmod = remainder (intm, intn)
  lblres.Caption = "Answer Is = " + Str(intmod)
End Sub
```

با اجرای این رویه ابتدا مقادیری که در کادرهای متن تایپ شده‌اند، بعد از تبدیل در متغیرهای intm و intn ذخیره می‌شوند؛ سپس تابع remainder با متغیرهای intm و intn فراخوانی می‌شود و در نتیجه اجرای برنامه به بخش تعریف رویه تابعی remainder منتقل شده و بین متغیرهای ارسالی intm و intn با آرگومان‌های intno۱ و intno۲ ارتباط برقرار می‌کند، سپس اجرای دستورات موجود در رویه تابعی آغاز می‌شود و با انجام محاسبات و رسیدن به دستور remainder= intresult و انجام آن, اجرای برنامه به محل فراخوانی یعنی رویداد Click دکمه remainder و دستور intmod= remainder (intm , intn) بازمی‌گردد و مقدار موجود در متغیر intresult در متغیر intmod ذخیره می‌شود و در پایان این مقدار در کنترل برچسب نمایش داده شده و رویداد خاتمه می‌یابد. نوع متغیر intmod با نوع مقدار بازگشتی در تعریف تابع و همین‌طور نوع متغیر intresult هماهنگی لازم را دارد.

۴ – پروژه و فرم را با نام remainder ذخیره کنید، سپس آن را اجرا نمایید.

۵ – اعداد ۱۲ و ۵ را به ترتیب در کادرهای متن وارد کنید، سپس روی دکمه Remainder کلیک کرده و نتیجه محاسبه را مشاهده نمایید.

۶ – پروژه را برای مقادیر دیگری نیز آزمایش کنید، سپس به پنجره ویژوال‌بیسیک بازگردید.

تمرین:

پروژه فوق را به‌گونه‌ای تنظیم کنید که اگر عدد اول از عدد دوم کوچک‌تر باشد با استفاده از یک رویه فرعی, خطای کاربر به وی گزارش شود و محاسبات نیز انجام نشود.

پروژه‌ای طراحی کنید که یک عدد دلخواه را از کاربر دریافت کند و با استفاده از یک رویه تابعی قدر مطلق آن را محاسبه کرده و نمایش دهد.

پروژه‌ای طراحی کنید که با استفاده از یک تابع، قرینه هر عدد دلخواه را نمایش دهد.

رویه‌ای تابعی بنویسید که هر عدد دلخواه x را دریافت کند و مقدار معکوس آن را $\frac{۱}{X}$ محاسبه کرده و بازگشت دهد.

۳–۷ روش‌های ارسال مقادیر به رویه‌های فرعی و تابعی

تاکنون نحوه استفاده از رویه‌های فرعی و تابعی و چگونگی انتقال مقادیر مورد نیاز به رویه‌ها را فرا گرفتید. در واقع دو روش مختلف برای ارسال مقادیر مورد نیاز به رویه‌ها وجود دارد که هر یک ویژگی‌هایی را به همراه دارند. اولین روش فراخوانی متغیرها با مرجع (Call By Reference) و دومین روش فراخوانی متغیرها با مقدار (Call By Value) است.

۱–۳–۷ روش فراخوانی با مرجع

در این روش، ارتباط مستقیمی بین متغیر ارسالی و آرگومان متناظر آن در رویه مربوطه برقرار می‌شود به گونه‌ای که متغیر و آرگومان متناظر با آن به محل مشترکی در حافظه اشاره می‌کنند. بنابراین هرگونه تغییر در محتویات آرگومان‌ها در رویه، سبب تغییر در متغیرهای ارسالی خواهد شد. در واقع شما تاکنون با این روش، مقادیر خود را به رویه‌ها

ارسال کرده‌اید.

در صورتی که بخواهید متغیرها را با روش ارسال با مرجع به یک رویه ارسال نمایید، کلمه کلیدی ByRef را در ابتدای نام آرگومان‌ها در تعریف رویه ذکر کنید. البته در صورت عدم استفاده از کلمه کلیدی ByRef نیز نحوه ارسال متغیرها به طور پیش فرض از نوع فراخوانی با مرجع می‌باشد.

فرض کنید می‌خواهید با استفاده از یک رویه فرعی محتویات دو متغیر را با یکدیگر جابه‌جا کنید، برای این منظور رویه زیر را در نظر بگیرید:

```
Private Sub moving(ByRef sngx As Single,ByRef sngy As Single)
  Dim sngtemp As Single
  sngtemp = sngy
  sngy = sngx
  sngx = sngtemp
End Sub
```

رویه فرعی moving دو آرگومان دارد که برای دریافت مقادیر دو متغیر به‌کار می‌روند و در ابتدای نام هر یک از آن‌ها از کلمه کلیدی ByRef استفاده شده است، این عبارت تعیین می‌کند که نحوه ارسال متغیرها به رویه به صورت فراخوانی با مرجع باشد.

هم‌چنین با تعریف یک متغیر واسطه با نام sngtemp در داخل رویه فرعی، ابتدا مقادیر یکی از متغیرها را در متغیر واسطه (sngtemp) ذخیره کرده، سپس مقدار متغیر sngx را در متغیر sngy و در آخر مقدار متغیر واسطه را در متغیر sngx کپی می‌کند. به این صورت مقادیری که در آرگومان‌های sngx و sngy قرار دارند با یکدیگر تعویض می‌شوند؛ اما باید ببینیم که چگونه از این رویه می‌توان برای تعویض مقدار دو متغیر در خارج از رویه استفاده کرد. فرض کنید که برای این کار رویه فرعی moving به صورت زیر فراخوانی می‌شود:

```
Dim x As Single
Dim y As Single
x = 28
y = -10
Call moving(x, y)
Print x, y
```

پس از تعریف و مقداردهی دو متغیر x و y, رویه فرعی فراخوانی شده است. در این‌جا اجـرای برنامـه به محل رویه منتقل می‌شـود و چون از کلمه کلیـدی ByRef در ابتدای نام آرگومان‌ها در تعریف رویه اسـتفاده شده اسـت، ارتباط مستقیمی بین متغیرهای ارسالی و آرگومان‌های رویه moving به وجود می‌آید؛ در نتیجه تغییر در محتویات آرگومان‌های sngx و sngy سـبب تغییـر در محتویات متغیرهای ارسـالی x و y و تعویض محتویات دو متغیر می‌شـوند. بنابراین پس از خاتمه اجرای رویه، دسـتور Print مقادیر جابه‌جا شـده را برای متغیرهای x و y در روی فرم نمایش خواهد داد.

نکته اگر رویه فرعی moving به صورت Call moving (x+۱, y+۱) فراخوانی شـود مقدار متغیرها پس از فراخوانی رویه همان مقادیر قبل از فراخوانی خواهد بود.

اگر مقادیر ثابت به رویه فرعی moving ارسـال شـود کلمه‌های کلیـدی ByRef هیچ تأثیری روی نحوه ارسال آن‌ها نمی‌گذارد.

در روش فراخوانی با مرجع، هم‌نام یا غیرهم‌نام بودن متغیرهای ارسالی با آرگومان‌های رویه اهمیتی ندارد.

تمرین:

پروژه‌ای طراحی کنید تا بتوانید با اسـتفاده از یک فرم, یک کنترل دکمه فرمان و با اسـتفاده از رویه فرعی moving تعویض محتویات دو متغیر را با اسـتفاده از روش فراخوانی با مرجع بررسی کنید.

۲–۳–۷ روش فراخوانی با مقدار

گاهی اوقات لازم اسـت تا بین متغیرهای ارسـالی و آرگومان‌های متناظر آن‌ها ارتباطی بـه وجود نیاید و در صـورت تغییر در مقدار آرگومان‌ها، در محتویات متغیرهای ارسـالی تغییر حاصل نشـود. در این صورت امکان استفاده از روش فراخوانی با مرجع مفید نیست. روش دیگـری که برای انجام این گونه عملیات مناسـب اسـت، روش فراخوانی با مقدار می‌باشـد. در این روش با ذکر کلمـه کلیدی ByVal به‌جای عبـارت ByRef از ایجاد ارتباط مسـتقیم بیـن متغیرهای ارسـالی و آرگومان‌ها جلوگیـری به عمل می‌آیـد و فقط مقادیر موجود در متغیرهای ارسـالی در داخل آرگومان‌های متناظر آن‌ها کپی خواهد شد. به عنوان

مثال اگر تعریف رویه فرعی moving را به این صورت تغییر دهید:

```
Private Sub moving(ByVal sngx As Single, ByVal sngy As Single)
```

در صورت فراخوانی رویه فرعی moving با حالت فوق، در واقع عدد ۲۸ در آرگومان sngx رویه و عدد ۱۰ – در آرگومان sngy رویه کپی خواهند شد و پس از تعویض محتویات این دو آرگومان در داخل رویه وخروج از رویه، مقادیر مربوط به این آرگومان‌ها از بین خواهد رفت و با بازگشت به محل فراخوانی و اجرای دستور Print همان مقادیر قبل از فراخوانی برای متغیرهای x و y نمایش داده می‌شوند.

تمرین:
رویه moving را در پروژه‌ای که در تمرین قبل ایجاد کرده‌اید به شکل فراخوانی با مقدار تغییر دهید سپس آن را اجرا کرده و نتیجه را با حالت قبل بررسی کنید.

مثال ۴: یک رویه فرعی بنویسید تا دو عدد را دریافت کرده، میانگین آن‌ها را محاسبه کند و نمایش دهد. برای این کار یک رویه average را به صورت زیر بنویسید:

```
Private Sub average(ByVal sngno1 As Single,ByVal sngno2 _
                    As Single, ByRef sngave As Single)
  sngave = (sngno1 + sngno2) / 2
End Sub
```

در این رویه از سه آرگومان استفاده شده است. آرگومان‌های sngno۱ و sngno۲ برای ارسال اعداد به داخل رویه و به صورت فراخوانی با مقدار تعریف شده‌اند، زیرا لازم نیست این دو آرگومان در متغیرهایی که در زمان فراخوانی به رویه ارسال می‌شوند، تغییری ایجاد کنند. اما آرگومان سوم یعنی sngave با روش ارسال با مرجع تعریف شده است؛ زیرا پس از محاسبه معدل دو عدد در محل فراخوانی مورد استفاده قرار می‌گیرد. برای چنین رویه‌ای، فراخوانی به‌صورت زیر خواهد بود:

```
Call    average (x , y , average)
```

در فراخوانی فوق ابتدا مقادیر متغیرهای ارسالی x و y در آرگومان‌های sngno۱ و sngno۲ کپی می‌شوند و بین متغیر average و آرگومان sngave در رویه به‌گونه‌ای ارتباط

برقرار می‌شـــود که تغییر در مقدار آرگومان sngave, محتویات متغیر ارســـالی average را نیز تحت تأثیر قرار می‌دهد و سبب اعمال همان تغییرات روی آن خواهد شد. می‌بینید که با این روش نیز می‌توان با استفاده از رویه‌های فرعی نتیجه محاسبات را به محل فراخوانی بازگشـــت داد. به این صورت پس از محاسبه متوسط دو عدد و ذخیره این مقدار در متغیر sngave رویـــه خاتمـــه یافته و به محل فراخوانـــی باز می‌گردید. اکنـــون می‌توانید در محل فراخوانی رویه با استفاده از متغیر average به معدل دو عدد دسترسی پیدا کنید.

تمرین:

یک پروژه از نوع Standard EXE به همراه یک فرم به گونه‌ای طراحی کنید تا با استفاده از رویه average بتوان معدل هر دو عدد دلخواه را محاسبه و روی فرم مشاهده کرد.

۴–۷ نحوه استفاده از نام آرگومان‌ها در رویه‌ها

تاکنون با استفاده از رعایت ترتیب در ارسال آرگومان‌های رویه‌ها توانستیم داده‌ها را به آرگومان‌های متناظرشـــان نسبت دهیم، شـــما می‌توانید با استفاده از نام آرگومان‌ها در زمان فراخوانـــی، ترتیـــب ذکر داده‌ها را بدون در نظر گرفتن ترتیـــب قرارگیری آن‌ها در تعریف رویه، به رویه ارسال کنید.

این روش باعث می‌شـــود در زمان زیاد بودن تعداد آرگومان‌ها کار فراخوانی آسان‌تر و احتمال اشتباه در ارسال داده‌ها به آرگومان‌ها کمتر شود. برای استفاده از نام آرگومان‌ها در زمان فراخوانی می‌توانید به صورت زیر عمل کنید: مقدار ارسالی =: نام آرگومان

فرض کنید می‌خواهید یک رویه تابعی بنویســـید که به وســـیله آن بتوان مجموع اعداد زوج بین دو عدد طبیعی دلخواه را محاسبه کرد. برای این کار به این دستورات توجه کنید.

(m<n)

```
Function Myfunc(m As Integer, n As Integer) As Single
    Dim i As Integer, sum As Single
    If m Mod 2 = 0 Then m = m+2 Else m = m+ 1
    sum= 0
```

```
    For i = m To n-1 Step 2
       sum=sum+i
    Next i
    Myfunc = sum
End Function
```

در تعریف این تابع دیده می‌شود دو آرگومان m و n (با فرض m<n) به تابع ارسال می‌شوند. در تابع ابتدا زوج یا فرد بودن m به وسیله یک if بررسی می‌شود تا بتوان عدد زوج بعد از m را تعیین کرد اگر m زوج باشد حاصل M mod 2 = 0 درست خواهد بود و با اضافه شدن دو واحد به m عدد زوج بعدی محاسبه می‌شود اما اگر m فرد باشد نتیجه M mod 2 = 0 نادرست خواهد بود و با اضافه شدن یک واحد به m باز عدد زوج بعدی محاسبه می‌شود، سپس مجموع اعداد زوج بین m و n با یک حلقه For و توسط دستور Sum = Sum+i به دست می‌آید و در پایان مقدار Sum به عنوان مجموع اعداد زوج بین m و n بازگشت داده می‌شود. حلقه For نیز تا رسیدن به مقدار n - ۱ اجرا شده تا در صورت زوج بودن n، خود n در مجموع وارد نشود.

بعد از درک نحوه کارکرد تابع به فراخوانی آن می‌پردازیم. تاکنون نحوه فراخوانی چنین توابعی را فراگرفته‌اید، مثلاً برای نمایش مجموع اعداد زوج بین ۲ و ۱۲ از فرمان بعد استفاده می‌کنیم:

```
Print  Myfunc (2,12)
```

اگر در فراخوانی جای اعداد ۲ و ۱۲ عوض شود نتیجه کار نامناسب خواهد بود. اما با فراخوانی با استفاده از نام آرگومان‌ها امکان اشتباه از بین می‌رود. برای نمونه در مثال فوق فراخوانی با استفاده از نام آرگومان‌ها به صورت زیر خواهد بود:

```
Print  Myfunc (n := 12 , m := 2)
```

می‌بینید که برای فراخوانی با استفاده از نام آرگومان‌ها باید از نام آرگومان به همراه «:=» در قبل از مقدار ارسالی استفاده کنید. البته فراخوانی قبل را می‌توان به صورت زیر نیز نوشت:

```
Print  Myfunc (m := 2 , n := 12)
```

تمرین:
رویه تابعی Myfunc را به گونه‌ای تنظیم کنید که بدون فرض M<N نیز بتواند محاسبات را به‌طور صحیح انجام دهد.

۵–۷ خروج از یک رویه با استفاده از دستورات Exit Sub و Exit Function

گاهی اوقات لازم است تا قبل از این‌که تمام دستورات یک رویه اجرا شده و به انتهای رویه برسید، از رویه خارج شده و به محل فراخوانی بازگردید. به این منظور برای خروج از یک رویه فرعی از دستور Exit Sub و برای خروج از یک رویه تابعی از دستور Exit Function استفاده کنید.

مثال ۵: یک رویه تابعی تعریف کنید تا تعداد ارقام هر عدد طبیعی دلخواه را محاسبه کرده و نمایش دهد. برای این کار رویه تابعی digits را به صورت زیر تعریف کنید:

```
Private Function digits(ByVal sngno As Integer) As Integer
    Dim intdigits As Integer
    If sngno < 0 Then
        digits = -1
        Exit Function
    End If
    While (sngno > 0 )
        0 ngno = sngno \1
        intdigits = intdigits + 1
    Wend
    digits = intdigits
End Function
```

این تابع دارای یک آرگومان است که برای انتقال عدد مورد نظر به داخل تابع استفاده شده است. علاوه بر این روش ارسال عدد مورد نظر نیز ارسال با مقدار خواهد بود تا تغییر روی مقدار آرگومان sngno در رویه تابعی، تأثیری روی متغیر ارسالی نگذارد. در ضمن نوع مقدار داده بازگشتی یعنی تعداد ارقام نیز از نوع صحیح است؛ بنابراین نوع داده بازگشتی در انتهای خط اول رویه و از نوع Integer تعریف می‌شود.

به‌هرحال در صورت فراخوانی این تابع و اجرای آن ابتدا مقدار متغیر ارسالی در

آرگومان sngno کپی می‌شود، سپس با اجرای یک دستور If مقدار عدد ارسال شده بررسی می‌شود و در صورتی‌که مقدار آن کوچک‌تر از صفر باشد، مقدار ۱- را بازگشت می‌دهد. از این مقدار می‌توان در محل فراخوانی برای اطمینان از نحوه ارسال صحیح مقدار به تابع استفاده کرد سپس با استفاده از دستور Exit Function اجرای رویه خاتمه یافته و به محل فراخوانی بازمی‌گردید. این عمل سبب خواهد شد تا کاربر از ارسال مقادیر عددی اشتباه مطلع شود، اما اگر مقدار ارسالی بزرگ‌تر از صفر باشد با استفاده از یک حلقه While...Wend تعداد ارقام عدد محاسبه شده و در متغیر intdigits ذخیره می‌شود و پس از خاتمه حلقه این مقدار با استفاده از دستور digits= intdigits بازگشت داده خواهد شد.

تمرین:
پروژه‌ای طراحی کنید تا با استفاده از یک رویه فرعی بتواند تعداد ارقام هر عدد طبیعی دلخواه را محاسبه کرده و در یک کادر پیام نمایش دهد، به علاوه در صورت ارسال مقادیر منفی به رویه، پیام خطایی نمایش داده شود و اجرای رویه نیز خاتمه یابد.

۶-۷ رویه‌های محلی و عمومی

رویه‌هایی که تاکنون تعریف شده‌اند از نوع محلی یا به عبارت دیگر Private می‌باشند و فقط در همان ماژولی که تعریف می‌شوند قابل شناسایی و فراخوانی هستند، اما گاهی اوقات لازم است تا یک رویه تابعی یا فرعی را که در یک ماژول فرم تعریف شده است در ماژول فرم دیگری فراخوانی کنید.

در این صورت در زمان اجرای برنامه و با فراخوانی رویه مزبور پیام‌خطایی مطابق‌شکل ۶-۷ نمایش داده می‌شود.

شکل ۶-۷

۱-۶-۷ نحوه ایجاد رویه‌های عمومی در ماژول فرم

برای آن‌که عملیات فراخوانی یک رویه از داخل ماژول فرم دیگری امکان‌پذیر باشد به جای کلمه کلیدیPrivate، از کلمه کلیدی Public استفاده کنید. به این شکل رویه به

عنـوان یک رویه عمومی معرفی خواهد شـد و علاوه بر فراخوانـی آن در ماژولی که تعریف شـده است در سـایر ماژول‌ها امکان فراخوانی خواهد داشت، به چنین رویه‌هایی رویه‌های عمومی می‌گویند.

برای فراخوانی رویه‌های عمومی باید قبل از ذکر نام رویه, نام ماژول فرمی را که رویه در آن تعریف شده است، تایپ کنید و سپس کاراکتر نقطه را تایپ کرده و نام رویه را ذکر نمایید. این مطلب را می‌توان به این صورت بیان کرد:

(مقادیرارسالی)نام‌رویه‌فرعی.نام‌فرمی‌که‌رویه‌فرعی‌در‌آن‌تعریف‌شده‌است Call

(مقادیرارسالی) نام رویه تابعی. نام‌فرمی که رویه تابعی در آن تعریف شده است.

نکته در صورتی‌که حوزه شناسـایی (Scope) یک رویه در زمان تعریف، تعیین نشـود رویه به طور پیش‌فرض به صورت عمومی تعریف خواهد شد.

مثـال ۶: پـروژه‌ای طراحی کنید که میزان مصرف برق هر سـاختمان دلخواهی را محاسبه کرده، مقادیر فعلی و قبلی را که از کنتور برق قرائت شده‌اند، وارد کند. با محاسبه میزان مصرف، هزینه برق مصرفی در سـاختمان مورد نظر به دسـت می‌آید. به این منظور مراحل بعد را به ترتیب انجام دهید:

۱ – یک پروژه جدید از نوع Standard EXE ایجاد کنید، سپس یک فرم مطابق شکل ۷–۷ برای دریافت مقادیر ورودی ایجاد کنید.

شکل ۷–۷

در این فرم از چهار کادر متن برای دریافت تاریخ قرائت کنتور و میزان مصرف برق در مراجعه قبلی و مراجعه فعلی استفاده می‌شود و از دکمه Compute برای انجام محاسبات و نمایش نتیجه آن روی فرم دیگری استفاده می‌شود.

۲- برای انجام محاسبات از یک رویه تابعی و یک رویه فرعی استفاده کنید، این رویه‌ها را در ماژول فرم و به این صورت تنظیم کنید:

```
Public Function expense (ByVal sngpv As Single, ByVal sngcv As Single) As Currency
  Dim sngresult As Single , intlevel  As Integer
  Dim curpay As Currency
  sngresult = sngcv - sngpv
  Select Case sngresult
        Case Is < 500  :
            intlevel = 1
        Case Is < 1000 :
            intlevel = 2
        Case Is < 1500 :
           intlevel = 3
        Case Else
            intlevel = 4
   End Select
 Call payment (intlevel, sngresult, curpay)
 expense = curpay
End Function
Private Sub payment(intlevel As Integer, ByVal sngresult _
       As Single, ByRef curpay As Currency)
       Select Case intlevel
       Case 1 : curpay = sngresult * 100
       Case 2 : curpay = sngresult * 250
       Case 3 : curpay = sngresult * 500
       Case 4 : curpay = sngresult * 1000
       End Select
 End Sub
```

رویه تابعی expense به صورت عمومی تعریف شده و دارای دو آرگومان از نوع Single بـــرای دریافت مقادیر مصرف قبلی و فعلی اســـت. این تابـــع با توجه به مقدار مصرف برق بر اســـاس کیلووات، ســـطح مصرف و هزینه آن را محاسبه کرده و به صورت یک مقدار از نوع Currency بازمی‌گرداند، پس از فراخوانی این تابع ابتدا مقادیر قرائت شده به ترتیب در آرگومان‌های sngpv و sngcv کپی می‌شوند، سپس تفاضل این دو مقدار از یکدیگر کسر شده و به عنوان میزان مصرف در متغیر sngresult ذخیره می‌شوند و بر همین اساس با استفاده از یک دســـتور Select Case ســـطح مصرف (intlevel) نیز تعیین می‌شود تا با توجه به این سطح, هزینه مصرف برق محاســـبه شود. به این منظور چهار سطح مصرف از کم تا خیلی زیاد در نظر گرفته شده است، پس از محاسبه سطح مصرف برق, نوبت محاسبه هزینه آن می‌شود. این کار با استفاده از یک رویه فرعی انجام می‌شود. رویه payment با دریافت سطح مصرف (intlevel) و مقدار مصرف برق (sngresult), هزینه آن را با استفاده از یک دستورSelect Case بر اساس سطح مصرف محاسبه کرده و در متغیر curpay ذخیره می‌کند. به علاوه متغیر curpay به صورت فراخوانی با مرجع به رویه ارسال می‌شود تا میزان هزینه در محل فراخوانی قابل استفاده باشد، این رویه از نوع محلی تعریف شده است. پس از آن‌که رویه payment خاتمه یافت، اجرای برنامه به محل فراخوانی آن یعنی رویه تابعی expense بازگشته و با استفاده از دستور curpay =expense مقدار هزینه مصرف برق را به محل فراخوانی بازمی‌گرداند.

۳– نام کادرهای متن مربوط به تاریخ قرائت و میزان مصرف قبلی را به ترتیب روی tx - pd و txtpv و تاریخ قرائت و میزان مصرف فعلی را به ترتیب روی txtcd و txtcv تنظیم کنید.

۴ – بـــرای نمایش میـــزان هزینه مصرف برق در یک فرم جداگانـــه، احتیاج به یک فرم جدید خواهید داشـــت، بنابراین یک فرم جدید مطابق شـــکل ۸–۷ ایجاد کنید، سپس فرم و پروژه را با نام viewpayment ذخیره کنید.

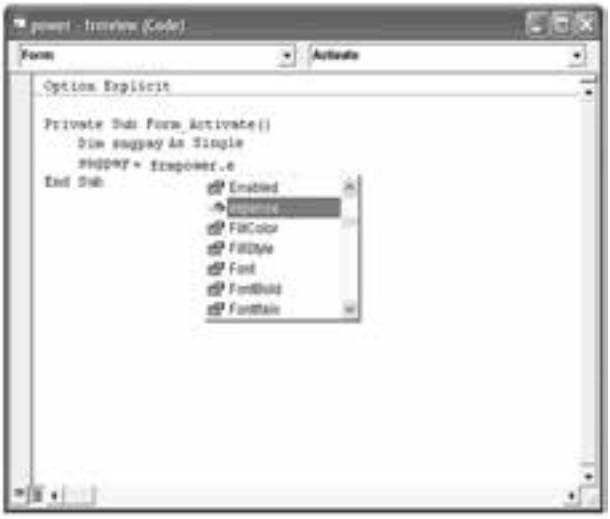
شکل ۸–۷

سپس رویداد Activate این فرم و رویداد Click دکمه OK را به صورت زیر تنظیم کنید:

```
Private Sub Form_Activate()
    Dim sngpay As Single
    Dim sngpv As Single
    Dim sngcv As Single
    sngpv = Val(frmpower.txtpv.Text)
    sngcv = Val (frmpower.txtcv.Text)
    sngpay = frmpower.expense (sngpv, sngcv)
    lblview.Caption = "Payment For Date " + Trim (frmpower.txtpd.Text) + _
    " To Date "+Trim (frmpower.txtcd.Text)
    lblpay.Caption = sngpay
End  Sub

Private Sub cmdok_Click()
   Unload Me
   frmpower.SetFocus
   frmpower.txtpd.SetFocus
End  Sub
```

در این فرم از رویداد Activate فرم View Payment استفاده شده است تا پس از نمایش فرم، گزارش محاسبات در روی فرم به نمایش درآید. در این رویه با فراخوانی تابع expense محاسبات انجام می‌شود. همان‌طور که ملاحظه می‌کنید، برای فراخوانی این تابع ابتدا نام فرمی که تابع در ماژول آن تعریف شده، یعنی frmpower ذکر شده است. البته این کار زمانی مؤثر است که رویه به صورت عمومی (Public) تعریف شده باشد در این صورت در زمان تایپ نام فرم (frmpower) و کاراکتر نقطه یک لیست حاوی نام رویه expense که در ماژول frmpower تعریف شده است، نمایش داده خواهد شد (شکل ۹-۷). نکته دیگری که در فراخوانی رویهexpense مهم است، نحوه ارسال‌مقادیر تایپ شده در کادرهای متن است.

برای دسترسی خصوصیات کنترل‌های یک فرم در فرم دیگر می‌توانید ابتدا نام فرم، سپس نام کنترل و بعد نام خصوصیت را ذکر کنید؛ به این صورت مقادیر مصرف برق با استفاده از تابع Val به نوع عددی تبدیل شده و به تابع ارسال می‌شوند. مقدار هزینه مصرف برق با استفاده از تابع expense محاسبه شده و برگردانده می‌شود و در نهایت این مقدار در متغیر sngpay ذخیره می‌شود. در ادامه اجرای رویه Activate تاریخ‌های مربوط به قرائت کنتور به وسیله کنترل برچسب lblview و هزینه مصرف برق به وسیله کنترل برچسب lblpay نمایش داده می‌شوند.



شکل ۹–۷

رویداد Click دکمه OK نیز به‌گونه‌ای تنظیم شده است تا در صورتی‌که کاربر روی این دکمه کلیک کند، فرم View Payment بسته شده و فوکوس با متد SetFocus به فرم frmpower و سپس کنترل کادر متن txtpd داده شود.

۵ – به ماژول فرم frmpower بروید و دستور frmview.show را در رویداد Click دکمه Compute و دستور Unload Me را در رویداد Click دکمه Exit تایپ کنید.

۶ – فرم و پروژه را با نام power ذخیره کنید. پروژه را اجرا کنید و داده‌ها را مطابق شکل ۱۰–۷ در کادرهای متن تایپ کرده و روی دکمه Compute کلیک کنید، نتیجه محاسبات مطابق شکل ۱۱–۷ در فرم دوم قابل مشاهده خواهد بود.

۷–با کلیک روی دکمه Exit به اجرای پروژه خاتمه دهید و به پنجره ویژوال‌بیسیک بازگردید.

شکل ۷–۱۰

شکل ۷–۱۱

تمرین:
پـروژه power را به‌گونـه‌ای تنظیم کنید که رویه فرعـی payment در ماژول فرم View Payment تعریف شود.

۷–۶–۲ نحوه ایجاد رویه‌های محلی و عمومی با استفاده از ماژول کد

روش دیگـری نیز در ویژوال‌بیسـیک وجود دارد که در آن بـه جای تعریف رویه‌های موردنظر در ماژول فرم از ماژول کد (Code Module) استفاده می‌شود. ماژول کد یکی دیگر از اجزای تشکیل دهنده در پروژه‌هاست. ماژول‌های کد فقط از دستورالعمل‌ها تشکیل می‌شوند و شکل ظاهری مانند فرم‌ها ندارند. یکی دیگر از کاربردهای ماژول کد، تعریف متغیرهای عمومی اسـت. نحوه تعریف و فراخوانی رویه‌ها در یک ماژول کد مانند ماژول فرم است.

مثـال ۷: پـروژه power را به گونه‌ای تغییر دهید تا رویه‌های مربوطه را با اسـتفاده از مـاژول کد تعریف و فراخوانی کند. به این منظور عملیات زیر را به ترتیب انجام دهید:

۱– ابتـدا یک ماژول کد به پروژه اضافه کنید به این منظور در پنجره ویژوال‌بیسـیک و در پنجره پروژه کلیک راسـت کنید سـپس به ترتیب روی گزینـه Add و بعد روی گزینه Module کلیک کنید (شـکل ۷–۱۲). کادرمحاوره Add Module مطابق شـکل ۷–۱۳ نمایش داده می‌شود.

۲– در کادر محـاوره Add Module، روی دکمـه Open کلیـک کنید تا یک ماژول کد به پروژه power اضافه شـود. اکنون به پنجـره خصوصیات بروید و خصوصیت Name ماژول کد را روی مقدار mdlpower تنظیم کنید.

شکل ۱۲–۷

شکل ۱۳–۷

۳ – در پنجره پروژه روی آیکن ماژول کد کلیک راست کنیدوگزینهSave ModuleAs... را انتخاب کنید، سپس ماژول کد را با نام power ذخیره کنید. همان‌طور که در کادر محاوره ذخیره‌سازی ماژول مشاهده می‌کنید، پسوند فایل‌های ماژول کد BAS است.

۴ – در ایـــن مرحله به مـــاژول فرم frmpower بروید و رویه‌های expense و payment را به ماژول کد mdlpower انتقال دهید.

۵ – بـــه ماژول فرم frmview بروید و عبـــارت frmpower. را از ابتدای نام رویه expense حذف کنید. توجه داشته باشید اگر رویه‌ای در ماژول کد به صورت عمومی تعریف شود، تمام ماژول‌ها می‌توانند آن را فراخوانی کنند و نیازی به اســـتفاده از نام ماژول کد در زمان فراخوانی رویه نیست.

نکته در صورتی که یک متغیر در یک ماژول کد به‌صورت عمومی تعریف شـــود، در تمام رویه قابل شناسایی و استفاده است

۶ – تغییـــرات را ذخیـــره کرده و پروژه را اجرا کنید و با همـــان مقادیر مثال ۵ آزمایش کنید.

۷ – در پایان روی دکمه Exit کلیک کنید تا اجرای پروژه خاتمه یابد، ســـپس به پنجره ویژوال‌بیسیک بازگردید.

۷-۷ رویه‌های رویداد (Event Procedures)

تاکنون از این‌گونه رویه‌ها بارها استفاده کرده‌اید و همان‌طورکه می‌دانید یکی دیگر از ویژگی‌های زبان برنامه‌نویسی ویژوال‌بیسیک پشتیبانی از رویدادهاست. این نوع از رویه‌ها در زمان ایجاد یک شئ مانند کنترل یا فرم همراه با آن‌ها به طور هم‌زمان به وجود می‌آیند و زمانی که رویداد مربوط به یک شئ رخ می‌دهد، رویه رویداد و دستورات موجود در آن به طور خودکار اجرا می‌شوند. شکل کلی یک رویه رویداد به این صورت است:

```
Private  Sub  نام شئ _ نام رویداد (آرگومان‌های رویداد)
        دستور(ات)
End  Sub
```

شئ می‌تواند یک کنترل یا یک فرم باشد. در رویه‌های رویداد کنترل‌ها، نام شئ، مقدار خصوصیت Name کنترل و در رویه‌های رویداد فرم‌ها، از کلمه کلیدی Form به عنوان نام شئ استفاده می‌شود.

در بعضی از رویه‌های رویداد بخش دیگری به نام آرگومان‌های رویداد نیز وجود دارند. آرگومان‌ها در واقع, متغیرهایی هستند که با توجه به نوع رویداد، اطلاعاتی را به داخل رویه منتقل می‌کنند تا برنامه‌نویس با استفاده از آن‌ها مقاصد خود را در رویداد پیگیری کند. رویه‌های رویداد مربوط به ماوس و صفحه کلید از این دسته رویدادها هستند که از آرگومان‌ها استفاده می‌کنند.

البته تاکنون از رویه‌های رویدادی مانند Click، DblClick، Load، Unload و نظایر آن‌ها در رابطه با فرم‌ها و کنترل‌ها استفاده کرده‌اید. در مباحث آینده نیز نحوه استفاده از رویه‌های رویداد دیگری را فرامی‌گیرید. در این‌جا لازم است به ارایه توضیحاتی در رابطه با مهم‌ترین رویدادهای فرم‌ها و کنترل‌ها بپردازیم.

۷-۷-۱ رویه رویداد Activate

این رویه رویداد مربوط به فرم‌هاست و زمانی اجرا می‌شود که فرم، فوکوس می‌گیرد. وقتی پروژه‌ای از چند فرم تشکیل شده باشد و فوکوس از یک پنجره به پنجره دیگر

برنامه منتقل شـود، رویداد Activate مربوط به پنجره‌ای که فوکوس را در اختیار گرفته اسـت، اجرا خواهد شـد. لازم به ذکر اسـت که این رویه رویداد فاقد هر گونه آرگومان می‌باشد. این رویداد پس از رویداد Load اجرا می‌شود.

نکته رویـداد Activate درزمـان انتقـال فوکوس از پنجـره یک‌برنامه به پنجره برنامه دیگر اجرا نخواهد شد.

۷–۷–۲ رویه رویداد Click

این رویه نیز فاقد آرگومان اسـت و زمانی اجرا می‌شـود که کاربر روی فرمی که فعال است یا کنترل موجود در آن کلیک کند.

۷–۷–۳ رویه رویداد Deactivate

ایـن رویـداد نیز فاقد آرگومان و مخصوص فرم‌هاسـت و عملکردی شـبیه به رویداد Activate دارد، با این تفاوت که زمان اجرای آن هنگامی است که فرم، فوکوس خود را بین پنجره‌های یک برنامه از دست می‌دهد و اطلاعات فرم در حافظه باقی می‌ماند.

۷–۷–۴ رویه رویداد DblClick

این رویه رویداد در فرم‌ها و بسـیاری از کنترل‌هـا وجود دارد و فاقد هرگونه آرگومان اسـت. این رویداد زمانی رخ می‌دهد که کاربر روی فرمی که فعال اسـت یا کنترل موجود در آن، عمل دابل‌کلیک انجام دهد.

۷–۷–۵ رویه رویداد Load

این رویه نیز مربوط به فرم‌هاسـت و زمانی اجرا می‌شـود کـه اطلاعات مربوط به فرم وارد حافظه اصلی شـده و فرم، روی دسـک‌تاپ نمایش داده شود. در صورتی‌که بخواهید دستوراتی را در زمان اجرای برنامه انجام دهید، از این رویداد استفاده کنید.

۷–۷–۶ رویه رویداد Unload

این رویداد نیز مخصوص فرم‌هاست و فاقد آرگومان می‌باشد. زمانی که فرم را می‌بندید

یـا به عبارت دیگر اطلاعـات فرم از حافظه کامپیوتر خارج شـود، ایـن رویداد اجرا می‌شود. در صورتی‌که بخواهید دستوراتی را در زمان خاتمه برنامه یا بسته شدن فرم اجرا کنید، از این رویه استفاده کنید.

۷-۷-۷ رویه رویداد GotFocus

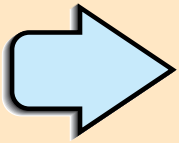
این رویداد نیز بین فرم‌ها و کنترل‌ها مشـترک اسـت و زمانی که فرم یا کنترل فوکوس را در اختیار می‌گیرند، اجرا می‌شود. در هنگامی که یک فرم، فوکوس را به دست می‌آورد ابتدا رویداد Activate و سپس رویداد GotFocus اجرا می‌شود.

نکته در صورتی‌که در یک فرم رویداد Activate و GotFocus به‌طور هم‌زمان مورد استفاده قرار گیرند فقط رویداد Activate اجرا خواهد شد.

در صورتی‌که در یک فرم کنترلی وجود داشـته باشد که بتواند فوکوس بگیرد (مانند کنترلText Box) رویداد GotFocus اجرا نمی‌شود.

۸-۷-۷ رویه رویداد LostFocus

این رویداد نیز بین فرم‌ها و کنترل‌ها مشـترک اسـت و زمانی که فرم و کنترل، فوکوس خود را از دسـت می‌دهند، اجرا می‌شود. هنگامی که یک فرم، فوکوس را از دست می‌دهد ابتدا رویداد LostFocus و سپس رویداد Deactivate اجرا می‌شود.



Learn in English

Introduction to Procedures

You can simplify programming tasks by breaking programs into smaller logical components. These components called procedures can then become building blocks that let you enhance and extend Visual Basic. Procedures are useful for condensing repeated or shared tasks, such as frequently used calculations, text and control manipulation, and database operations. There are several types of procedures used in Visual Basic:

- Sub procedures do not return a value.
- Function Procedures return a value.
- Event Procedure.
- Visual Basic Procedure

واژه‌نامه

Component	جزء
Condense	خلاصه‌کردن
Enhance	افزودن
Extend	توسعه‌دادن
Frequently	تکراری
Introduction	معرفی، آشنایی
Manipulation	دستکاری‌کردن
Private	خصوصی
Procedure	رویه
Public	عمومی
Scope	میدان دید

خلاصه مطالب

- در برنامه‌نویسی به روش ساخت یافته، برنامه به بخش‌های کوچک‌تر به‌نام رویه تقسیم می‌شود.
- رویه‌ها در ویژوال‌بیسیک به چهار گروه رویه‌های فرعی و تابعی، رویه‌های رویداد و رویه‌های آماده ویژوال‌بیسیک تقسیم می‌شوند.
- رویه‌های رویداد در زمان ایجاد یک شئ ایجاد می‌شوند و در زمانی‌که رویداد مربوطه رخ می‌دهد به‌طور خودکار اجرا می‌شوند.
- رویه‌های فرعی و تابعی را می‌توانید با تایپ در ماژول فرم یا با استفاده از کادرمحاوره Add Procedure تعریف کنید.
- برای فراخوانی رویه‌های فرعی از دستور Call استفاده می‌شود.
- برای تعریف یک رویه محلی کلمه کلیدی Private را در ابتدای تعریف رویه قرار دهید.
- یک رویه محلی فقط در همان ماژولی که تعریف می‌شود قابل شناسایی و فراخوانی است.
- برای تعریف یک رویه عمومی، کلمه کلیدی Public را در ابتدای تعریف رویه قرار دهید.
- یک رویه عمومی در تمام ماژول‌های موجود در پروژه، قابل شناسایی و فراخوانی است.
- برای فراخوانی رویه‌ها از دو روش فراخوانی با مقدار یا فراخوانی با مرجع استفاده می‌شود.
- در فراخوانی رویه‌ها با مقدار یک کپی از داده‌ها در آرگومان قرار داده می‌شود.
- در فراخوانی رویه‌ها با مرجع تغییر در مقدار آرگومان‌های رویه سبب تغییر مقدار متغیرهای ارسالی می‌شود.
- در هنگام فراخوانی رویه‌ها می‌توان از نام آرگومان‌ها استفاده کرد.
- برای خروج از یک رویه فرعی و تابعی به ترتیب از دستورات Exit Sub و Exit FunCtion استفاده می‌شود.

آزمون نظری

۱ – کدام رویداد هنگامی که یک فرم، فوکوس خود را از دست می‌دهد، اجرا می‌شود؟

الف– Activate ب– Deactivate ج– Load د– Unload

۲ – کدام گزینه در رابطه با تفاوت رویه‌های فرعی و رویه‌های تابعی درست است؟

الف– ارسال متغیرها به رویه ب– بازگشت یک مقدار به محل فراخوانی

ج– تعریف آرگومان‌ها د– محل تعریف رویه

۳ – پس از فراخوانی رویه تابعی myname به‌صورت «Call myname («Visual», «Basic») مقدار بازگشتی چیست؟

```
Public Function myname (ByRef strfname As String, ByVal strlname As String ) As String
  Dim strname As String
  strname = strfname + strlname
  myname = strname
End Function
```

الف– «Visual» ب– «Visual Basic» ج– «Basic» د– «VisualBasic»

۴ – برای تعریف یک رویه عمومی از کلمه کلیدی ..... استفاده می‌شود.

الف– Private ب– Public ج– Dim د– Sub

۵ – استفاده از کدام گزینه برای تعریف یک رویه فرعی مناسب است؟

الف– Sub . . . End Sub ب– Function . . . End Function

ج– Private د– Public

۶ – پسوند فایل‌های ماژول کد در ویژوال‌بیسیک چیست؟

الف– VBC ب– BAS ج– CLM د– FBC

۷ – درصورتی‌که در یک رویه تابعی، نوع داده بازگشتی تعیین نشود از چه نوعی به طور پیش‌فرض استفاده می‌شود؟

الف– Boolean ب– Integer ج– Byte د– String

۸ – به‌طور پیش‌فرض نحوه ارسال متغیرها به یک رویه از چه نوعی است؟

الف– ارسال با مقدار ب– ارسال با مرجع ج– محلی د– استاتیک

۹ – در صورت تعریف یک رویه با کلمه کلیدی Private در ماژول کد.....

الف– رویه مربوطه در تمام ماژول‌های کد قابل فراخوانی است.

ب- رویه مربوطه در تمام ماژول‌ها قابل فراخوانی است.
ج- رویه مربوطه در تمام ماژول‌های فرم قابل فراخوانی است.
د- رویه مربوطه فقط در همان ماژول کدی که تعریف شده است قابل فراخوانی است.

۱۰ - پس از فراخوانی تابع test به‌صورت test (x As Long) مقدار بازگشتی چند است؟ (با فرض این‌که x=۳ باشد.)

```
Public Function test (ByVal   y  As Long) As Long
  test = y * y * y
End Function
```

الف- ۳ ب- ۹ ج- ۲۷ د- ۷۱

۱۱ - نوع مقدار بازگشتی در رویه تابعی زیر چیست؟

```
Private Function Compute (into As Integer) As Boolean
```

الف- منطقی ب- عدد صحیح ج- رشته د- عدد اعشاری

۱۲ - عملکرد رویه فرعی Subchange چیست؟

```
Public Sub Subchange (ByRef m As String, ByRef n As String)
Dim strchange As String
strchange = m
m = n
n = strchange
End sub
```

الف- محتویات m و n را در رویه فرعی عوض می‌کند.
ب- محتویات m را در n قرار می‌دهد.
ج- محتویات m و n را در محل فراخوانی عوض می‌کند.
د- محتویات m و n را در رویه فرعی و هم در محل فراخوانی عوض می‌کند.

13 - Which of the following procedures can return a value?

a- Sub Procedure b- VB Procedure
c- Functions d- Event Procedure

۱۴- انواع رویه‌ها در ویژوال‌بیسیک را نام برده و هر یک را به طور خلاصه شرح دهید.
۱۵- تفاوت بین رویه‌های فرعی و تابعی را بیان کنید.
۱۶- انواع روش‌های فراخوانی رویه‌ها را توضیح دهید
۱۷- فرق بین رویه‌های عمومی و محلی را بیان کنید.
۱۸- نحوه ایجاد و استفاده از یک ماژول کد را توضیح دهید.

آزمون عملی

۱ – پروژه‌ای طراحی کنید که دمای یک اتاق را براســاس فارنهایت دریافت کرده و با استفاده از یک رویه تابعی مقدار آن‌را به سلسیوس محاسبه کرده و نمایش دهد.

$$c = \frac{۵}{۹}(F - ۳۲)$$

۲ – پروژه‌ای طراحی کنید که یک عدد را دریافت کرده و با استفاده از یک رویه تابعی بخش‌پذیری آن‌را بر اعداد ۳ و ۷ مشخص نماید.

۳ – پروژه‌ای طراحی کنید که یک عدد به عنوان مقدار زمان بر اساس ثانیه دریافت کرده و با استفاده از یک رویه فرعی مقدار ساعت آن‌را محاسبه نموده و نمایش دهد.

۴ – پروژه‌ای طراحی کنید که با اســتفاده از یک رویه تابعی زوج یا فرد بودن هر عدد دلخواهی را مشخص کند.

۵ – پروژه‌ای طراحی کنید که موجودی یک مشتری بانک را براساس ریال دریافت کند و با استفاده از یک رویه فرعی آن‌را به واحد یورو تبدیل کرده و نمایش دهد.

(۱۷۰۰ ریال = ۱ یورو)

واحد کار هشتم

هدف جزیی

توانایی استفاده از انواع رویه‌های آماده در ویژوال‌بیسیک

هدف‌های رفتاری

پس از مطالعه این واحد کار از فراگیر انتظار می‌رود که:

۱- بتواند با توابع رشته‌ای مانند Asc، Chr، InStr، Left، Right، Len، Mid، Replace، Space، StrComp، و String کارکند.

۲- بتواند باتوابع تاریخ و ساعت مانند ، Date، Day، Month، MonthName،Weekday ، Second، Minute ، Hour،Year، WeekdayName و Timer کار کند.

۳- بتواند از کنترل‌های کادر علامت (CheckBox)، دکمه انتخاب (OptionButton)، کادر لیست (ListBox) و کادر لیست ترکیبی (ComboBox) استفاده کند و خصوصیات، رویدادها و متدهای آن‌ها را توضیح دهد.

کلیات

در ویژوال‌بیسیک رویه‌های متعددی وجود دارند که به صورت آماده همراه زبان برنامه‌نویسی ویژوال‌بیسیک در اختیار شما قرار می‌گیرند. این رویه‌ها می‌توانند از نوع رویه‌های فرعی یا رویه‌های تابعی باشند که با توجه به نیازهای روزمره برنامه‌نویسان گردآوری شده‌اند تا کار برنامه‌نویسی آسان‌تر شود و زمان لازم برای طراحی و خطازدایی دستورات کاهش پیدا کند. این رویه‌های آماده می‌توانند دارای یک یا چند آرگومان یا فاقد هر گونه آرگومان باشند، بعضی از آن‌ها مقداری را به عنوان نتیجه محاسبات بازگشت داده و بعضی هیچ مقداری را بازگشت نمی‌دهند. در صورت استفاده از این رویه‌ها ویژوال‌بیسیک در زمان اجرای برنامه، دستورات مربوط به رویه مورد نظر را در اختیار برنامه قرار می‌دهد. تاکنون با بعضی از این رویه‌ها مانند MsgBox, InputBox, Val, Str و غیره آشنا شده‌اید. در این‌جا به معرفی بعضی از رویه‌های موجود در ویژوال‌بیسیک می‌پردازیم.

۱–۸ توابع رشته‌ای ویژوال‌بیسیک

۱–۱–۸ تابع Asc

این تابع یک مقدار رشته‌ای را دریافت می‌کند و کداسکی (ASCII) اولین کاراکتر آن را به‌دست آورده، برمی‌گرداند. مقادیری که این تابع بازمی‌گرداند مطابق جدول کدهای اسکی یعنی صفر تا ۲۵۵ می‌باشد. شکل کلی نحوه استفاده از این تابع به صورت زیر است:

Asc (string)

string یک عبارت رشته‌ای است که می‌تواند یک ثابت یا متغیر رشته‌ای یا مقدار بازگشتی یک تابع که از نوع رشته‌ای است، باشد. به عنوان مثال به مواردی که در جدول ۱–۸ ارایه شده است، توجه کنید:

جدول ۱–۸

مثال	خروجی
Print Asc ("Computer")	۶۷
Print Asc ("Hard Disk")	۷۲
Print Asc ("D")	۶۸
Print Asc ("1")	۴۹
Print Asc ("123")	۴۹

مثال ۱: یک رویه تابعی بنویسید تا در پروژه‌هایی که از کادر متن برای دریافت مقادیر عددی استفاده می‌شود با استفاده از آن بتوان اعدادی را که در کادر متن وارد می‌شوند و در اولین کاراکتر آن‌ها از کاراکترهای غیررقمی استفاده شده است، تشخیص داده و از تبدیل این مقادیر به عدد جلوگیری به عمل آید.

همان‌گونه که در مثال‌های قبل مشاهده کردید گاهی اوقات لازم است که مقادیر عددی را از کادرهای متن به صورت یک رشته دریافت کنید و سپس آن‌ها را با استفاده از تابع Val به مقادیر عددی تبدیل کنید، اما در صورتی که یک مقدار رشته‌ای با کاراکترهای غیررقمی شروع شود، تابع Val مقدار صفر و در صورت استفاده از کاراکترهای غیررقمی در سایر مکان‌ها بجز اولین کاراکتر، تابع Val تا رسیدن به اولین کاراکتر غیر رقمی، رشته را به عدد تبدیل می‌کند. برای جلوگیری از این مسأله می‌توانید از تابع Asc استفاده کنید. بنابراین تابع checkinput را به صورت زیر طراحی کنید:

```
Private Function checkinput(ByVal strno As String) As Boolean
If  Asc(strno) >=  Asc ("0") And  Asc(strno) <= Asc ("9") Then
   checkinput  = True
Else
   checkinput = False
End   If
End   Function
```

این تابع دارای یک آرگومان ورودی است که رشته strno را دریافت می‌کند و رشته مورد نظر با روش فراخوانی با مقدار در این متغیر کپی می‌شود، سپس با استفاده از تابع Asc کداسکی اولین کاراکتر از رشته strno بررسی می‌شود. اگر این مقدار بین ۴۸ تا ۵۷ باشد نشان دهنده این است که اولین کاراکتر از نوع رقمی است (محدوده کاراکترهای صفر تا ۹ مقادیر اسکی ۴۸ تا ۵۷ است) و در نتیجه مقدار منطقی True را بازمی‌گرداند؛ ولی اگر رشته با کاراکتر غیررقمی شروع شود، مقدار False بازگردانده خواهد شد، به این صورت می‌توانید با بررسی مقدار بازگشتی در رویه فراخوان از مقدار وارد شده در کادر متن مورد نظر مطلع شوید.

نکته اگر مقدار ارسالی به تابع Asc یک رشته خالی باشد ("") در زمان اجرای برنامه خطا رخ داده و پیام Invalid procedure call or argument نمایش داده می‌شود.

تمرین:

رویه قبل را به‌گونه‌ای تنظیم کنید تا در صورت ارسال یک رشته خالی در کادر متن از بروز خطا به وسیله تابع Asc جلوگیری شود.

۲–۱–۸ تابع Chr

عملکرد تابع Chr برعکس تابع Asc است. این تابع نیز یک آرگومان از نوع عددی دارد که کد اسکی کاراکتر مورد نظر را تعیین می‌کند و با توجه به کداسکی, کاراکتر مربوط به آن را به عنوان مقدار بازگشتی باز می‌گرداند. شکل کلی نحوه استفاده از این تابع به صورت زیر است:

Chr (code)

آرگومان code یک عدد صحیح بین صفر تا ۲۵۵ است. به عنوان مثال به مواردی که در جدول ۲–۸ ارایه شده است، توجه کنید.

جدول ۲–۸

خروجی	مثال
a	Print Chr (97)
!	Print Chr (33)
t	Print Chr (116)

مثال ۲: یک رویه فرعی بنویسید تا با استفاده از دو حلقه تودرتو شکل زیر را ایجاد کرده و امکان تعیین نوع کاراکتر را نیز داشته باشد.

```
$
$$
$$$
$$$$
$$$$$
```

برای انجام این کار رویه printchar را به صورت زیر بنویسید:

```
Private Sub printchar (ByVal intcode As Integer)
Dim inti , intj As Integer
For inti = 1 To 5
        For intj = 1 To inti
                Print Chr(intcode);
                Next intj
                Print
          Next inti
End Sub
```

همان‌طور که در این رویه فرعی مشاهده می‌کنید، یک آرگومان با نام intcode وجود دارد که از نوع عدد صحیح است و کداسکی کاراکتر مورد نظر را برای نمایش شکل تعیین می‌کند. پس از فراخوانی این رویه کداسکی مورد نظر در آرگومان intcode ذخیره می‌شود، سپس با استفاده از دو حلقه For و تابع Chr شکل مورد نظر با کاراکتر تعیین شده ایجاد می‌شود. حلقه اول در رویه، کنترل سطرها را به عهده دارد و حلقه دوم در داخل آن با توجه به مقدار شمارنده حلقه اول (inti) کاراکترها را در هر سطر نمایش می‌دهد. در واقع با هر بار اجرای حلقه اول, حلقه دوم به اندازه مقدار متغیر inti تکرار خواهد شد. وجود کاراکتر; در انتهای اولین دستور Print نیز سبب می‌شود تا کاراکترها در هر سطر در کنار هم نمایش داده شوند و دستور Print دوم باعث می‌شود تا پس از خاتمه نمایش کاراکترها در یک سطر, کاراکترهای بعدی در سطر جدید نمایش داده شوند و به این صورت شکل مورد نظر روی فرم ایجاد می‌شود. برای فراخوانی این رویه می‌توانید از دستور زیر استفاده کنید:

```
Call printchar (36)
```

۸-۱-۳ تابع Len

این تابع می‌تواند یک رشته را دریافت کند و تعداد کاراکترهای آن‌را به صورت یک عدد صحیح محاسبه نماید. این تابع یک آرگومان از نوع رشته‌ای دارد که وظیفه انتقال رشته را به رویه به عهده دارد. شکل کلی نحوه استفاده از این تابع به صورت زیر است که در آن string یک عبارت رشته‌ای است.

```
Len(string)
```

به عنوان مثال دستور زیر مقدار ۱۲ را نمایش خواهد داد:

```
Print   Len("Microsoft XP")
```

نکته تابع Len برای داده‌های نوع تاریخ نیز مانند رشته‌ها تعداد کاراکترها را مشخص می‌کند. تابع Len برای داده‌های نوع منطقی عدد ۲ را بازگشت می‌دهد.

۸-۱-۴ تابع Left

یکی دیگر از توابع رشته‌ای است که می‌تواند تعداد معینی از کاراکترها را از یک عبارت رشته‌ای و از سمت چپ آن جدا کرده و بازگرداند. شکل کلی نحوه استفاده از این

تابع به صورت زیر است:

Left (string,length)

آرگومان string یک عبارت رشته‌ای است و رشته موردنظر را دریافت می‌کند. آرگومان length نیز یک عبارت از نوع عددی اســت. این تابع به تعداد آرگومان length، کاراکترها را از ســمت چپ رشته string بازمی‌گرداند. نوع مقدار بازگشتی این تابع از نوع رشته‌ای می‌باشد. به عنوان مثال به مواردی که در جدول ۳–۸ ارایه شده است، توجه کنید.

جدول ۳-۸

مثال	خروجی
Print Left ("Microsoft Windows",2)	Mi
Print Left ("Microsoft Windows",5)	Micro
Print Left ("Microsoft Windows",17)	Microsoft Windows

مثال۳: یک رویه فرعی بنویســید تا شــماره دانشجویی یک دانشجوی دلخواه را دریافت کرده و سال ورودی را نمایش دهد.

(با فرض این‌که شماره دانشجویی یک کد ۷ رقمی است که دو رقم اول سال و دو رقم دوم ماه ورودی را به دانشــگاه مشــخص می‌کند.) به عنوان مثال کد ۸۸۰۷۲۴۵ مشخص می‌کند دانشجو در سال ۸۸ و ماه مهر به دانشگاه وارد شده است.

```
Private  Sub  yearentry (lngid As Long)
  Dim intyear As Integer
    intyear = Left (lngid, 2)
  intyear = intyear + 1300
  Print intyear
End Sub
```

در این رویه آرگومان lngid شماره دانشجویی را دریافت می‌کند سپس با استفاده از تابع Left، ۲ رقم سمت چپ آن‌را که تعیین‌کننده سال ورود دانشجو است از آرگومان lngid جدا کرده و در متغیر intyear ذخیره می‌نماید. در مرحله بعد عدد ۱۳۰۰به مقدار قبلی در متغیر intyear اضافه می‌شود تا سال به صورت ۴ رقمی به وسیله دستور Print نمایش داده شود.

۸-۱-۵ تابع Right

عملکرد این تابع مشابه تابع Left است با این تفاوت که این تابع تعداد کاراکترهای موردنظر را از سمت راست رشته‌ای که دریافت می‌کند، بازگشت می‌دهد.

شکل کلی نحوه استفاده از این تابع به صورت زیر است:

Right (string,length)

آرگومان string یک عبارت رشته‌ای است و رشته موردنظر را دریافت می‌کند. آرگومان length نیز یک عبارت از نوع عددی است که تعداد کاراکترهایی را که از سمت راست رشته string جدا می‌شوند، معین می‌کند. نوع مقدار بازگشتی این تابع از نوع رشته‌ای است.

مثال ۴: رویه فرعی yearentry را به‌گونه‌ای تغییر دهید که علاوه بر سال، ماه ورود دانشجو به دانشگاه را نیز نمایش دهد. به این منظور می‌توانید دستورات زیر را به رویه فرعی yearentry اضافه کنید.

```
Dim intmonth As Integer
  intmonth = Left (lngid, 4)
  intmonth = Right (intmonth, 2)
Print intmonth
```

در این دستورات پس از تعریف متغیر موردنیاز با استفاده از تابع Left چهار رقم سمت چپ شماره دانشجو در متغیر intmonth ذخیره می‌شود سپس با استفاده از تابع Right دو رقم سمت راست ارقام جدا شده، به‌دست می‌آید که ماه را مشخص می‌کنند و به‌وسیله یک دستور Print نمایش داده می‌شوند.

۸-۱-۶ تابع Mid

این تابع می‌تواند تعداد معینی کاراکتر را از یک رشته جدا کند و دارای یک آرگومان اختیاری و دو آرگومان اجباری است. شکل کلی نحوه استفاده از این تابع به این صورت است:

Mid (string , start , length)

تابع Mid با توجه به تعدادی که آرگومان length تعیین می‌کند، کاراکترهایی را از رشته string بازمی‌گرداند. آرگومان start محلی را مشخص می‌کند که عمل جدا کردن کاراکترها از آن‌جا شروع می‌شود. اگر مقدار این آرگومان بزرگ‌تر از تعداد کاراکترهای رشته string باشد، رشته‌ای با طول صفر بازگشت داده می‌شود. آرگومان start از نوع عددی Long و

آرگومان length از نوع عددی Long می‌باشند و استفاده از آن اختیاری است. اگر از این آرگومان استفاده نشود تابع Mid کاراکترهایی را که از آرگومان start شروع می‌شوند تا انتهای رشته بازگشت خواهد داد. به عنوان مثال به مواردی که در جدول ۴–۸ ارایه شده است، توجه کنید.

جدول ۴-۸

مثال	خروجی
Print Mid ("Microsoft Windows",1,2)	Mi
Print Mid ("Microsoft Windows",6,4)	soft
Print Mid ("Microsoft Windows",3)	crosoft Windows
Print Mid ("Microsoft Windows",30)	-

مثال ۵: یک رویه فرعی با نام uppercase با تابع Mid بنویسید. تا یک متن را دریافت کند و حروف کوچک آن‌را به حروف بزرگ تبدیل نماید. برای این کار رویه مزبور را به صورت زیر تغییر دهید:

```
Private   Sub uppercase(ByVal strtext As  String)
Dim  strchar  As String * 1, strresult As String
Dim  intlength As Long, inti As Long
intlength = Len(strtext)
For  inti  =  1  To  intlength
     strchar  = Mid(strtext, inti, 1)
     If   Asc(strchar)  >=  Asc ("a")  And  Asc(strchar) <= Asc ("z")  Then
          strresult = strresult + Chr(Asc(strchar) - 32)
     Else
          strresult = strresult + strchar
     End   If
Next   inti
txttext.Text = strresult
End   Sub
```

در این رویه به جای تابع Left از تابع Mid استفاده شده است تا کاراکترها یکی یکی جدا و بررسی شوند. لازم به ذکر است که در این تابع و در آرگومان دوم از متغیر inti استفاده

شـــده که شـــمارنده حلقه اســـت و برای آرگومان ســـوم (که تعداد کاراکترهایی را که جدا می‌شـــوند، معین می‌کند) مقدار ۱ منظور شده است. بنابراین در هر بار اجرای حلقه موقعیت مربوط به کاراکتری که جدا می‌شـــود یک واحد زیاد خواهد شـــد و چون تعداد یک کاراکتر برای جدا شدن انتخاب شده است هر بار یک کاراکتر از رشته، مورد بررسی قرار خواهد گرفت؛ این امر باعث می‌شود تا استفاده از تابع Right نیز الزامی نباشد و بتوان این خط از رویه را نیز حذف کرد.

تمرین:

پـــروژه‌ای بـــا نـــام editor و از نـــوع Standard EXE، مطابق شـــکل ۱-۸ ایجاد کنید که با استفاده از دو رویه مشـــابه رویـــه uppercase و یک کادر متن، واژه‌پردازی طراحی شود که کاربر بتواند در صورت نیاز با استفاده از دکمه‌های فرمان، محتویات آن‌را بـــه حروف کوچـــک یا بـــزرگ تبدیل کند.

شکل ۱-۸

۸-۱-۷ تابع Replace

این تابع می‌تواند رشته‌ای را داخل رشته دیگر پیدا کند و آن را با رشته سوم جایگزین کرده و به صورت یک رشـــته جدید بازگرداند. شـــکل کلی نحوه اســـتفاده از این تابع به صورت زیر است:

Replace(string,find,replace,start,count,compare)

این تابع شـــش آرگومان دارد، ســـه آرگومان اول اجباری و سه آرگومان دوم اختیاری هستند. تابع، رشته find را در داخل رشته string جستجو کرده و رشته replace را به جای آن در رشـــته string جایگزین می‌کند. آرگومان start از نوع عددی بوده و موقعیت شروع جستجو را در رشته string تعیین می‌کند و اگر از آن استفاده نشود، مقدار ۱ در نظر گرفته می‌شـــود. آرگومان count نیز از نوع عددی است و تعداد دفعات جستجو و جایگزینی را تعیین می‌کند. اگر از این آرگومان استفاده نشود به طور پیش‌فرض مقدار ۱- در نظر گرفته می‌شود که عمل جستجو و جایگزینی را تا رسیدن به انتهای رشته string انجام خواهد داد.

آخرین آرگومان compare است که اگر مقدار آن برابر یک (vbTextCompare) باشد، تابع Replace در جستجوی رشته find در string بین حروف کوچک و بزرگ تفاوت قائل نمی‌شود، اما اگر مقدار آن برابر صفر(vbBinaryCompare) باشد بین حروف کوچک و بزرگ تفاوت قائل می‌شود. در این‌جا ذکر این نکته ضروری است که تابع Replace هیچ تغییری در رشته string ایجاد نمی‌کند و حاصل عملیات جستجو و جایگزینی به صورت یک رشته مستقل بازگشت داده خواهد شد. به عنوان مثال به مواردی که در جدول ۵–۸ ارایه شده است، توجه کنید.

جدول ۵–۸

مثال	خروجی
Print Replace ("ABC ABC ABC","A","123")	123BC 123BC 123BC
Print Replace ("ABC ABC ABC","A","123",3)	C123BC 123BC
Print Replace ("ABC ABC ABC","A","123",2)	123BC 123BC ABC
Print Replace ("ABC aBC ABC","A","123")	123BC aBC 123BC
Print Replace ("ABC aBC ABC","A","123",1)	123BC 123BC 123BC
Print Replace ("ABC ABC ABC","B","")	AC AC AC
Print Replace ("ABC ABC ABC","D","123")	ABC ABC ABC

نکته اگر مقدار آرگومان Replace یک رشته خالی ("") باشد تمام مواردی که در رشته string پیدا می‌شوند، حذف خواهند شد.

اگر مقدار آرگومان start بزرگ‌تر از یک باشد و رشته find قبل از موقعیت تعیین شده (یعنی آرگومان start) در رشته string وجود داشته باشد، این بخش‌ها حذف می‌شوند.

تمرین:

واژه‌پرداز editor را مطابق شکل ۲–۸ به گونه‌ای تنظیم کنید تا کاربر توانایی پیدا کردن و جایگزین کردن عبارات را در متن تایپ شده داشته باشد.

شکل ۲–۸

۸-۱-۸ تابع StrComp

از این تابع برای مقایسه دو رشته با یکدیگر استفاده می‌شود. اگر دو رشته با هم مساوی باشند عدد صفر را بازگشت خواهد داد و در غیر این صورت اگر رشته اول بزرگ‌تر از رشته دوم باشد عدد ۱ و در صورتی که رشته اول کوچک‌تر از رشته دوم باشد عدد ۱- بازگردانده خواهد شد. شکل کلی نحوه استفاده از این تابع به صورت زیر است:

StrComp (string1,string2,compare)

آرگومان‌های string1 و string2 از نوع رشته‌ای هستند و شامل دو عبارتی هستند که با هم مقایسه می‌شوند. آرگومان compare از نوع عددی بوده و استفاده از آن اختیاری است و در صورتی که مقدار آن برابر با ۱ (vbTextCompare) باشد تابع StrComp بین حروف کوچک و بزرگ تفاوت قائل نمی‌شود ولی اگر مقدار آن برابر صفر (vbBinaryCompare) باشد بین حروف کوچک و بزرگ تفاوت قائل می‌شود. به عنوان مثال به مواردی که در جدول ۶-۸ ارایه شده است، توجه کنید.

جدول ۶-۸

مثال	خروجی
Print StrComp("basic","Basic",1)	0
Print StrComp("basic","Basic",0)	1
Print StrComp("Basic","basic")	-1

نکته در صورت عدم استفاده از آرگومان Compare به‌طور پیش‌فرض مقدار صفر در نظر گرفته می‌شود.

مثال ۶: به رویه تابعی بعد توجه کنید. این رویه دو رشته را دریافت کرده و معین می‌کند که آیا دو رشته مساوی هستند یا خیر.

```
Private Function strcompare(strstring1 As String, strstring2 As String) As Boolean
    If StrComp(strstring1, strstring2, vbTextCompare) = 0 Then
        strcompare = True
    Else
        strcompare = False
    End If
End Function
```

این تابع دو آرگومان رشته‌ای دارد که عبارت‌های رشته‌ای برای انجام مقایسه را دریافت می‌کند و با استفاده از تابع StrComp آن‌ها را مورد مقایسه قرار می‌دهد و در صورتی که دو رشته برابر باشند، مقدار صفر بازگشت داده می‌شود و این مقدار در یک دستور If بررسی شده و در نتیجه دستور پس از Then اجرا می‌شود و مقدار منطقی True را به عنوان مقدار بازگشتی بازمی‌گرداند. اما در صورت عدم تساوی دو رشته، تابع، مقدار منطقی False را بازگشت خواهد داد. البته در آرگومان سوم تابع StrComp نیز از مقدار vbText- Compare به‌جای عدد یک استفاده شده است که سبب می‌شود تابع بین حروف کوچک و بزرگ تفاوتی قائل نشود. بنابراین اگر این تابع به صورت("HARD","hard ")strcompare فراخوانی شود مقدار True و اگر به‌صورت("Ram" و"Disk") strcompare فراخوانی شود مقدار False را بازگشت خواهد داد.

تمرین:
برای پروژه editor یک کادر محاوره ورود به نرم‌افزار ایجاد کنید که دارای یک کلمه رمز و کلمه کاربر باشد، به علاوه برای بررسی درستی کلمه رمز و کاربر از تابع StrComp استفاده کنید.

نکته در صورتی که آرگومان char یک رشته باشد، فقط اولین کاراکتر تکرار می‌شود.

۹–۱–۸ تابع InStr

این تابع می‌تواند دو رشته را دریافت کرده و موقعیت یکی را در دیگری پیدا کند. این تابع دارای چهار آرگومان است و شکل کلی نحوه استفاده از آن به صورت زیر است:

InStr (start, string1 , string2, compare)

این تابع دارای دو آرگومان رشته‌ای string1 و string۲ است که رشته string2 را در string۱ جستجو می‌کند و در صورتی که نتیجه جستجو مثبت باشد موقعیت آن را به صورت یک عدد بازمی‌گرداند و در غیر این صورت مقدار صفر را برمی‌گرداند. آرگومان start از نوع عددی بوده و استفاده از آن اختیاری است این آرگومان می‌تواند نقطه شروع جستجوی string2 را در string1 تعیین کند و در صورتی که از آن استفاده نشود، موقعیت جستجو از اولین کاراکتر آغاز می‌شود. آرگومان چهارم compare است که مانند آرگومان

قبلی اختیاری است و می‌توانید از آن استفاده نکنید، اگر مقدار این آرگومان یک (vbTextCompare) باشد تابع در هنگام جستجو بین حروف کوچک و بزرگ تفاوت قائل نمی‌شود، اما اگر مقدار این آرگومان صفر (vbBinaryCompare) باشد یا از آن استفاده نشود، در زمان جستجو بین حروف کوچک و بزرگ تفاوت قائل خواهد شد.

به عنوان مثال دستور(InStr (1,"Visual Basic","i") رشته "i" را در رشته "Visual Basic"جستجو می‌کند و عدد ۲ را باز می‌گرداند، اما دستور(InStr (5, "Visual Basic" , "i") مقدار ۱۱ را باز می‌گرداند. چون جستجو از کاراکتر پنجم آغاز می‌شود، بنابراین موقعیت رشته "i" در کلمه Basic بازگشت داده می‌شود. برای مثال می‌توانید به مواردی که در جدول ۷-۸ ارایه شده است، توجه کنید.

جدول ۷-۸

مثال	نتیجه
Print InStr (1, "cpu CPU CPU" , "CPU")	۵
Print InStr ("CPU CPU CPU" , "cpu")	۰
Print InStr (1, "cpu CPU CPU" , "cpu" ,0)	۱
Print InStr (4, "CPU CPU CPU" , "cpu" , 1)	۵
Print InStr (1, "CPU CPU CPU" , "")	۱
Print InStr (3, "CPU CPU CPU" , " ")	۴

نکته در توابعی که از آرگومان compare برای تنظیم نحوه مقایسه استفاده می‌شود، اگر از این آرگومان استفاده نشود و دستور Option Compare Text در بخش تعاریف ماژول فرم نوشته شود، تابع بین حروف کوچک و بزرگ تفاوت قائل نخواهد شد.

در توابعی که آرگومان compare به کار می‌رود می‌توانید به‌جای مقادیر صفر و یک به ترتیب از ثابت‌های vbBinaryCompare و vbTextCompare استفاده کنید.

تمرین:

یک رویه تابعی بنویسید تا یک متن را دریافت کند و اولین جمله آن‌را بازگرداند.

۸–۲ توابع تاریخ و ساعت

ویژوال‌بیسیک علاوه بر رویه‌های آماده‌ای که در رابطه با داده‌های عددی و رشته‌ای دارد، دارای رویه‌هایی برای انجام عملیات روی داده‌های تاریخ و ساعت نیز می‌باشد که به معرفی آن‌ها می‌پردازیم. اما ذکر یک نکته در این جا لازم است و آن این که این رویه‌ها با توجه به تنظیمات منطقه‌ای که در برنامه Regional and Language Options وجود دارد، نتیجه‌های مختلفی را در برخواهند داشت. برای هماهنگی با مثال‌های کتاب توصیه می‌شود که از تنظیمات پیش‌فرض ویندوز استفاده کنید و در صورت تغییر در این تنظیمات, تنظیمات ناحیه‌ای را روی حالت انگلیسی (English) قرار دهید.

۸–۲–۱ تابع Date

این تابع، تاریخ سیستم را در اختیار شما قرار می‌دهد و فاقد هر گونه آرگومان است به علاوه با استفاده از این تابع همراه با دستور Date می‌توانید تاریخ سیستم را نیز تنظیم کنید. برای تنظیم تاریخ سیستم با دستور Date از این الگو استفاده کنید:

```
Date = date
```

مقدار date می‌تواند از نوع‌داده رشته‌ای، تاریخ یا مقدار بازگشتی تابع Date باشد. به عنوان مثال موارد ارایه شده در جدول ۸–۸ نحوه استفاده از دستور Date را نشان می‌دهند.

جدول ۸–۸ دستور Date

Date = "2-20-1970"
Date = # 12/7/ 2003#
Date = "Aug 18,2004"

مثال ۷: پروژه‌ای طراحی کنید که کاربر بتواند تاریخ سیستم را به وسیله آن تنظیم کند. برای این کار عملیات زیر را به ترتیب انجام دهید:

۱ – برنامه ویژوال‌بیسیک را اجرا کنید و یک پروژه از نوع Standard EXE به همراه یک فرم مطابق شکل ۸–۳ و جداول ۸–۹ الی ۸–۱۱ ایجاد کنید.

جدول ۸–۹ خصوصیات فرم

خصوصیت	مقدار
Name	frmset
Caption	Set Date

۸ – ۳

جدول ۸-۱۰ خصوصیات کنترل‌ها

کنترل / خصوصیت	Label	Label	Label	TextBox	TextBox
Name	lbls۱	lbls۲	lbls۳	txtmonth	txtday
Caption	/	/	mm dd yyyy	_	__

جدول ۸-۱۱ خصوصیات کنترل‌ها

کنترل / خصوصیت	TextBox	Command Button	Command Button
Name	txtyear	cmdok	cmdcancel
Caption	_	&OK	&Cancel

در این مرحله رویداد Click دکمه OK را به صورت زیر تنظیم کنید:

```
Private Sub cmdok_Click()
  If (Val(txtday.Text) > 0 And Val(txtday.Text) < =31) And _
  (Val(txtmonth.Text)> 0 And Val(txtmonth.Text)< =12)And _
  (Val(txtyear.Text)>1900 And Val(txtyear.Text)<=3000)Then
        Date = Trim(txtmonth.Text) + String (1, "/") + _
        Trim(txtday.Text)+String(1,"/")+Trim (txtyear.Text)
  Else
     MsgBox "Invalid Date !",  ,"ERROR"
  End If
End Sub
```

در این رویداد با استفاده از یک دستور If مقادیر تایپ شده در کادرهای متن روز (txtday)، ماه (txtmonth) و سال (txtyear) بررسی می‌شود در صورتی‌که مقادیر اشتباهی وارد شود، پیام خطایی در یک کادر پیام نمایش داده می‌شود اما در صورت مناسب بودن مقادیر ورودی با استفاده از عملگر +، توابع Trim و تابع String، داده‌های ورودی به شکل رشته‌ای حاوی تاریخ تنظیم می‌شوند و در پایان، این مقدار رشته‌ای به تابع Date داده می‌شود تا تاریخ سیستم را تنظیم کند.

دستور Unload Me را در رویداد Click دکمه Cancel بنویسید، سپس پروژه و فرم را با نام setdate ذخیره کنید.

پروژه را اجرا کرده و مقادیر ۱۰، ۲۵ و ۲۰۰۵را به ترتیب برای ماه، روز و سال تایپ کنید، سپس روی دکمه OK کلیک کنید.

در ویندوز به کادر محاوره ویژگی‌های تاریخ و ساعت بروید و تاریخ ثبت شده را بررسی کنید.

این‌بار مقادیری را به صورت غلط تایپ کنید و روی دکمهOK کلیک کنید. همان‌طور که می‌بینید پیام خطایی روی دسک‌تاپ مشاهده می‌شود.

به اجرای پروژه خاتمه داده و به پنجره ویژوال‌بیسیک بازگردید.

۲–۲–۸ تابع Day

تابع Day یک مقدار از نوع تاریخ را دریافت کرده و یک عدد بین ۱ تا ۳۱ به عنوان شماره روز بازمی‌گرداند. شکل کلی نحوه استفاده از این تابع به صورت زیر است:

Day (date)

این تابع یک آرگومان دارد که می‌تواند از نوع تاریخی, رشته‌ای یا عددی باشد که بیانگر تاریخ مشخصی است. به عنوان مثال به موارد ارایه شده در جدول ۱۲–۸ توجه کنید.

در مثال چهارم جدول ۱۲–۸ از یک مقدار عددی از نوع صحیح استفاده شده است؛ این مقدار با توجه به روزهایی که از ابتدای سال سپری می‌شود، مورد بررسی قرار خواهد گرفت و در نتیجه مقدار روزها از ابتدای سال تا ماه جاری از عدد مربوطه کم می‌شود تا مقدار بازگشتی بین مقدار ۱ تا ۳۱ قرار گیرد.

جدول ۸-۱۲

مثال	نتیجه
Print Day (# 1/5/1980 #)	۵
0Print Day ("march 15,198 ")	۱۵
Print Day ("1/25/1980 ")	۲۵
Print Day (50)	۱۸

۸-۲-۳ تابع Now

این تابع می‌تواند در هر لحظه تاریخ و ساعت جاری سیستم را در اختیار شما قرار دهد. مقدار بازگشتی این تابع از نوع تاریخ می‌باشد و فاقد آرگومان است. شکل کلی نحوه استفاده از آن به این صورت است:

Now

به عنوان مثال نتیجه فراخوانی این تابع می‌تواند یک عبارت رشته‌ای به صورت # 12/10/2004 6:43:11P m # باشد.

۸-۲-۴ تابع Month

تابع Month با دریافت یک مقدار از نوع تاریخ، عددی بین صفر تا ۱۲ را که بیانگر شماره ماه تاریخ مربوط است، باز می‌گرداند. شکل کلی نحوه استفاده از این تابع به صورت بعد است:

Month (date)

آرگومان date یک عبارت عددی یا رشته‌ای است که بیانگر یک مقدار تاریخی می‌باشد. برای مثال به موارد ارایه شده در جدول ۸-۱۳ توجه کنید.

جدول ۸-۱۳

مثال	نتیجه
Print Month (" 8/10/2000 ")	۸
Print Month (" 11/23/1980 ")	۱۱
Print Month (" March 2,1960 ")	۳
Print Month (# 9-1-99 #)	۹
Print Month (# 10-30-1974 #)	۱۰

۸–۲–۵ تابع MonthName

این تابع می‌تواند با دریافت شماره ترتیب ماه‌های سال، نام آن‌ها را به صورت یک عبارت رشته‌ای بازگرداند. این تابع دارای دو آرگومان است و شکل کلی نحوه استفاده از آن به صورت زیر است:

MonthName (month , abbreviate)

آرگومان اول این تابع month است که از نوع عددی بوده و شماره ماه‌های سال را از ۱ تا ۱۲ دریافت می‌کند. آرگومان دوم abbreviate اختیاری و از نوع منطقی است، اگر این آرگومان True باشد نام ماه به صورت خلاصه و در صورتی که False باشد نام ماه به صورت کامل بازگردانده می‌شود. در صورت عدم استفاده از این آرگومان مقدار پیش‌فرض False است. به عنوان مثال به موارد ارایه شده در جدول ۸–۱۴ توجه کنید.

جدول ۸–۱۴

مثال	نتیجه
Print MonthName (40 False)	April
Print MonthName (40True)	Apr
Print MonthName (5)	May

۸–۲–۶ تابع Weekday

این تابع یک عبارت رشته‌ای یا تاریخی را دریافت کرده و یک عدد صحیح که بیانگر شماره روزهای هفته است، باز می‌گرداند. شکل کلی نحوه استفاده از این تابع به صورت زیر است:

Weekday (date, firstday)

این تابع دارای دو آرگومان است. آرگومان date از نوع تاریخی یا رشته‌ای است که بیانگر یک تاریخ می‌باشد و آرگومان firstday یک آرگومان اختیاری است که روز اول هفته را برای محاسبه شماره روز تعیین می‌کند. مقدار پیش‌فرض برای این آرگومان روز یکشنبه است. این آرگومان می‌تواند یکی از مقادیر ارایه شده در جدول ۸–۱۵ باشد. این تابع عدد صحیحی بین ۱ تا ۷ را به ترتیب برای روزهای یکشنبه، دوشنبه تا جمعه بازمی‌گرداند. به عنوان مثال به موارد ارایه شده در جدول ۸–۱۶ توجه کنید.

جدول ۸-۱۵

روز هفته	ثابت عددی	ثابت رشته‌ای
یکشنبه	۱	vbSunday
دوشنبه	۲	vbMonday
سه‌شنبه	۳	vbTuesday
چهارشنبه	۴	vbWednesday
پنج‌شنبه	۵	vbThursday
جمعه	۶	vbFriday
شنبه	۷	vbSaturday

جدول ۸-۱۶

مثال	نتیجه
Print Weekday ("March 14, 1970")	۷
Print Weekday (#1/15/1960#)	۶
Print Weekday (#1/15/1960#, 5)	۲
Print Weekday (#1/15/1960#, vbFriday)	۱
Print Weekday ("11/20/1980")	۵

۸-۲-۷ تابع Year

این تابع یک عبارت رشته‌ای، تاریخی، عددی یا ترکیبی (Variant) را که حاوی داده‌ای از نوع تاریخ است، دریافت کرده و مقدار سال را به صورت یک عدد صحیح بازگشت می‌دهد. شکل کلی نحوه استفاده از این تابع به صورت زیر است:

Year (date)

آرگومان date به یک‌مقدار از نوع تاریخ اشاره می‌کند. به عنوان مثال به موارد ارایه شده در جدول ۸-۱۷ توجه کنید.

جدول ۸-۱۷

مثال	نتیجه
Print Year ("Aug 22, 69")	۱۹۶۹
Print Year ("11/22/1980")	۱۹۸۰
Print Year (#10/9/2000#)	۲۰۰۰
Print Year ("3-2-95")	۱۹۹۵

۸-۲-۸ تابع Time

این تابع ساعت سیستم را در اختیار شما قرار می‌دهد و فاقد هرگونه آرگومان است. به علاوه با استفاده از این تابع همراه با دستور Time می‌توانید ساعت سیستم را نیز تنظیم کنید. برای تنظیم ساعت سیستم با دستور Time از الگوی زیر استفاده کنید:

Time = time

time یک عبارت از نوع رشته‌ای، ساعت یا مقدار بازگشتی تابع Time است. به عنوان مثال موارد ارایه شده در جدول ۸-۱۸ نحوه استفاده از دستور Time را نشان می‌دهند.

جدول ۸-۱۸ دستور Time

Time = "17:2:30"
Time = "5:17"
Time ="8:15 PM "
Time =# 8:15:20 AM #

در جدول فوق دستورات اول و دوم بدون استفاده از عبارات AM و PM زمان سیستم را تنظیم می‌کنند. در چنین مواقعی با توجه به مقدار ساعت تعیین شده، زمان سیستم روی قبل یا بعدازظهر تنظیم می‌شود. در مثال اول چون ساعت ۱۷ ذکر شده است، زمان روی ۵ بعدازظهر و در مثال دوم چون ساعت ۵ ذکر شده است، زمان روی ۵ صبح تنظیم خواهد شد.

مثال ۸: پروژه‌ای طراحی کنید که کاربر بتواند ساعت سیستم را به‌وسیله آن تنظیم کند. برای این‌کار عملیات زیر را به ترتیب انجام دهید:

۱ – یک پروژه از نوع Standard EXE ایجاد کنید که شامل فرمی مطابق شکل ۴–۸ و جداول ۱۹–۸ الی ۲۱–۸ باشد.

جدول ۱۹–۸ خصوصیات فرم

خصوصیت	مقدار
Name	frmset
Caption	Set Time

۸ – ۴

جدول ۲۰–۸ خصوصیات کنترل‌ها

کنترل / خصوصیت	Label	Label	Label	Label	TextBox
Name	lblh	lblm	lbls	lblampm	txthour
Caption	Hour :	Minute :	Second :	AM/PM	_

جدول ۲۱–۸ خصوصیات کنترل‌ها

کنترل / خصوصیت	TextBox	TextBox	TextBox	Command Button	Command Button
Name	txtminute	txtsecond	txtampm	cmdset	cmdexit
Caption	_	_	_	&Set	&Exit

۲ – رویداد دکمه Set را به صورت زیر تنظیم کنید:

```
Private Sub cmdset_Click()
        Dim strtime As String
        strtime = Trim(txthour.Text)+":"+Trim(txtminute.Text)+ _
                ":"+Trim(txtsecond.Text)+" "+Trim(txtampm.Text)
        Time =strtime
End Sub
```

ابتدا با استفاده از خصوصیت Text کادرهای متن مقدار ساعت، دقیقه، ثانیه و قبل یا بعد از ظهر زمان موردنظر به صورت یک رشته در متغیر strtime ذخیره می‌شود و سپس این مقدار با استفاده از تابع Time به عنوان زمان سیستم ثبت می‌شود.

۳ – پروژه و فرم را با نام settime ذخیره کنید، سپس آن‌را اجرا نمایید.

۴ – مقادیر ۱۰ ، ۲۵، ۱۸ و AM را به ترتیب برای ساعت، دقیقه، ثانیه و به عنوان زمان قبل از ظهر درکادرهای متن مربوطه تایپ کنید و روی دکمه Set کلیک کنید. سپس تغییرات را در نوار وظیفه ویندوز بررسی کنید.

۵ – به اجرای پروژه خاتمه داده و به پنجره ویژوال‌بیسیک بازگردید.

تمرین:

با استفاده از تابع Time و کنترل Timer یک زمان‌سنج دیجیتال را طراحی کنید.

۸–۲–۹ تابع Hour

این تابع با دریافت یک مقدار رشته‌ای یا تاریخی شامل زمان، مقدار ساعت را به‌صورت یک عدد صحیح بین صفر و ۲۳ بازمی‌گرداند. شکل کلی نحوه استفاده از این تابع به صورت زیر است:

Hour (time)

آرگومان time می‌تواند از نوع رشته‌ای، تاریخ و زمان یا ترکیبی (Variant) باشد که بیانگر مقادیر زمانی هستند. به عنوان مثال به موارد ارایه شده در جدول ۲۲–۸ توجه کنید.

نکته اگر آرگومان time مقادیر زمانی بزرگ‌تر از مقادیر مجاز برای ساعت، دقیقه یا ثانیه باشد، در هنگام اجرای برنامه پیام خطای Syntax error یا Type mismatch نمایش داده می‌شود.

اگر در آرگومان time مقدار بعدازظهر (PM) یا قبل از ظهر (AM) تعیین نشود، مقدار ساعت با توجه به مقدار زمان در آرگومان time محاسبه می‌شود.

جدول ۲۲-۸

نتیجه	مثال
۸	Print Hour (#8:18:10 AM#)
۲۲	Print Hour (#22:05:00 #)
۲۲	Print Hour (" 10:50:43 PM ")
۱۳	Print Hour (" 13:20 ")
۴	Print Hour (" 4:10 ")
ساعت جاری سیستم	Print Hour (Now)

۸-۲-۱۰ تابع Minute

این تابع با دریافت یک مقدار رشته‌ای یا تاریخی شامل زمان، مقدار دقیقه را به‌صورت یک عدد صحیح بین صفر و ۵۹ بازمی‌گرداند. شکل کلی نحوه استفاده از این تابع به این صورت است:

Minute (time)

آرگومان time می‌تواند از نوع رشته‌ای، تاریخ و زمان یا ترکیبی (Variant) باشد که بیانگر مقادیر زمانی هستند. برای مثال به موارد ارایه شده در جدول ۲۳-۸ توجه کنید.

جدول ۲۳-۸

نتیجه	مثال
۱۸	Print Minute (#8:18:10 AM#)
۵	Print Minute (#22:05:00#)
۵۰	Print Minute (" 10:50:43 PM ")
۲۰	Print Minute (" 13:20")
۱۰	Print Minute (" 4:10:3")
دقیقه با توجه به ساعت جاری سیستم	Print Minute (Now)

نکته اگر مقادیر زمانی مورد استفاده بزرگ‌تر از مقادیر مجاز برای ساعت، دقیقه یا ثانیه باشند، در هنگام اجرای برنامه پیام خطای Syntax error یا Type mismatch مشاهده می‌شود.

۸-۲-۱۱ تابع Second

این تابع نیز با دریافت یک مقدار رشته‌ای یا تاریخی شامل زمان، مقدار ثانیه را به صورت یک عدد صحیح بین صفر تا ۵۹ بازمی‌گرداند. شکل کلی نحوه استفاده از این تابع به صورت زیر است:

Second (time)

آرگومان time می‌تواند از نوع رشته‌ای، تاریخ و زمان یا ترکیبی (Variant) باشد که بیانگر مقادیر زمانی هستند. برای مثال به موارد ارایه شده در جدول ۸-۲۴ توجه کنید.

جدول ۸-۲۴

مثال	نتیجه
Print Second (# 8:18:10 Am #)	۱۰
Print Second (#22:05:00 #)	۰
Print Second (" 10:50:43 PM")	۴۳
Print Second (" 13:20:2")	۲
Print Second (Now)	مقدار ثانیه با توجه به ساعت جاری سیستم

نکته اگر مقادیر زمانی مورد استفاده بزرگ‌تر از مقادیر مجاز برای ساعت، دقیقه یا ثانیه باشند، درهنگام اجرای برنامه پیام خطای Syntax error یا Type mismatch مشاهده می‌شود.

۸-۳ کنترل کادر در لیست

کنترل‌های کادر لیست یکی دیگر از کنترل‌های ویژوال‌بیسیک هستند که به کاربر اجازه می‌دهند تا از بین چند مقدار مختلف یکی را انتخاب کند. به عنوان مثال می‌توان به کادرهای لیستی که در کادرمحاوره تنظیمات ویژگی‌های نمایشی ویندوز وجود دارند اشاره کرد.

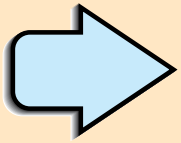
کادرهای لیست انواع مختلفی شامل کادرهای لیست معمولی (ListBox) و کادرهای لیست ترکیبی بازشو (ComboBox) دارند. تفاوتی که بین این دو نوع کنترل علاوه بر شکل ظاهری وجود دارد این است که کادرهای لیست ساده فاقد کادرمتن هستند اما کادرهای لیست ترکیبی بازشو از یک کادر لیست ساده و یک کادرمتن تشکیل می‌شوند.

جدول ۸-۲۵

خاصیت/ متد/رویداد	توضیح
List	این خاصیت اعضای موجود در کادر لیست را نگهداری می‌کند، این خاصیت در تمام انواع کنترل‌های کادر لیست وجود دارد.
Sorted	این خاصیت در تمام کنترل‌های کادر لیست وجود دارد و از نوع منطقی است. اگر مقدار این خاصیت را روی True تنظیم کنید. اعضای موجود در کادر لیست به صورت صعودی مرتب می‌شوند، اما اگر مقدار آن روی False تنظیم شود اعضای موجود در کنترل به همان ترتیبی که در خاصیت List تایپ شده‌اند، نمایش داده می‌شوند.
Style	اگر این خاصیت در کنترل‌های ListBox روی مقدار Standard-۰ تنظیم شود، کنترل به صورت یک کادر لیست ساده و در صورتی که روی مقدارCheckBox-۱ تنظیم شود، هر یک از اعضا در کنترل به همراه یک کادر علامت نمایش داده می‌شوند، این خاصیت می‌تواند در کنترل‌های ComboBox سه مقدار را کسب کند؛ اگر روی مقدارDropdown Combo-۰ تنظیم شود کنترل به صورت یک کادر لیست بازشو به همراه یک کادر متن و اگر روی مقدار Simple Combo List-۱ تنظیم شود، کنترل به صورت یک کادر لیست بازشو ساده به همراه یک کادرمتن و در صورتی که روی مقدارDropdown List-۲ تنظیم شود به صورت یک کادر لیست بازشو و بدون کادر متن نمایش داده می‌شود.
Locked	این خاصیت در کنترل کادر لیست بازشو از نوع Dropdown Combo و Simple Combo قابل استفاده است و عملکرد آن مشابه همین خاصیت در کنترل کادر متن است.
Text	این خاصیت در کنترل کادر لیست ساده و کنترل‌های کادر لیست بازشو بجز نوع Dropdown List قابل استفاده است و عنوان عضو انتخاب شده در کنترل را در خود نگهداری می‌کند.

ادامه جدول ۸-۲۵

خاصیت/ متد/رویداد	توضیح
AddItem	این متد می‌تواند عضوی را به کنترل کادر لیست اضافه کند و شکل کلی نحوه استفاده از آن به صورت بعد است: (عنوان عضو) AddItem . نام کنترل کادر لیست
Clear	به وسیله این متد می‌توان تمام اعضای موجود در کنترل کادر لیست را حذف کرد. شکل کلی نحوه استفاده از این متد به صورت زیر است: Clear . نام کنترل کادر لیست
RemoveItem	این متد می‌تواند هر یک از اعضای موجود در کنترل را براساس شماره اندیس آن‌ها در خاصیت List حذف کند. شماره اندیس از مقدار صفر برای اولین عضو آغاز می‌شود و به ترتیب قرار گرفتن اعضا در لیست افزایش می‌یابد. شکل کلی نحوه استفاده از این متد به این صورت است: شماره‌اندیس عضو در کنترل) RemoveItem . نام‌کنترل‌کادرلیست
Click	این رویداد در تمام انواع کادرهای لیست قابل استفاده است و زمانی اجرا می‌شود که کاربر یکی از اعضای موجود در کادر لیست را انتخاب کند.
Change	این رویداد در کنترل‌های کادر لیست بازشو از نوع Dropdown Combo و Simple Combo قابل استفاده است و زمانی اجرا می‌شود که محتویات کادر متن در کادر لیست تغییر کند.



Learn in English

Len Function: Returns a Long containing the number of characters in a string or the number of bytes required to store a variable.

Syntax

Len(*string* | *varname*)

The **Len** function syntax has these parts:

Part	Description
String	Any valid string expression. If **string** contains **Null**, **Null** is returned.
Varname	Any valid variable name. If varname contains **Null**, **Null** is returned. If *varname* is a **Variant**,**Len** treats it the same as a **String** and always returns the number of characters it contains.

واژه‌نامه

Abbreviate	کوتاه کردن، مختصر کردن
Argument	آرگومان
Binary	دودویی، مبنای دو
Compare	مقایسه کردن
Day	روز
Editor	واژه‌پرداز
Error	خطا
Expression	عبارت
Hour	ساعت
Invalid	اشتباه
Length	طول
Minute	دقیقه
Mismatch	عدم تطابق، مطابقت نداشتن
Month	ماه
Month Name	نام ماه
Remove	رفع‌کردن، برداشتن
Replace	جایگزین کردن
Right	راست
Second	ثانیه
Sort	مرتب‌کردن
Space	فضای خالی
Start	شروع
Syntax	شکل نوشتاری دستور
Syntax	نحو، ترکیب
Time	زمان
Treat	رفتار کردن
Week Day	روز هفته
Week day Name	نام روز هفته
Year	سال

خلاصه مطالب

- تابع ASC یک رشته را دریافت کرده و کداسکی اولین کاراکتر را بر می‌گرداند.
- تابع Chr کداسکی یک کاراکتر را دریافت کرده و کاراکتر معادل آن را بر می‌گرداند.
- تابع InStr دو رشته را دریافت کرده، یکی را در دیگری جستجو می‌کند.
- تابع Left تعداد معینی از کاراکترهای یک رشته را از سمت چپ آن برمی‌گرداند.
- تابع Right تعداد معینی از کاراکترهای یک رشته را از سمت راست آن برمی‌گرداند.
- تابع Len یک رشته را دریافت کرده و طول آن را معین می‌کند.
- تابع Mid یک رشته را دریافت کرده و تعداد معینی کاراکتر را از آن جدا می‌کند.
- تابع Replace رشته‌ای را در رشته دیگری جایگزین می‌کند.
- تابع StrComp دو رشته را دریافت کرده و با هم مقایسه می‌کند.
- با استفاده از تابع Date می‌توان تاریخ سیستم را به دست آورده یا تنظیم کرد.
- تابع Day یک مقدار از نوع تاریخ را دریافت کرده و شماره روز آن را برمی‌گرداند.
- با استفاده از تابع Now می‌توان تاریخ و ساعت سیستم را به دست آورد.
- تابع Month یک مقدار از نوع تاریخ را دریافت کرده و شماره ماه آن را برمی‌گرداند.
- تابع MonthName با دریافت شماره ماه، نام آن را برمی‌گرداند.
- تابع Weekday یک عبارت رشته‌ای یا تاریخی را دریافت کرده و شماره ترتیب روز را در هفته برمی‌گرداند.
- تابع Year یک عبارت رشته‌ای، تاریخی را دریافت کرده و مقدار سال آن را برمی‌گرداند.
- با استفاده از تابع Time می‌توان ساعت سیستم را به دست آورد.
- تابع Hour یک عبارت رشته‌ای یا تاریخی شامل زمان را دریافت کرده و مقدار ساعت را برمی‌گرداند.
- تابع Minute یک عبارت رشته‌ای یا تاریخی شامل زمان را دریافت کرده و مقدار دقیقه را برمی‌گرداند.
- تابع Second یک عبارت رشته‌ای یا تاریخی شامل زمان را دریافت کرده و مقدار ثانیه را برمی‌گرداند.
- با استفاده از تابع Timer می‌توان ثانیه‌های بعد از نیمه شب را محاسبه کرد.

آزمون نظری

۱ – حاصل اجرای دستور"123" Len ,"Windows" Right چیست؟

الف– "۱۲۳" ب– "Win" ج– "ows" د– "ind"

۲ – به‌وسیله کدام تابع می‌توان تعدادی از کاراکترهای یک رشته را جدا کرد؟

الف– Instr ب– Mid ج– Len د– Asc

۳ – حاصل اجرای دستور (3, "Computer") Left چیست؟

الف– "Com " ب– "ter" ج– "puter" د–""

۴ – کدام تابع امکان جستجوی یک رشته را در رشته دیگر فراهم می‌کند؟

الف– Len ب– InStr ج– Replace د– Space

۵ – کدام گزینه در رابطه با خروجی دستور (“10:12:36”) Second درست است؟

الف– ۱۰ ب– ۱۲ ج– ۳۶ د– ۴۸

۶ – حاصل عبارت ("Computer",5,5) Mid چیست؟

الف– omput ب– uter ج– puter د– Computer

۷ – به وسیله کدام تابع می‌توان ساعت و تاریخ جاری سیستم را به دست آورد؟

الف– Date ب– Time

ج– Now د– گزینه‌های الف و ج صحیح هستند.

۸ – حاصل عبارت(Replace("Book","o", "t",3 چیست؟

الف– "Botk" ب– "Bttt" ج– "Bttk" د– "Btok"

۹ – Which of the following data types does return by the Len Function?

a- integer b- string c- date d- boolean

۱۰– تفاوت و شباهت توابع Date و Now را بیان کنید.

۱۱– نحوه عملکرد توابع زیر را توضیح دهید.

الف– Left ب– Mid ج– Strcomp د– Instr

۱۲– نحوه عملکرد توابع زیر را توضیح دهید.

الف– Day ب– Asc ج– Month Name د– Replace

۱۳– نحوه عملکرد توابعYear، Hour وMinute را توضیح دهید.

آزمون عملی

۱ – یک رویه فرعی بنویسید که یک عدد را دریافت کرده، کوچک‌ترین رقم آن‌را پیدا کند و نمایش دهد.

۲ – یک رویه فرعی بنویسید که یک رشته را دریافت کرده و کاراکترهای آن‌را به صورت معکوس نمایش دهد. به‌عنوان مثال‌عبارت Computer را به‌صورت retupmoC نمایش دهد.

واحد کار نهم

هدف جزیی

نحوه استفاده از رویدادهای ماوس و صفحه کلید

هدف‌های رفتاری

پس از مطالعه این واحد کار از فراگیر انتظار می‌رود که:

۱- رویدادهای ماوس و صفحه کلید را بیان کند.

۲- بتواند از رویدادهای ماوس و صفحه کلید استفاده کند.

۳- کنترل‌های شکل (Shape) و خط (Line) را شرح داده، خصوصیات آن‌ها را توضیح دهد.

۴- بتواند از کنترل‌های شکل (Shape) و خط (Line) استفاده کند.

کلیات

یکی از مسایلی که در برنامه‌نویسی برای سیستم‌های عامل با رابط گرافیکی مانند ویندوز مورد توجه قرار می‌گیرد، عکسل‌العمل‌های مناسب در رابطه با اتفاقاتی مانند فشردن دکمه‌های ماوس، حرکت اشاره‌گر آن یا فشردن کلیدها در صفحه کلید است. در زبان ویژوال‌بیسیک رویدادهای مناسب در این زمینه برای برآورده کردن نیازهای برنامه‌نویسان در نظر گرفته شده‌اند.

۹–۱ رویدادهای ماوس

رویدادهای ماوس در ویژوال بیسیک به چهار گروه تقسیم می‌شوند که عبارتند از:Click، DblClick، MouseDown، MouseUp، MouseMove و DragDrop.در این واحد کار رویدادهایی را که در این رابطه در ویژوال بیسیک وجود دارند، فرا می‌گیرید.

۹–۱–۱ رویداد MouseDown

شکل کلی این رویه رویداد به صورت زیر است و زمانی اجرا می‌شود که یکی از دکمه‌های ماوس به پایین فشرده شود.

Private Sub نام شئ_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single)

همان‌طور که مشاهده می‌کنید این رویداد برخلاف رویدادهایی که تاکنون دیده‌اید دکمه‌ای را که کاربر فشرده است،Button دارای چهار آرگومان است. آرگومان اول یعنی مشخص می‌کند. این مقدار می‌تواند یکی از مقادیر موجود در جدول ۹–۱ باشد به این شکل می‌توانید هر دکمه‌ای از ماوس را که فشرده می‌شود، شناسایی کنید.

جدول ۹-۱

توضیح	ثابت عددی	ثابت رشته‌ای
درصورتی‌که دکمه سمت چپ ماوس فشرده شود.	۱	vbLeftButton
درصورتی‌که دکمه سمت راست ماوس فشرده شود.	۲	vbRightButton
درصورتی‌که دکمه وسط ماوس فشرده شود.	۴	vbMiddleButton
هر دو	۳	Right Left

آرگومان Shift معین می‌کند که در هنگام فشرده شدن دکمه‌های ماوس کدام‌یک از کلیدهای Alt، Ctrl، Shift یا ترکیبی از آن‌ها فشرده شده است. مقادیر مربوط به آرگومان Shift در جدول ۲-۹ ارایه شده‌اند.

جدول ۲-۹

توضیح	ثابت عددی	ثابت رشته‌ای
درصورتی‌که کلید Shift فشرده شود.	۱	vbShiftMask
درصورتی‌که کلید Ctrl فشرده شود.	۲	vbCtrlMask
درصورتی‌که کلید Alt فشرده شود.	۴	vbAltMask
درصورتی‌که کلیدهای Shift+Ctrl فشرده شوند.	۳	vbShiftMask+vbCtrlMask
درصورتی‌که کلیدهای Shift+Alt فشرده شوند.	۵	vbShiftMask+vbAltMask
درصورتی‌که کلیدهای Ctrl+Alt فشرده شوند.	۶	vbCtrlMask+vbAltMask
درصورتی‌که کلیدهای Shift+Ctrl+Alt فشرده شوند.	۷	vbShiftMask+vbCtrlMask+ vbAltMask

آرگومان‌های X و Y از نوع عدد اعشاری بوده و موقعیت اشاره‌گر ماوس را در زمان فشرده شدن کلیدهای آن معین می‌کند. مقدار X فاصله از سمت چپ و Y فاصله از بالای فرم یا کنترل مربوطه می‌باشند. نام شئ نیز می‌تواند نام یک کنترل یا عبارت Form برای فرم‌ها باشد.

۹-۱-۲ رویداد MouseUp

شکل کلی این رویه رویداد به صورت زیر است و زمانی اجرا می‌شود که یکی از دکمه‌های ماوس که به پایین فشرده شده است، رها شود.

```
Private Sub نام شئ_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
```

آرگومان‌های این رویداد دقیقاً مشابه رویداد MouseDown است.

۹-۱-۳ رویداد MouseMove

شکل کلی این رویه رویداد به صورت زیر است و زمانی اجرا می‌شود که اشاره‌گر ماوس روی فرم یا کنترل مربوطه حرکت کند.

```
Private Sub نام شئ_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
```

آرگومان‌های این رویداد نیز دقیقاً مشابه دو رویداد قبلی می‌باشند.

مثـال ۱: پروژه‌ای طراحی کنید که تصویری مطابق شـکل ۱-۹ روی یک فرم نمایش دهد و اگر کاربر روی تصویر کلیک کند تصویر با حالت خوشـحال نشان داده شـود و در صورتی‌که کاربر کلیک راست کند تصویـر با حالت ناراحت نشـان داده شود.

شکل ۱-۹

۱ – یک پروژه از نوع Standard EXE ایجاد کرده و یک فرم با نام frmface و یک کنترل تصویر با نام imgface مطابق شکل ۱-۹ ایجاد کنید.

۲ – در کنتـرل تصویر با اسـتفاده از کادرمحـاوره Load Picture تصویر FACE10 را از مسیر

D:\ Program Files\ Microsoft Visual studio\ Common\ Graphics\ Icons\ Misc

انتخاب کنید. (با فرض این‌که برنامه ویژوال استودیو در درایو :D نصب شده باشد.)

۳ – رویداد Mouse Down کنترل imgface را به‌صورت زیر تنظیم کنید.

```
Private Sub imgface_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single)
  If Button = vbLeftButton Then
    imgface.Picture = LoadPicture _
    ("C:\Program Files\Microsoft Visual Studio\Common\Graphics\Icons\Misc\FACE02.ICO")
  If Button = vbRightButton Then
    imgface.Picture = LoadPicture _
    ("C:\Program Files\Microsoft Visual Studio\Common\Graphics\Icons\Misc\FACE04.ICO")
End Sub
```

در این رویداد با استفاده از دستور If مقدار آرگومان Button بررسی می‌شود اگر این مقدار برابر با Vb Left Button باشـد به این معنی اسـت که دکمه سـمت چپ ماوس فشرده شده است بنابرایـن تابع Load Picture تصویر صورت را با حالـت خنده نمایش می‌دهد. به همین

ترتیب اگر دکمه سمت راست ماوس فشرده شود نتیجه بررسی شرط موجود در دستور If دوم درست (True) می‌شود. بنابراین تصویر صورت با حالت ناراحت نمایش داده می‌شود.

۴ – پروژه و فرم را با نام Show Face ذخیره نمایید سپس پروژه را اجرا کنید و روی تصویر یک بار کلیک و بار دیگر کلیک راست کنید و نتیجه را بررسی نمایید.

۵ – اجرای برنامه را متوقف کرده و به پنجره ویژوال‌بیسیک بازگردید.

تمرین:

پروژه ShowFace را به شکلی تنظیم کنید تا پس از رها کردن دکمه سمت چپ یا راست، تصویر اول دوباره نمایش داده شود.

۴–۱–۹ رویداد DragDrop

شکل کلی این رویه رویداد به صورت زیر است و زمانی اجرا می‌شود که یک کنترل در روی فرم با استفاده از اشاره‌گر ماوس از یک محل به محل دیگر منتقل ((Drag & Drop شود.

```
Private Sub نام شئ _ (Source As Control, X As Single, Y As Single) DragDrop
```

آرگومان Source شئ که عملیات انتقال (Drag & Drop) روی آن انجام شده است، تعیین می‌کند. اگر شئ یک کنترل باشد نام کنترل و اگر فرم باشد کلمه Form مورد استفاده قرار می‌گیرد. در رابطه با این رویداد باید توجه داشته باشید که رویداد DragDrop یک کنترل یا فرم زمانی اجرا می‌شود که کنترل دیگری پس از Drag شدن روی آن Dropشود. آرگومان‌های X و Y موقعیت نقطه‌ای را که عمل رها کردن (Drop) در آن‌جا صورت گرفته است، مشخص می‌کنند. اما لازم است بدانید که برای انجام عمل انتقال یک کنترل در روی فرم باید خصوصیت DragMode را برای کنترل موردنظر روی مقدار 1-Automatic تنظیم کنید و به علاوه دستور Source Move x,y را در رویداد DragDrop فرم قرار دهید.

مثال ۲: پروژه Calculator را که در واحدکار پنجم طراحی کرده‌اید به شکلی تغییر دهید تا بتوان با عمل Drag & Drop محتویات کادرمتن txtnum1 را در کادرمتن txtnum2 کپی کرد. برای این کار عملیات زیر را به ترتیب انجام دهید:

۱– پروژه Calculator را باز کنید.

۲ – خصوصیت Drag Mode کنترل کادرمتن txtnum۱ را روی مقدار ۱-Automatic تنظیم کنید تا امکان انجام عمل Drag & Drop روی کنترل کادرمتن امکان‌پذیر باشد.

۳ – رویداد Drag Drop کنترل کادرمتن txtnum۲ را به این صورت تنظیم کنید.

```
Private Sub txtnum2_DragDrop(Source As Control, X As Single, Y As Single)

  txtnum2.Text = txtnum1.Text

End Sub
```

وقتی کنترل txtnum1 در زمان اجرای برنامه با عمل Drag & Drop روی کنترل txtnum2 قرار بگیرد این رویداد اجرا می‌شود و محتویات کادرمتن txtnum۱ را روی txtnum2 ذخیره می‌کند.

۴ – تغییرات را ذخیره کنید و برنامه را اجرا نمایید.

۵ – یک عدد در کادرمتن اول تایپ کنید سپس با عمل Drag & Drop کادرمتن اول را روی کادرمتن دوم بیندازید و نتیجه را بررسی کنید.

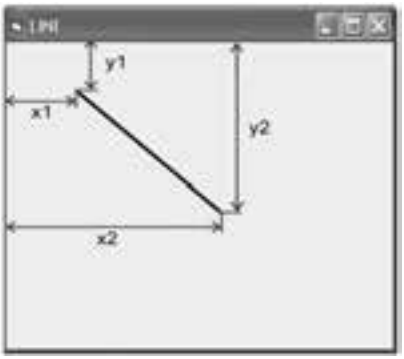
۶ – اجرای برنامه را خاتمه داده و به پنجره ویژوال‌بیسیک بازگردید.

تمرین:

پروژه Calculator را به شکلی تنظیم کنید تا بتوان با عمل Drag & Drop محتویات کادرمتن دوم را نیز در کادرمتن دوم کپی نمود.

۹–۲ کنترل خط (Line)

با استفاده از این کنترل می‌توان انواع خطوط افقی و عمودی و مورب را رسم نمود. در این کنترل خصوصیت‌های x_1 و y_1 مختصات نقطه شروع برای ترسیم خط و خصوصیت‌های x_2 و y_2 مختصات نقطه انتهایی خط را مشخص می‌کنند (شکل ۲–۹). به علاوه با استفاده از خصوصیت BorderColor می‌توان رنگ خط و با خصوصیت BorderWidth ضخامت خط را تعیین نمود.



شکل ۲–۹

این کنترل دارای خصوصیت دیگری به نام BorderStyle نیز می‌باشد که به وسیله آن می‌توان انواع خطوط خط‌چین و خط نقطه و ... را مطابق جدول ۳-۹ مشخص کرد.

جدول ۳-۹

نوع خط	مقدار خصوصیت
خط به‌صورت نامرئی رسم می‌شود.	0- Transparent
———————————	1- Solid
– – – – – – – – – – – – – –	2- Dash
.....	3- Dot
– . – . – . – . – . – . – . –	4- Dash- Dot
– .. – .. – .. – .. – .. – .. –	5- Dash- Dot- Dot
———————————	6- Inside Solid

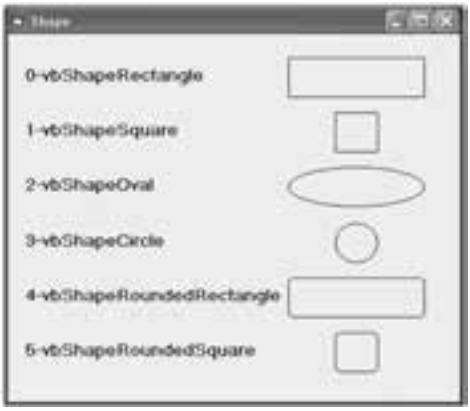
۳-۹ کنترل شکل (Shape)

با این کنترل می‌توانید اشکال مختلف هندسی مثل دایره، بیضی، مربع و غیره را نمایش دهید. خصوصیت‌های این کنترل در جدول ۴-۹ ارائه شده‌اند.

جدول ۴-۹ خصوصیات کنترل Shape

توضیح	خصوصیت
رنگ داخل شکل را مشخص می‌کند.	BackColor
اگر روی مقدار Transparent - ۰ تنظیم شود شکل به‌صورت شفاف نمایش داده می‌شود، اگر روی Opaque - ۰ تنظیم شود شکل با رنگی که در خصوصیت BackColor تعیین شده است نمایش داده می‌شود.	BackStyle
رنگ کادر دور شکل را مشخص می‌کند.	BorderColor
نوع کادر دور شکل را مشخص می‌کند.	BorderStyle
ضخامت کادر دور شکل را مشخص می‌کند.	BorderWidth
نوع شکل هندسی را مشخص می‌کند.	Shape

مقادیری که خصوصیت Border-Style می‌تواند کسب کند مشابه مقادیر همین خصوصیت در کنترل Line است، خصوصیت Shape نیز مقادیر مختلفی را کسب می‌کنند که در شکل ۳-۹ نمایش داده شده‌اند.



شکل ۳-۹

نکته واحد اندازه‌گیری پیش‌فرض در ترسیم اشکال هندسی در روی فرم، Twip است. اگر بخواهید این واحد اندازه‌گیری را تغییر دهید، می‌توانید از خصوصیت ScaleMode فرم استفاده کنید. این خصوصیت واحدهای اندازه‌گیری مختلفی مانند Twip، Pixel، Character، Inch و Millimeter و Centimeter را در اختیار شما قرار می‌دهد.

۹-۴ رویدادهای صفحه کلید

در ویژوال‌بیسیک سه رویداد برای کلیدهای صفحه کلید در نظر گرفته شده است این رویدادها شامل رویدادهای KeyDown، KeyUp و KeyPress هستند.

۹-۴-۱ رویداد KeyDown

شکل کلی این رویه رویداد به‌صورت زیر است و زمانی اجرا می‌شود که یکی از کلیدهای صفحه کلید فشرده شوند.

Private Sub نام شئ_ KeyDown(KeyCode As Integer,Shift As Integer)

این رویداد دو آرگومان دارد. آرگومان KeyCode کداسکی کلید فشرده شده را معین می‌کند. در صفحه‌کلید به هر کلید یک عدد صحیح مثبت بین صفر تا ۲۵۵ نسبت داده شده است که با عنوان کداسکی (ASCII) نام‌گذاری شده‌اند.آرگومان Shift مقداری را با توجه به فشرده شدن کلیدهای Ctrl، Alt و Shift یا ترکیبی از آن‌ها را که به‌طور هم‌زمان با

کلید اصلی فشرده شده‌اند در اختیار شما قرار می‌دهد. مقادیر مربوط به این آرگومان‌ها به صورت ضمیمه در انتهای کتاب ارایه شده‌اند. برای کنترل‌ها از نام کنترل و برای فرم‌ها کلمه Form به‌عنوان نام شئ استفاده می‌شود.

۹-۴-۲ رویداد KeyUp

شکل کلی این رویداد نیز به صورت زیر است و زمانی اجرا می‌شود که کلید فشرده شده در صفحه کلید رها می‌شود.

Private Sub نام‌شئ_ KeyUp(KeyCode As Integer,Shift As Integer)

آرگومان‌های این رویداد کاملاً مشابه رویداد KeyDown هستند و نام شئ می‌تواند نام یک کنترل یا عبارت Form برای فرم‌ها باشد.

نکته برای تشخیص کاراکترهایی که با استفاده از کلید Shift تایپ می‌شوند (مانند !، @ و غیره) از آرگومان KeyCodeو Shiftبه‌طور هم‌زمان استفاده کنید.

۹-۴-۳ رویداد KeyPress

شکل کلی این رویه رویداد به‌صورت زیر است و زمانی اجرا می‌شود که یکی از کلیدهای صفحه کلید فشرده شده و رها شود:

Private Sub نام شئ -KeyPress(KeyAscii As Integer)

این رویه یک آرگومان از نوع عدد صحیح دارد که کداسکی کلید فشرده شده را مشخص می‌کند. مقادیر مربوط به این آرگومان به صورت ضمیمه در انتهای کتاب ارایه شده‌اند. نام شئ می‌تواند نام یک کنترل یا عبارت Form برای فرم‌ها باشد.

تفاوتی که بین این رویداد با رویدادهای دیگر صفحه کلید وجود دارد این است که رویداد KeyPress برای حروف بزرگ و کوچک حرفی کدهای متفاوتی ارایه می‌کند، در صورتی که رویدادهای KeyDown و KeyUp کدهای یکسانی را بازگشت می‌دهند. به عنوان مثال رویداد KeyPress برای کاراکتر A کد ۶۵ و برای کاراکتر a کد ۹۷ تولید می‌کند، اما رویداد KeyDown برای هر دو کاراکتر کد ۶۵ را تولید می‌کند. به‌علاوه رویداد KeyPress برای سایر کاراکترها مانند علایم > ، < ، ؟ و غیره کدهایی را مطابق جدول ضمیمه در انتهای کتاب تولید می‌کند.

نکته
- رویداد KeyDown نسبت به KeyPress اولویت دارد و زودتر اجرا می‌شود.
- در رویداد KeyPress کد مربوط به کلیدهای رقمی در بخش عددی صفحه کلید مطابق با کد کلیدهای رقمی در بخش حرفی صفحه کلید است.
- رویداد KeyPress ترکیب کلید Ctrl با کاراکترهای حرفی و علائم[و] را پشتیبانی می‌کند.
- رویداد KeyPress از کلیدهای تابعی، حرکت مکان‌نما و نظیر آن‌ها پشتیبانی نمی‌کند.
- رویداد KeyPress از کلیدهای Enter، Backspace و ESC پشتیبانی می‌کند.

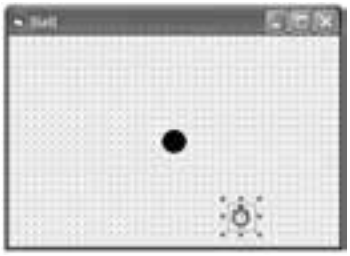
۹-۴-۴ خصوصیت KeyPreview

این خصوصیت یکی از خصوصیات مهم در فرم‌هاست و نحوه اجرای رویه‌های رویداد صفحه کلید را برای فرم و کنترل‌های موجود در روی آن معین می‌کند. این خصوصیت از نوع منطقی می‌باشد و مقدار پیش‌فرض آن مقدار False است. در این حالت اگر کلیدی در صفحه کلید فشرده شود، رویدادهای صفحه کلید مربوط به کنترلی که فوکوس را در اختیار دارد، اجرا می‌شوند. اما اگر مقدار این خصوصیت روی True تنظیم شده باشد ابتدا رویدادهای صفحه کلید فرم اجرا می‌شوند و سپس رویدادهای صفحه کلید مربوط به کنترلی که فوکوس را در اختیار دارد، اجرا می‌شوند. در ضمن اگر هیچ کنترلی وجود نداشته باشد رویدادهای صفحه کلید فرم اجرا می‌شوند.

نکته وقتی فوکوس روی کنترل دکمه فرمان قرار می‌گیرد، کلیدهای Enter و حرکت مکان‌نما نمی‌توانند رویدادهای صفحه‌کلید فرم و کنترل دکمه فرمان را اجرا نمایند.

مثال ۳: یک بازی طراحی کنید که بازیکن، یک توپ متحرک را با استفاده از کلیدهای جهت‌دار در داخل یک قاب کنترل کند و از برخورد آن با دیوارهای این قاب جلوگیری نماید، در صورتی که توپ با دیوارها برخورد کند بازی خاتمه یافته و براساس مدت زمانی که بازیکن توانسته است توپ را کنترل کند، امتیاز وی محاسبه می‌شود. به علاوه کاربر بتواند با فشردن کلید ترکیبی Ctrl+Q هر زمان که مایل باشد از بازی خارج شود و با استفاده از کلید ترکیبی Alt+P حرکت توپ را به طور موقت متوقف کرده و با استفاده از کلید ترکیبی Shift+S توپ را مجدداً به حرکت در آورد.

۱ – برنامه ویژوال بیسیک را اجرا کنید و یک پروژه از نوع Standard EXE به همراه یک فرم با عرض و ارتفاع، ۴۷۸۵ و ۳۷۵۰، مطابق شکل ۴–۹ ایجاد کنید.



شکل ۴ –۹

همان‌طور که مشاهده می‌کنید یک توپ با کنترل شکل (Shape) و یک کنترل زمان‌سنج برای ایجاد حرکت در توپ استفاده شده است، خصوصیات آن‌ها را مطابق جدول ۵–۹ تنظیم کنید.

جدول ۵–۹

کنترل / خصوصیت	Timer	Shape
Name	tmrtimer	shpball
BackColor	——	سیاه
BackStyle	——	1-Opaque
Enabled	False	——
Interval	۲۰	——

۲ – در بخش تعاریف، رویداد KeyDown و Load فرم را به صورت زیر تنظیم کنید:

```
Dim intdirection As Byte

Private Sub Form_KeyDown(KeyCode As Integer,Shift As Integer)

Select Case KeyCode

    Case 13: tmrtimer.Enabled = True

    Case 37: intdirection = 1

    Case 38: intdirection = 2

    Case 39: intdirection = 3

    Case 40: intdirection = 4

End Select

Select Case Shift
```

```
    Case vbCtrlMask:
        If KeyCode=81 Then Unload Me
    Case vbAltMask:
        If KeyCode=80 Then tmrtimer.Enabled=False
    Case vbShiftMask:
        If KeyCode=83 Then tmrtimer.Enabled=True
End Select
End Sub

Private Sub Form_Load()
    intdirection = 1
End Sub
```

همان‌طورکه در رویداد KeyDown فرم مشاهده می‌کنید با استفاده از یک دستور Select Case کلید فشرده شده بررسی می‌شود. اگر مقدار آرگومان KeyCode برابر ۱۳ باشد به معنی فشرده شدن کلید Enter خواهد بود، در این صورت خصوصیت Enabled کنترل زمان‌سنج True می‌شود تا کنترل زمان سنج شروع به کار کند. به این ترتیب فشردن کلیدهای جهت‌دار، کدهای ۳۷ تا ۴۰ را تولید می‌کنند که متناسب با کلید فشرده شده متغیر عمومی intdirection مقداردهی می‌شود. از این متغیر برای ایجاد حرکت متناسب با کلید فشرده شده در رویه رویداد کنترل زمان سنج استفاده می‌شود. این متغیر در رویه رویداد Load فرم مقدار ۱ را به دست می‌آورد. اگر مقدار این متغیر ۱ باشد به معنی حرکت به چپ، مقدار ۲ به معنی حرکت به بالا، مقدار ۳ به معنی حرکت به راست و مقدار ۴ به معنی حرکت به پایین خواهد بود.

سپس برای مدیریت کلیدهای ترکیبی Ctrl+Q، Alt+P و Shift+S از دستور Select استفاده می‌شود و با بررسی مقدار آرگومان Sift ابتدا فشرده شدن کلیدهای Ctrl، Alt و Shift بررسی می‌شود. برای این کار در هر Case از ثابت‌های مربوط به هر یک از این کلیدها استفاده شده است و برای تشخیص کلیدهای Q، P و S همراه کلیدهای Ctrl، Alt و Shift در Case هر یک از آن‌ها یک دستور If قرار داده شده است تا با استفاده از آرگومانKeyCode کلید حرفی نیز شناسایی شود و در صورت درست بودن نتیجه بررسی دستور موردنظر اجرا شود.

۳ – در این مرحله رویداد Timer کنترل زمان‌سنج را به صورت زیر تنظیم کنید:

```
Private Sub tmrtimer_Timer()
  Select Case intdirection
    Case 1:  shpball.Left = shpball.Left - 50
    Case 2:  shpball.Top = shpball.Top - 50
    Case 3:  shpball.Left = shpball.Left + 50
    Case 4:  shpball.Top = shpball.Top + 50
End Select
If shpball.Left <= 18 Or shpball.Left >= 4319 Or _
      shpball.Top <= 22 Or shpball.Top >= 2724 Then
                                  tmrtimer.Enabled = False
   MsgBox "Game Over !!!!", vbCritical, "THE END"
  End If
End Sub
```

می‌بینید که با استفاده از یک دستور Select Case مقدار متغیر intdirection که در رویداد KeyDown با توجه به کلید فشرده شده مقداردهی شده است، کنترل می‌شود و با توجه به این مقدار حرکت مناسب برای توپ در نظر گرفته می‌شود. برای مثال اگر مقدار متغیر intdirection، برابر ۱ باشد به این معنی است که کلید جهت‌دار چپ فشرده شده است، در نتیجه از خصوصیت Left کنترل شکل ۵۰ واحد کم می‌شود که این سبب حرکت توپ به سمت چپ خواهد شد و این کار تا زمانی‌که کاربر کلید دیگری از کلیدهای جهت‌دار را فشار ندهد، هر ۲۰ میلی‌ثانیه یک بار تکرار می‌شود. این امر سبب خواهد شد تا توپ با یک حرکت ملایم به سمت چپ حرکت کند.

به همین صورت وقتی کلید جهت‌دار راست فشرده شود Case سوم اجرا می‌شود و با اضافه کردن ۵۰ واحد به خصوصیت Left توپ، آن را به سمت راست فرم حرکت می‌دهد و اگر کلید جهت‌دار بالا یا پایین را فشار دهد به ترتیب Case دوم یا چهارم اجرا شده و با کاهش یا افزایش خصوصیت Top توپ، آن را به بالا یا پایین حرکت خواهد داد. در ضمن با استفاده از یک دستور If، در هر بار مکان قرارگیری توپ بررسی می‌شود تا در صورت برخورد توپ با دیواره‌های فرم، کنترل زمان سنج از کار بیفتد و پیام Game Over

با استفاده از یک کادر پیام نشان داده شود.

۴ – پروژه و فرم را با نام ball ذخیره کرده، سپس آن را اجرا کنید.

۵ – کلید Enter را فشرده و بلافاصله با کلیدهای جهت‌دار، توپ را کنترل کنید، سپس کلیدها را رها کنید تا توپ با دیواره‌های فرم برخورد کرده و پیام GameOver نمایش داده شود.

۶ – به پنجره ویژوال بیسیک بازگردید و برای آن‌که زمان سپری شده نیز نمایش داده شود یک کنترل زمان سنج دیگر روی فرم قرار دهید، سپس خصوصیت Interval آن را روی ۱۰۰۰ میلی‌ثانیه و خصوصیت Name و Enabled آن را به ترتیب روی tmrclock و False تنظیم کنید.

۷ – اکنون به رویداد KeyDown بروید و در اولین Case دستور Select Case دستور tmrclock.Enabled=True را تایپ کنید تا با شروع به کار زمان سنج اول، زمان سنج دوم نیز شروع به کار کند.

۸ – دستور Dim lngtime As Long را در بخش تعاریف ماژول فرم و دستور lngtime=lngtime+۱ را در رویداد() tmrclock_Timer تایپ کنید.

۹ – به رویداد() tmrtimer_Timer بروید و دستور MsgBox را به صورت بعد تغییر دهید تا در پایان بازی زمان به دست آمده نمایش داده شود.

```
MsgBox «Game Over ! your Time Is: " _
+Str(lngtime)+" Second.",vbCritical,"The End"
```

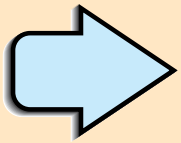
۱۰ – تغییرات ایجاد شده را ذخیره کنید و پروژه را مجدداً اجرا و آزمایش نمایید و در پایان به اجرای پروژه خاتمه داده، به پنجره ویژوال‌بیسیک بازگردید.

تمرین:

بازی ball را طوری تنظیم کنید که امکانات زیر به آن اضافه شود:

الف– امکان افزایش و کاهش سرعت حرکت توپ با کلیدهای + و ـ وجود داشته باشد.

ب– امکان توقف بازی با کلید Pause و شروع مجدد آن با کلید Esc وجود داشته باشد.



Learn in English

Responding to Mouse Events

You can use the MouseDown, MouseUp, and MouseMove events to enable your applications to respond to both the location and the state of the mouse. These mouse events are recognized by most controls.

Event	Description
MouseDown	Occurs when the user presses any mouse button.
MouseUp	Occurs when the user releases any mouse button.
MouseMove	Occurs each time the mouse pointer is moved to a new point on the screen.

واژه‌نامه

خط تیره	Dash
نقطه	Dot
میانی، وسط	Middle
اشاره‌گر	Pointer
تشخیص دادن	Recognize
پاسخ دادن	Responding
شکل	Shape
توپُر	Solid
حالت، وضعیت	State
شفاف	Transparent

خلاصه مطالب

- رویدادهای ماوس عبارتند از: MouseDown، MouseUp، MouseMove و DragDrop
- رویداد MouseDown زمانی اجرا می‌شود که یکی از کلیدهای ماوس فشرده شود.
- رویداد MouseUp زمانی اجرا می‌شود که یکی از کلیدهای ماوس که فشرده شده است، رها شود.
- رویداد MouseMove زمانی اجرا می‌شود که اشاره‌گر ماوس روی فرم یا یک کنترل به حرکت درآید.
- رویداد DragDrop زمانی اجرا می‌شود که موقعیت یک کنترل روی فرم به وسیله ماوس تغییر کند.
- از کنترل Shape برای ایجاد اشکال هندسی و از کنترل Line برای ایجاد خط استفاده می‌شود.
- رویدادهای صفحه کلید عبارتند از: KeyDown، KeyUp و KeyPress.
- رویداد KeyDown و KeyPress زمانی اجرا می‌شوند که یک کلید در صفحه کلید فشرده شود.
- رویداد KeyUp زمانی اجرا می‌شود که یک کلید فشرده شده در صفحه کلید رها شود.
- خصوصیت KeyPreview نحوه اجرای رویدادهای صفحه کلید را بین فرم و کنترل آن تعیین می‌کند.

آزمون نظری

۱-کدام رویداد در صفحه کلید بین کاراکترهای حرفی کوچک و بزرگ تفاوت قائل می‌شود؟

الف- KeyDown ب- KeyPress ج- KeyUp د- KeyPreview

۲ - کدام خصوصیت در کنترل Shape, نوع شکل را معین می‌کند؟

الف- BorderStyle ب- Style ج- Shape د- Appearance

۳ - برای استفاده از رویداد DragDrop تنظیم کدام خصوصیت در کنترل الزامی است؟

الف- DragIcon ب- DragMode ج- DragDrop د- DropMode

۴ - کدام خصوصیت در کنترل Line نوع خط رسم شده را تعیین می‌کند؟

الف- Border Style ب- Border Width ج- Border Color د- Draw Mode

۵ - کدام گزینه در رابطه با رسم یک نقطه در روی فرم مناسب است؟

الف- Point ب- PSet ج- PointSet د- Step

۶ - کدام آرگومان در رویداد DragDrop، کنترل جابه‌جا شده را مشخص می‌کند؟

الف- Control ب- Source ج- Shift د- Ctrl

۷ - واحد اندازه‌گیری پیش‌فرض در کنترل Shape چیست؟

الف- سانتی‌متر ب- اینچ ج- Twip د- Pixel

۸ - در صورتی که خصوصیت KeyPreview روی مقدار True تنظیم شود ..... .

الف- رویدادهای صفحه کلید فرم زودتر از کنترل‌ها اجرا می‌شوند.

ب- رویدادهای صفحه کلید کنترل زودتر از فرم اجرا می‌شوند.

ج- فقط رویدادهای صفحه کلید کنترل‌ها اجرا می‌شوند.

د- فقط رویدادهای صفحه کلید فرم اجرا می‌شوند.

۹ - کدام آرگومان در رویدادKeyUp کداسکی کلید فشرده شده را مشخص می‌کند ؟

الف- KeyCode ب- Key ج- Shift د- Keys

۱۰ - کدام آرگومان در رویداد KeyDown می‌تواند فشرده شدن کلیدهای تغییر حالت مانند Alt را مشخص کند؟

الف- Shift ب- Alt ج- Control د- KeyCode

۱۱ -Which of the following mouse events can be used to respond mouse button clicks?

a- MouseUp b- MouseDown c- MouseMove d- DragDrop

۱۲– تفاوت بین رویدادهای صفحه کلید KeyDown و Keypress را توضیح دهید.

۱۳– رویدادهای ماوس را نام برده و هر یک را همراه با آرگومان‌های مربوطه به اختصار شرح دهید.

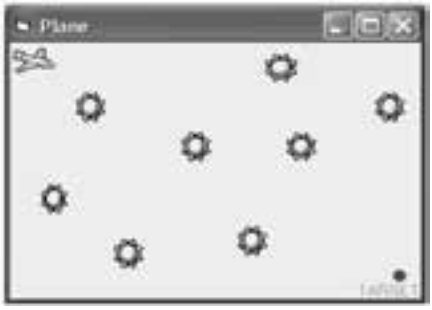
۱۴– نقش و کاربرد آرگومان‌های KeyCode و Shift را در رویدادهای KeyDown و KeyUp بیان کنید.

۱۵– وظیفه آرگومان KeyAscii را در رویداد KeyPress توضیح دهید.

۱۶– رویداد DragDrop را همراه با آرگومان‌های آن توضیح دهید.

۱۷– وظیفه آرگومان‌های Button و Shift را در رویدادهای ماوس مشخص کنید.

آزمون عملی



۱ – یک بازی طراحی کنید که مطابق شکل زیر, بازیکن یک هواپیمای متحرک را از میان موانع و بدون برخورد با آن‌ها عبور داده به مقصد برساند و بر اساس زمان تلف شده امتیاز وی محاسبه شود، به علاوه در صورت برخورد وی با دو مانع ادامه بازی امکان‌پذیر نباشد.

۲ – پروژه‌ای طراحی کنید که مطابق شکل بعد حرکت یک زیردریایی در زیر آب را شبیه‌سازی نماید و علاوه بر امکان حرکت زیردریایی با صفحه کلید، بتوان از پریسکوپ زیردریایی در نزدیکی سطح آب نیز استفاده کرد و در ضمن خورشید و ابرهای موجود در آسمان نیز به صورت متحرک باشند و شرایط طبیعی مثل روز و شب را پدید آورند.

واحد کار دهم

هدف جزیی

نحوه ایجاد انواع منو در ویژوال‌بیسیک

هدف‌های رفتاری

پس از مطالعه این واحد کار از فراگیر انتظار می‌رود که:

۱- انواع منوها را بیان کند.

۲- انواع منوها را ایجاد کند و بتواند از رویدادهای منوها استفاده کند.

۳- بتواند از کنترل‌های نوار پیمایش افقی و عمودی استفاده کرده، رویدادها و خصوصیات آن‌ها را به‌کاربرد.

۴- تعریف رابط گرافیکی MDI و SDI و تفاوت‌های آن‌ها را بیان کند.

۵- بتواند رابط گرافیکی MDI را ایجاد کند.

۶- نحوه ایجاد کادر محاوره باز کردن فایل را بداند.

یکی از مهم‌ترین اجزای تشکیل دهنده برنامه‌هایی که برای سیستم عامل ویندوز تهیه می‌شوند نوارهای منو و گزینه‌های موجود در آن‌هاست. نوارهای منو و گزینه‌های موجود در آن دسترسی کاربر به امکانات برنامه را آسان‌تر کرده و شکل ظاهری مناسبی را به رابط گرافیکی می‌دهند تا کاربر با انتخاب گزینه‌های مورد نظر خود، عملیات مربوطه را انجام دهد. منوها دارای انواع مختلفی هستند، در شکل ۱–۱۰ نمونه‌هایی از آن‌ها را مشاهده می‌کنید.

شکل ۱–۱۰

۱–۱۰ نحوه طراحی انواع منو در ویژوال‌بیسیک

برای ایجاد منوها در فرم‌های خود می‌توانید از ابزار Menu Editor در پنجره ویژوال‌بیسیک استفاده کنید، برای اجرای این برنامه می‌توانید یکی از این روش‌ها را انتخاب کنید:

۱ – در نوار ابزار استاندارد ویژوال‌بیسیک روی دکمه Menu Editor کلیک کنید.

۲ – گزینه Menu Editor را از منوی Tools انتخاب نمایید.

۳ – از کلید ترکیبی Ctrl+E استفاده کنید.

مثال ۱: یک واژه پرداز مطابق شکل‌های ۲–۱۰ الی ۴–۱۰ و جداول ۱–۱۰ و ۲–۱۰ طراحی کنید تا کاربر توانایی انتخاب قالب‌بندی‌های مناسب را برای متن تایپ شده داشته باشد، به این منظور برنامه ویژوال‌بیسیک را اجرا کنید و عملیات بعد را به ترتیب انجام دهید:

شکل ۲–۱۰

شکل ۳–۱۰

شکل ۴–۱۰

جدول ۱–۱۰ خصوصیات فرم

خصوصیت	مقدار
Name	frmeditor
Caption	Text Editor

جدول ۲–۱۰ خصوصیات کنترل

کنترل / خصوصیت	TextBox
Name	txttext
MultiLine	True
ScrollBars	Both-۳

۱ – یک پروژه از نوع Standard EXE ایجاد کنید که حاوی یک فرم با یک کنترل کادر متن مطابق شکل ۵–۱۰ باشد.

شکل ۵ –۱۰

در شکل‌های ۲–۱۰ الی ۴–۱۰ ملاحظه می‌کنید، که نوار منوی فرم دارای دو منوی File و Options است. در منوی File دو گزینه وجود دارد، گزینه New که برای تایپ یک متن جدید در کادر متن استفاده می‌شود و گزینه Exit که برای خروج از برنامه به کار می‌رود.

منوی دوم یعنی Options نیز از گزینه‌های Font Size و Font Color تشکیل شده است که هر یک شامل گزینه‌هایی برای تنظیم اندازه و رنگ قلم هستند با کلیک روی اندازه‌ها یا رنگ‌ها، متن موجود در کادر متن به طور خودکار تنظیم می‌شود.

۲ – برای ایجاد منو در روی فرم، در نوار ابزار استاندارد ویژوال‌بیسیک روی دکمه Menu Editor کلیک کنید تا کادر محاوره Menu Editor نمایش داده شود. برای فعال کردن این کادرمحاوره می‌توانید گزینه ...Menu Editor را از منوی Tools برگزینید یا از کلید ترکیبی Ctrl+E استفاده کنید.

۳ – برای ایجاد اولین منو، در کادر متن عنوان منو (Caption) عبارت &File را تایپ کنید، کاراکتر & سبب می‌شود کلید ترکیبی Alt+F برای باز کردن منو در نظر گرفته شود، به این نوع از کلید ترکیبی، کلید دسترسی سریع یا Hot Key گفته می‌شود. سپس در کادر متن نام منو (Name) عبارت mnufile را تایپ کنید. منوها و گزینه‌های موجود در آن‌ها نیز مانند فرم‌ها و کنترل‌ها دارای خصوصیت Name هستند و قوانین نام‌گذاری آن‌ها مانند فرم‌ها و کنترل‌هاست. مطابق شکل ۶–۱۰ عنوان منو در کادر لیست موجود در قسمت پایین کادر محاوره Menu Editor دیده می‌شود.

شکل ۶ –۱۰

۴ – برای ایجاد گزینه New داخل منوی File، ابتدا روی دکمه Next در کادرمحاوره Menu Editor کلیک کنید، سپس روی دکمه ➔ در کادر محاوره کلیک کنید. در کادر لیست عناوین منوها و در زیر عبارت &File، چهار کاراکتر نقطه ظاهر شده‌اند، این عمل سبب می‌شود که ویژوال‌بیسیک گزینه New را در منوی File نمایش دهد (شکل ۷–۱۰).

شکل ۷-۱۰

۵ - عبارات &New و mnunew را به ترتیب در کادرهای متن عنوان و نام کادر محاوره Menu Editor تایپ کنید. مجدداً روی دکمه Next کلیک کنید و به همین صورت گزینه Exit را در منوی File ایجاد کنید. به علاوه با استفاده از دکمه‌های ↑ و ↓ در کادر محاوره Menu Editor می‌توانید گزینه‌ها را جابه‌جا کنید و به وسیله دکمه Delete در کادر محاوره Menu Editor گزینه مورد نظر خود را حذف نمایید.

۶ – روی دکمه OK کلیک کنید و به پنجره طراحی فرم باز گردید. در پنجره طراحی روی منوی File کلیک کنید، می‌بینید منویی را که طراحی کرده‌اید در پنجره طراحی نیز قابل مشاهده است (شکل ۸-۱۰).

۷ – مجدداً به کادرمحاوره Menu Editor بروید، سپس در کادر لیست عناوین منوها روی گزینه &Exit کلیک کرده، بعد روی دکمه Next سپس دکمه ← کلیک کنید.

۸ – در کادرهای متن عنوان و نام منو عبارات &Options و mnuoptions را تایپ کنید.

شکل ۸-۱۰

۹ – مانند گزینه New، گزینه‌های Font Size و Font Color را مطابق شکل ۹-۱۰ در منوی Options بسازید.

۱۰ – در این مرحله گزینه‌های موجود در زیر منوهای، منوی Options را ایجاد کنید. بنابراین در کادر لیست عناوین منوها ابتدا روی گزینه Font &Color و بعد روی دکمه Insert و دکمه ← کلیک کنید.

۱۱ – در کادر متـــن عنـــوان و نام منو به ترتیب عبارات 10 و mnu10 را تایپ کنید و به همین صورت گزینه‌های ۱۲ و ۱۴ را ایجاد کنید.

چـــون می‌خواهید گزینه‌های زیـــر منوی Font Size دارای چک مارک باشـــند و گزینه انتخاب شده در این زیر منو با چک مارک نمایش داده شود، گزینه ۱۲ را در کادر اسامی منوها انتخاب کرده، سپس در کادر علامت Checked در کادر محاوره Menu Editor کلیک کنید (شکل ۹–۱۰). این کادر علامت می‌تواند گزینه‌ها را به همراه یک چک مارک نمایش دهد. از این گزینه‌ها زمانی استفاده می‌شود که کاربر بخواهد از چند گزینه موجود در یک منو یا زیر منو، یکی را انتخاب نماید.

۱۲ – مانند مرحله ۱۱ گزینه‌های Blue، Green و Red را در زیر منوی Font Color ایجاد کنید. در این‌جا می‌خواهید برای این گزینه‌ها کلیدهای ترکیبی تعریف کنید، بنابراین روی گزینه &Blue در کادر لیســـت عناوین منوها کلیک کنید، ســـپس از کادر لیســـت Shortcut، گزینه Ctrl+B را برگزینید و به همین شکل کلید ترکیبی Ctrl+G و Ctrl+R را برای گزینه‌های Green و Red انتخـــاب کنید. در ضمن چون می‌خواهید به طور پیش فرض رنگ متن آبی باشـــد، بنابراین گزینـــه Blue را غیرفعال کنید. به این منظور درکادر لیســـت عناوین منوها گزینـــه &Blue را برگزینید و روی کادر علامت Enabled کلیک کنید (شـــکل ۹–۱۰). این کادر علامـــت اجازه می‌دهد تـــا گزینه‌های مورد نظر خود را فعال یـــا غیر فعال کنید. در صورتی‌که این کادر علامت انتخاب شود گزینه مربوطه قابل استفاده خواهد بود و در غیر این صورت امکان کلیک روی گزینه ممکن نیست، روی دکمه OK کلیک کنید و در پنجره طراحی فرم گزینه‌های منویی را که ایجاد کرده‌اید، بررسی نمایید.

در ضمـــن برای مخفی کردن منوهـــا و گزینه‌های موجود در آن‌هـــا می‌توانید از کادر علامت Visible در کادر محاوره Menu Editor استفاده کنید، در صورتی‌که این کادر علامت از حالت انتخاب خارج شود منو یا گزینه مورد نظر مخفی خواهد شد.

۱۳ – اکنون در منـــوی File یک خط جداکننده بیـــن گزینه‌هـــای New و Exit ایجاد کنید. برای این کار کافی است یک گزینه جدیـــد بین گزینه‌های New و Exit در منوی File ایجـــاد کرده و برای عنوان آن از کاراکتـــر تفریق (–) اســـتفاده کنید (شکل ۹–۱۰).

شکل ۹ –۱۰

۱۴ – در این مرحله رویدادهای مربوط به هر گزینه را تنظیم کنید. برای این کار کافی است در پنجره طراحی فرم روی گزینه مورد نظر کلیک کنید یا در ماژول فرم رویداد Click گزینه‌ها را انتخاب کنید. به پنجره ماژول فرم بروید و رویداد Click گزینه‌های منوی Font Size را به صورت زیر تنظیم کنید:

```
Private Sub mnu10_Click()
   mnu10.Checked = True
   mnu12.Checked = False
   mnu14.Checked = False
   txttext.FontSize = 10
End Sub

Private Sub mnu12_Click()
   mnu10.Checked = False
   mnu12.Checked = True
   mnu14.Checked = False
   txttext.FontSize = 12
End Sub

Private Sub mnu14_Click()
   mnu10.Checked = False
   mnu12.Checked = False
   mnu14.Checked = True
   txttext.FontSize = 14
End Sub
```

از رویداد Click گزینه‌های mnu10، mun12 و mnu14 استفاده شده است. در هر رویداد با توجه به گزینه انتخاب شده خصوصیت Checked آن روی True تنظیم می‌شود، این امر سبب می‌شود چک مارک در کنار گزینه مورد نظر نمایش داده شود. به علاوه خصوصیت Checked دو گزینه دیگر False خواهد شد و با استفاده از خصوصیت FontSize اندازه قلم متن با توجه به گزینه انتخاب شده تنظیم می‌شود.

۱۵–اکنون رویدادهای مربوط به گزینه‌های زیر منوی FontColor را به این صورت تنظیم کنید:

```
Private Sub mnublue_Click()
    mnublue.Enabled = False
    mnugreen.Enabled = True
    mnured.Enabled = True
    txttext.ForeColor = vbBlue
End Sub

Private Sub mnugreen_Click()
    mnublue.Enabled = True
    mnugreen.Enabled = False
    mnured.Enabled = True
    txttext.ForeColor = vbGreen
End Sub

Private Sub mnured_Click()
    mnublue.Enabled = True
    mnugreen.Enabled = True
    mnured.Enabled = False
    txttext.ForeColor = vbRed
End Sub

Private Sub Form_Load()
    Call mnu12_ Click
    Call mnublue_ Click
End Sub
```

در این رویدادها با استفاده از خصوصیت Enabled گزینه‌ها و با توجه به گزینه انتخاب شده، مقدار این خصوصیت روی False تنظیم می‌شود تا گزینه انتخاب شده غیرفعال شود، زیرا با تغییر رنگ قلم به رنگ مربوطه، انتخاب مجدد همان گزینه لازم نیست، اما برای دو گزینه دیگر این خصوصیت روی True تنظیم می‌شود تا گزینه‌ها برای کاربر قابل انتخاب شوند.

به علاوه با استفاده از خصوصیت ForeColor کادر متن، رنگ متن تنظیم می‌شود و برای هماهنگی بیشتر، در رویداد Load فرم رویه رویدادهای mun12_Click و _mnublue Click فراخوانی می‌شوند تا اندازه و رنگ قلم روی مقدار ۱۲ و آبی تنظیم شوند. بنابراین هنگام اجرای برنامه اندازه و رنگ متن با گزینه‌های مربوطه در منو یکسان می‌شوند.

۱۶- پروژه و فرم را با نام texteditor ذخیره کنید، سپس آن را اجرا نمایید و نام و نام خانوادگی خود را در کادر متن بنویسید. مشاهده می‌کنید که رنگ قلم آبی و اندازه آن ۱۲ است.

۱۷- روی گزینه ۱۴ و سپس ۱۰ در منوی Font Size کلیک کنید و نتیجه را بررسی کنید، سپس روی گزینه‌های Green و Red در منوی Font Color کلیک کنید و نتیجه را بررسی کنید.

۱۸ - به اجرای پروژه خاتمه دهید و به پنجره ویژوال‌بیسیک بازگردید.

تمرین:

پروژه فوق را طوری تنظیم کنید که با انتخاب گزینه New محتویات کادر متن حذف شود و با انتخاب گزینه Exit برنامه واژه‌پرداز خاتمه یابد. به علاوه دارای این شرایط باشد:

الف- یک منوی جدید با عنوان Edit به نوار منو اضافه کنید که امکان جستجو و جای‌گزینی عبارت مورد نظر را در متن داشته باشد.

ب- یک گزینه به منوی Options اضافه کنید تا مانند گزینه Font Color بتواند رنگ زمینه متن را تغییر دهد.

۲–۱۰ نحوه ایجاد و استفاده از رابط گرافیکی چند سندی یا MDI [۱]

تاکنون پروژه‌هایی را که طراحی کرده‌اید از یک یا چند فرم تشکیل می‌شد که هر یک به طور مستقل روی دسک‌تاپ به نمایش درآمده و مورد استفاده قرار می‌گرفتند و بسته شدن، انتقال، به حداکثر رسانی یا به حداقل‌رسانی یک فرم تأثیری روی سایر فرم‌های برنامه نمی‌گذاشت، به این گونه برنامه‌ها رابط گرافیکی تک سندی یا فرم‌های SDI [۲] گفته می‌شود. اما ممکن است برنامه‌هایی مانند ویژوال‌بیسیک، Word، Excel یا انواع مشابهی را دیده باشید که در آن‌ها یک پنجره به عنوان پنجره اصلی وجود دارد و سایر فرم‌ها و پنجره‌ها در داخل این پنجره باز می‌شوند و به پنجره اصلی وابستگی دارند که در صورت انتقال یا بسته شدن پنجره اصلی، تمام پنجره‌های موجود در آن منتقل یا بسته می‌شوند، اما بسته شدن یا انتقال پنجره‌هایی که به پنجره اصلی وابسته هستند تأثیری روی پنجره اصلی برنامه نمی‌گذارد به این گونه از پنجره‌ها، رابط گرافیکی چند سندی یا MDI گفته می‌شود و به پنجره اصلی پروژه فرم والد (Parent) یاMDI و به پنجره‌ها و فرم‌های وابسته، فرم فرزند (Child) گفته می‌شود (شکل ۱۰–۱۰).

شکل ۱۰ –۱۰

یک پروژه می‌تواند فقط یک فرم MDI داشته باشد، اما استفاده از چند فرم Child در یک پروژه بلامانع است. در ضمن استفاده از بعضی کنترل‌ها مانند دکمه فرمان، کادر متن، کادر تصویر و نظایر آن‌ها در روی فرم‌های MDI امکان‌پذیر نیست.

۱– رابط کاربر چند سندی (Multi Document Interface) MDI
۲– رابط کاربر یک سندی (Single Document Interface) SDI

مثال ۲: پروژه texteditor را طوری تغییر دهید که مطابق شکل ۱۱-۱۰ به‌صورت یک رابط گرافیکی MDI مورد استفاده قرار گیرد و تایپ متن در یک فرم SDI انجام شود. به این منظور عملیات زیر را به ترتیب انجام دهید:

۱ - برنامه ویژوال‌بیسیک را اجرا کنید و یک پروژه از نوع Standard EXE ایجاد کنید.

۲ - در پنجره پروژه کلیک راست کنید و از زیرمنوی Add گزینه MDI Form را انتخاب کنید (شکل ۱۲-۱۰) تا یک فرم از نوع MDI ایجاد شود. در این صورت کادر محاوره Add MDI Form (مطابق شکل ۱۳-۱۰) نمایش داده می‌شود. روی دکمه Open کلیک کنید تا یک فرم MDI به پروژه اضافه شود (شکل ۱۴-۱۰). برای ایجاد یک فرم از نوع MDI می‌توانید یکی از روش‌های زیر را نیز انتخاب کنید:

الف- از منوی Project در پنجره ویژوال‌بیسیک گزینه Add MDI Form را انتخاب کنید.

ب- روی علامت مثلث دکمه Add Form در نوار ابزار استاندارد کلیک کرده و گزینه MDI Form را انتخاب کنید.

شکل ۱۱ -۱۰

شکل ۱۲ -۱۰

شکل ۱۳ -۱۰

شکل ۱۴ -۱۰

۳ – در پنجره پروژه، آیکن Form1 را انتخاب کرده و روی آن کلیک راست کنید و از منویی که ظاهر می‌شود، گزینه Remove Form1 را برگزینید تا فرم از پروژه خارج شود.

بـرای خارج کردن یک فـرم از پروژه می‌توانید گزینه (نام فـرمRemove) را از منوی Project در پنجره ویژوال‌بیسیک انتخاب کنید.

نکته در صورت خارج کردن یک فرم از پروژه, فایل فرم از روی دیسـک حذف نمی‌شـود.

۴ – مشخصات فرم MDI را مطابق جدول ۳–۱۰ تنظیم کنید، سپس فرم و پروژه را با نام neweditor ذخیره کنید.

جدول ۳-۱۰ خصوصیات فرم

مقدار	خصوصیت
mdieditor	Name
Text Editor	Caption

۵ – در ایـن مرحلـه فـرم text editor.frm را که در مثال قبل ایجـاد کرده‌اید به پروژه جاری اضافه کنید. سـپس به پنجره طراحی این فـرم بروید و کادر محاوره Menu Editor را فعال کنید و روی دکمه Delete کلیک کنید. گزینه File از کادر لیسـت اسـامی عناوین منوها حذف می‌شـود. به همین روش تمام گزینه‌ها را حذف کرده و در پایان روی دکمه OK کلیک کنید.

۶ – در ایـن مرحله باید بین فرم‌هـای MDI و frmeditor ارتباط لازم را برقرار کنید تا فرم frmeditor به فرم MDI وابسـته شـود. به این منظور در پنجره پروژه روی آیکن فرم frmeditor کلیک کنید، سـپس در پنجره خصوصیـات, خصوصیت MDIChild را انتخاب کنید و مقدار آن را روی True تنظیم کنید.

۷ – همان‌طور که می‌بینید نوار منو از فرم Text Editor حذف شده است. نام فرم حاصل را روی frmtext تنظیم کنید سپس آن را با نام text ذخیره کنید.

۸ – اکنون به پنجره طراحی فرم MDI بروید و منویی را که از فرم قبلی حذف کرده‌اید مجدداً به همان شکل ایجاد کنید.

۹ – به پنجره ماژول فرم frmtext بروید و تمام رویدادهای مربوط به منوها را انتخاب کنید و به ماژول فرم MDI انتقال دهید (ابتدا عمل Cut و بعد در ماژول فرم MDI عمل Paste انجام دهید)، سپس رویداد Load فرم MDI و سایر رویدادها را به صورت زیر تنظیم کنید:

```
Private Sub MDIForm_Load()
    mdieditor.Width = 8000
    mdieditor.Height = 7000
    frmtext.Top = 500
    frmtext.Left = 300
    frmtext.Show
    frmtext.txttext.FontSize = 12
    frmtext.txttext.ForeColor = vbBlue
End Sub
Private Sub mnu10_Click()
    mnu10.Checked = True
    mnu12.Checked = False
    mnu14.Checked = False
    frmtext.txttext.FontSize = 10
End Sub
Private Sub mnu12_Click()
    mnu10.Checked = False
    mnu12.Checked = True
    mnu14.Checked = False
    frmtext.txttext.FontSize = 12
End Sub
Private Sub mnu14_Click()
    mnu10.Checked = False
    mnu12.Checked = False
    mnu14.Checked = True
    frmtext.txttext.FontSize = 14
End Sub
Private Sub mnublue_Click()
    mnublue.Enabled = False
    mnugreen.Enabled = True
```

```
        mnured.Enabled = True
        frmtext.txttext.ForeColor = vbBlue
End Sub
Private Sub mnugreen_Click()
        mnublue.Enabled = True
        mnugreen.Enabled = False
        mnured.Enabled = True
        frmtext.txttext.ForeColor = vbGreen
End Sub
Private Sub mnured_Click()
        mnublue.Enabled = True
        mnugreen.Enabled = True
        mnured.Enabled = False
        frmtext.txttext.ForeColor = vbRed
End Sub
Private Sub mnunew_Click()
       frmtext.txttext.Text = ""
End Sub
Private Sub mnuexit_Click()
   Unload Me
End Sub
```

۱۰ – به پنجره ویژگی‌های پروژه بروید و فرم mdieditor را به عنوان اولین فرم برای نمایش در ابتدای اجرای پروژه انتخاب کنید، سپس تغییرات را ذخیره کنید و پروژه را اجرا نمایید.

۱۱ – پنجره‌های پروژه مطابق شکل ۱۱–۱۰ نمایش داده می‌شوند، سعی کنید پنجره فرزند را از محدوده فرم MDI خارج کنید. همان‌طور که انتظار می‌رود فرم فرزند به فرم MDI وابسته شده است.

۱۲ – روی دکمه فرم فرزند کلیک کنید، مشاهده می‌کنید که فرم MDI بسته نمی‌شود. به اجرای پروژه خاتمه داده و مجدداً آن‌را اجرا کنید و این بارروی دکمه پنجره MDI کلیک کنید. خواهید دید که پنجره فرزند نیز با پنجره MDI بسته می‌شود.

۱۳ – اجرای پروژه را متوقف کنید و به پنجره ویژوال‌بیسیک بازگردید.

تمرین:

پروژه قبل را طوری تنظیم کنید که در هنگام اجرای برنامه، فقط فرم MDI به همراه منوی File مشاهده شود و فرم فرزند به همراه سایر گزینه‌ها زمانی فعال شود که گزینه New از منوی File انتخاب شده و فرم فرزند فاقد دکمه Close و منوی کنترل باشد.

۳-۱۰ کنترل‌های نوار پیمایش افقی و عمودی

تاکنون از نوارهای پیمایش در ویندوز و پنجره‌های برنامه استفاده کرده‌اید و می‌دانید از این اجزا زمانی استفاده می‌شود که محتویات مورد نیاز برای نمایش در پنجره از ابعاد پنجره بزرگ‌تر و بیشتر بوده و کاربر با به‌کارگیری نوار پیمایش، محتویات پنجره را مرور کند. به عنوان نمونه کاملاً مشخصی از نوار پیمایش, می‌توان به پنجره My Computer اشاره کرد که وقتی تعداد فایل‌ها و پوشه‌ها بیش از اندازه زیاد باشد؛ این نوارهای پیمایش هستند که به کاربر در مشاهده محتویات یک پوشه کمک می‌کنند یا نوارهای پیمایش در کادرهای متن و نظایر آن‌ها را می‌توان نام برد.

می‌دانید که این کنترل‌ها از دو دکمه مثلثی شکل و یک دکمه مستطیل شکل متحرک دربین‌دکمه‌های مثلثی تشکیل شده‌اند. کاربر می‌تواند با کلیک روی دکمه‌های مثلثی, کشیدن دکمه مستطیل شکل یا کلیک روی نقطه‌ای خالی از کنترل نوار پیمایش, حرکت کند.

در ویژوال‌بیسیک دو کنترل نوار پیمایش عمودی (VerticalScrollBar) و نوار پیمایش افقی (HorizontalScrollBar) برای استفاده در پروژه‌های برنامه‌نویسی وجود دارند. در این‌جا خصوصیت‌ها و رویدادهایی از این کنترل‌ها که مشترک هستند، بررسی می‌شود.

خصوصیات LargeChange و SmallChange

این دو خصوصیت مقدار تغییرات را هنگام استفاده از کنترل تعیین می‌کنند. خصوصیت SmallChange مقدار تغییرات را هنگامی که روی دکمه‌های مثلثی کنترل نوار پیمایش کلیک شود، تعیین می‌کند و خصوصیت LargeChange مقدار تغییرات را هنگامی که کاربر روی مکانی از نوار پیمایش بجز دکمه‌ها کلیک می‌کند، معین می‌نماید. این دو خصوصیت از نوع عددی هستند.

خصوصیات Max و Min

این دو خصوصیت مقدار حداکثر را که با رسیدن دکمه متحرک کنترل به انتهای نوار پیمایش و مقدار حداقل را که با رسیدن دکمه متحرک به ابتدای نوار پیمایش به‌دست می‌آید، تعیین می‌کنند. این دو خصوصیت نیز از نوع عددی صحیح هستند.

خصوصیت Value

این خصوصیت مقدار جاری را در نوار پیمایش معین می‌کند، این خصوصیت نیز ازنوع عددی است.

رویداد Change

این رویداد زمانی رخ می‌دهد که مقدار خصوصیت Value کنترل تغییر کند. این مسأله وقتی اتفاق می‌افتد که روی دکمه‌های مثلثی شکل یا در مکان خالی روی نوار پیمایش کلیک کنید.

رویداد Scroll

این رویداد زمانی اجرا می‌شود که کاربر دکمه متحرک مستطیل شکل را در نوار پیمایش به‌وسیله ماوس جابه‌جا کند.

مثال ۳: یک برنامه برای مشاهده تصاویر ایجاد کنید که مطابق شکل ۱۵–۱۰ توانایی نمایش تصویر دلخواه را داشته باشد و در صورتی‌که تصویر بزرگ‌تر از اندازه فرم باشد بتواند با استفاده از نوارهای پیمایش تمام تصویر را مرور کند.

۱ – یک پروژه از نوع Standard EXE ایجاد کنید که شامل یک فرم و کنترل تصویر (Image) با عرض و ارتفاع ۶۰۰۰ و ۵۰۰۰ با یک نوار منو مطابق شکل‌های ۱۵–۱۰ و ۱۶–۱۰ و جداول ۴–۱۰ و ۵–۱۰ باشد.

شکل ۱۵–۱۰

شکل ۱۶–۱۰

جدول ۵–۱۰ خصوصیات کنترل

کنترل / خصوصیت	TextBox
Name	Imgshow
Stretch	True

جدول ۴–۱۰ خصوصیات فرم

خصوصیت	مقدار
Name	frmeditor
Caption	Picture Viewer

۲ – در این مرحله رویدادهای مربوط به گزینه‌های منوها را به این صورت تنظیم کنید:

```
Private Sub mnuopen_Click()
        Dim txtpicture As String
   txtpicture = InputBox("Enter Path and File Name :", "Open")
   imgshow.Stretch = False
      imgshow.Picture = LoadPicture(txtpicture)
End Sub
Private Sub mnuzoomin_Click()
     imgshow.Stretch = True
   imgshow.Width = imgshow.width * 2
   imgshow.Height = imgshow.Height * 2
End Sub
Private Sub mnuzoomout_Click()
       imgshow.Stretch = True
   imgshow.Width = imgshow.width / 2
   imgshow.Height = imgshow.Height / 2
End Sub
Private Sub mnunormal_Click()
       imgshow.Stretch = False
End Sub

Private Sub mnuexit_Click()
       Unload Me
End Sub
```

همان‌طور که مشاهده می‌کنید در رویداد گزینه Open با استفاده از کادر ورود داده نام و مسیر فایل تصویر از کاربر دریافت می‌شود، سپس خصوصیت Stretch کنترل تصویر روی False تنظیم می‌شـــود تا تصویر با اندازه واقعی مشـــاهده شود و در ادامه تابع LoadPicture تصویر مورد نظر را در کنترل تصویر نمایش خواهد داد.

در رویـــداد گزینه‌هـــای Zoom In و Zoom out ابتدا خصوصیـــت Stretch کنترل تصویر روی True تنظیـــم می‌شـــود تا امکان تغییر اندازه تصویر با تغییـــر اندازه کنترل امکان‌پذیر باشد. سپس با ضرب یا تقسیم عرض کنترل تصویر بر عدد ۲، حالت بزرگ‌نمایی زیادتر یا کمتر به‌وجود می‌آید.

رویداد گزینه Normal نیز می‌تواند با False کردن خصوصیت Stretch کنترل تصویر آن را مجدداً به حالت عادی خود باز گرداند.

۳ – برای ایجاد نوار پیمایش عمودی در جعبه ابزار روی آیکن VScrollBar دابل کلیک کنید، سپس روی آیکن HScrollBar دابل کلیک کنید تا یک نوار پیمایش افقی نیز روی فرم ایجاد شود، سپس خصوصیات آن‌ها را مطابق جدول ۶-۱۰ تنظیم کنید.

جدول ۶-۱۰ خصوصیات کنترل‌ها

کنترل / خصوصیت	VScrollBar	HScrollBar
Name	vsbimage	hsbimage
LargeChange	۱۰۰	۱۰۰
SmallChange	۲۰	۲۰

۴–اکنون رویدادهای Change و Scroll دو کنترل نوار پیمایش را به صورت زیر تنظیم نمایید:

```
Private Sub vsbimage_Change()
    imgshow.Top = -vsbimage.Value
End Sub
Private Sub vsbimage_Scroll()
    imgshow.Top = -vsbimage.Value
End Sub
Private Sub hsbimage_Change()
     imgshow.Left = -hsbimage.Value
End Sub
Private Sub hsbimage_Scroll()
     imgshow.Left = -hsbimage.Value
End Sub
```

در این رویدادها مشاهده می‌کنید، برای آن‌که عمل پیمایش روی تصاویر امکان‌پذیر باشد، مقدار خصوصیت Value کنترل نوار پیمایش عمودی به خصوصیت Top کنترل تصویر نسبت داده شده است. این امر سبب می‌شود تا در صورت پیمایش به وسیله هر یک از روش‌های ممکن تصویر با توجه به مقدار خصوصیت‌های SmallChange و LargeChange به سمت بالا حرکت کند. استفاده از علامت منفی نیز به همین علت است در غیر این صورت با انجام عمل پیمایش، تصویر به پایین حرکت نخواهد کرد. به همین شکل کنترل نوار پیمایش افقی نیز با خصوصیت Left کنترل تصویر هماهنگ می‌شود تا با حرکت در نوار پیمایش افقی به سمت جلو، قسمت‌های جلوتر تصویر قابل مشاهده شود.

۵ – در این مرحله دستورات زیر را به رویداد Click گزینه‌های Open، Zoom In، Zoom out و Normal اضافه کنید. این دستورات سبب می‌شوند تا حداکثر میزان حرکت در نوارهای پیمایش (خصوصیت Max کنترل‌های نوار پیمایش) با توجه به ابعاد تصویر تنظیم شوند و با تغییر در میزان بزرگ‌نمایی یا باز شدن تصویر جدید، مقدار خصوصیت نوارهای پیمایش صفر شده تا تصویر از بالاترین بخش آن قابل رؤیت باشد.

```
hsbimage.Max = imgshow.Width

vsbimage.Max = imgshow.Height

hsbimage.Value =0

vsbimage.Value =0
```

۶ - پروژه و فرم را با نام Picture Viewer ذخیره کرده، سپس آن را اجرا کنید و روی گزینه Open در منوی File کلیک کرده، نام و مسیر یک فایل تصویر از نوع BMP یا JPG را در کادر ورود داده تایپ کنید سپس روی دکمه OK کلیک کنید.

۷ – گزینه‌های منوی View را آزمایش کنید و با استفاده از نوارهای پیمایش، تصویر را مرور کنید. سپس تصویر دیگری را باز کرده، مراحل ۶ و ۷ را مجدداً تکرار نمایید.

تمرین:

امکانات زیر را به برنامه Picture Viewer اضافه کنید.

الف– در صورتی که ابعاد تصویر کوچک‌تر از ابعاد فرم باشد، نوارهای پیمایش مخفی می‌شوند.

ب– گزینه‌هایی در منوی View اضافه کنید که به وسیله آن بتوان نوارهای پیمایش را مخفی و آشکارکرد.

مثال ۴: پروژه Picture Viewer را به گونه‌ای تنظیم کنید که بتوان با کلیک راست روی تصویر نیز به گزینه‌های منوی View دسترسی پیدا کرد (شکل ۱۷–۱۰). برای این کار عملیات بعد را به ترتیب انجام دهید:

شکل ۱۷–۱۰

۱–پروژه PictureViewer را باز کنید و به ماژول فرم و رویداد imgshow_MouseDown بروید.

۲ – سپس این رویداد را به صورت زیر تنظیم کنید:

```
Private Sub imgshow_MouseDown(Button As Integer , _
                              Shift As Integer, X As Single, Y As Single)
    If Button = vbRightButton Then PopupMenu mnuview
End Sub
```

مشاهده می‌کنید که با استفاده از دستور PopupMenu می‌توانید منوهای موضوعی‌خود را با استفاده از کلیک راست ایجاد کنید. در این‌جا با استفاده از رویداد MouseDown و یک دستور If، فشرده شدن دکمه راست ماوس بررسی می‌شود؛ سپس دستور PopupMenu گزینه‌های منوی View را در زمان کلیک راست روی کنترل تصویر در اختیار شما قرار می‌دهد.

۳ – تغییرات را ذخیره کرده، پروژه را اجرا کنید و یک تصویر دلخواه را نمایش دهید، سپس روی تصویر کلیک راست کنید.

مطابق شکل ۱۷–۱۰ منوی View روی فرم ظاهر می‌شود، اکنون با استفاده از این منو عملکرد برنامه را مجدداً بررسی کنید.

۴ – به اجرای پروژه خاتمه داده و به پنجره ویژوال‌بیسیک بازگردید.

۱۰-۴ کنترل کادر محاوره (CommonDialog)

این کنترل امکان ایجاد کادرهای محاوره برای بازکردن و ذخیره‌سازی فایل‌ها، انجام عملیات چاپ، انتخاب رنگ‌ها و فونت‌ها و نمایش راهنما را فراهم می‌آورد. آیکن این کنترل به‌طور پیش‌فرض در جعبه ابزار دیده نمی‌شود؛ به منظور افزودن این کنترل به جعبه ابزار به ترتیب زیر عمل کنید:

۱ – روی منوی Project در نوار منوی ویژوال‌بیسیک کلیک کنید.

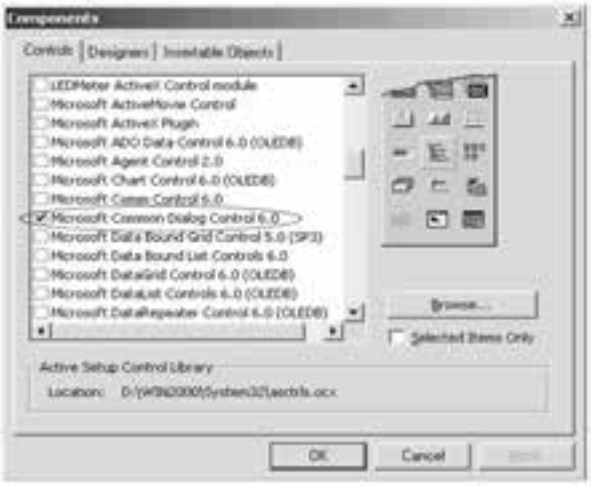
۲ – در منوی Project روی گزینه Components کلیک کنید.

۳ – پس از باز شدن پنجره Components روی زبانه Controls در این پنجره کلیک کنید.

۴ – گزینه Microsoft Common Dialog Control 6.0 را در لیستی که نمایش داده شده است انتخاب کنید (روی مربع ابتدای جمله کلیک کنید).

۵ – دکمه فرمان OK را بفشارید تا به محیط طراحی فرم بازگردید. آیکن کنترل مزبور در جعبه ابزار اضافه شده است (شکل‌های ۱۸-۱۰ و ۱۹-۱۰).

شکل ۱۸-۱۰ نحوه اضافه کردن کنترل CommonDialog به جعبه ابزار



شـــکل ۱۹–۱۰ پنجـــره Components برای اضافه کردن کنتـــرل CommonDialog به جعبه ابزار

پس از اضافه شـــدن آیکن کنترل می‌توانید این کنترل را مانند سایر کنترل‌ها به فرم‌های خود اضافـــه کنید. این کنترل هنگام اجرای برنامه در روی فرم مشـــاهده نمی‌شـــود و فقط زمانی کادرهـــای محاوره نمایش داده می‌شـــوند که متدهای مربوط به آن‌ها فراخوانی شـــوند. این کنترل به‌وســـیله متدهای زیر می‌تواند کادرهای محاوره مربوطه را نمایش دهد این متدها را در جدول ۷–۱۰ مشاهده می‌کنید.

جدول ۷–۱۰ انواع متدهای کنترل CommonDialog

نام کادرمحاوره	نام متد
کادرمحاوره باز کردن فایل‌ها	ShowOpen
کادرمحاوره ذخیره‌سازی فایل‌ها	ShowSave
کادرمحاوره رنگ‌ها	ShowColor
کادرمحاوره فونت	ShowFont
کادرمحاوره چاپگر	ShowPrinter
کادرمحاوره راهنما	ShowHelp

کنترل CommonDialog در این حالت می‌تواند خصوصیات زیر را در دسترس شما قرار دهد:

خصوصیت FileName: این خصوصیت نام و مســـیر فایلی را که توسط کاربر انتخاب شده است، نگهداری می‌کند. نوع این خصوصیت رشته‌ای است.

خصوصیت Filter: این خصوصیت می‌تواند نوع فایل‌هایی را که باید در کادر محاوره

نمایش داده شـــوند تعیین کند، مثلاً اگر بخواهید فایل‌هایی با پســـوند txt و doc را در کادرمحاوره ببینید، خصوصیت Filter را به‌صورت زیر تنظیم کنید.

```
dlg.Filter="Text Files(*.txt)|*.txt|Documents(*.doc)|*.doc"
```

هر بخش در خصوصیت Filter شـــامل دو قسمت می‌شـــود؛ قسمت اول یک توضیح اســـت که در لیســـت Files of types دیده می‌شود و قسمت دوم عبارتی است که می‌تواند نحوه نمایش فایل‌ها را تعیین کند.

خصوصیـــت :FilterIndex در صورتـــی که بیش از یک فایل را به‌وســـیله خصوصیت Filter برای نمایش در کادر محاوره تعیین کرده باشید، به‌طور پیش‌فرض اولین نوع فایل در خصوصیت Filter در کادرمحاوره در نظر گرفته می‌شـــود. به‌وسیله خصوصیت FilterIndex می‌توانیـــد گزینه پیش‌فرض را انتخاب کنید. خصوصیت FilterIndex می‌تواند مقادیر عددی صحیح از ۱ به بالا را کسب کند و بر اساس قرار گرفتن گزینه‌های مورد نظر در خصوصیت Filter محاســـبه شود. مقدار پیش‌فرض این خصوصیت اولین گزینه یا مقدار ۱ است. شکل کلی نحوه تنظیم این خصوصیت به این صورت است:

نام کنترل CommonDialog . FilterIndex = مقدارعددی

خصوصیت FileTitle: به‌وســـیله ایـــن خصوصیت می‌توان به نام فایل انتخاب شـــده دسترسی پیدا کرد. این خصوصیت فقط نام فایل را بدون مسیر آن نگهداری می‌کند.

خصوصیت InitDir: زمانی که کادر محاوره Open یا Save As نمایش داده می‌شود اسامی فایل‌ها و پوشـــه‌های مسیر جاری در دیسک نمایش داده می‌شود. در صورتی که بخواهید مســـیر ویژه‌ای را برای کادرمحاوره در نظر بگیرید، می‌توانید مســـیر مورد نظر را در این خصوصیت ذخیره کنید. شکل کلی نحوه استفاده از این خصوصیت به‌صورت زیر است:

نام کنترل CommonDialog .InitDir = مسیر

مسیر یک عبارت رشته‌ای است که موقعیت فایل مورد نظر را تعیین می‌کند.

نکته خصوصیات فوق را می‌توانید از طریق کادر محاوره Property Pages نیز تنظیم کنید. برای دسترسی به پنجره مزبور، در پنجره خصوصیات، کنترل CommonDialog را انتخاب کرده و روی خصوصیت Custom آن کلیک کـنید، در ادامه در روبـه‌روی این خصوصیت روی دکمه ... کلیک کنید, کادر محاوره Property Pages نمایش داده خواهد شـــد. در ادامه روی زبانه Open / Save As کلیک کنید، می‌توانید مقادیر مورد نظر را برای هر خصوصیت در این پنجره و در بخش مربوط به هر خصوصیت بنویسید.

شکل ۲۰-۱۰

مثال ۵: پروژه Picture Viewer را به‌گونه‌ای تنظیم کنید تا بتوان با استفاده از کادر محاوره باز کردن فایل‌ها، هر تصویر دلخواهی را انتخاب و مشاهده کرد. به این منظور عملیات زیر را به ترتیب انجام دهید:

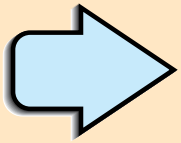
۱ – پروژه Picture Viewer را باز کنید.

۲ – یک کنترل کادرمحاوره روی فرم قرار دهید و خصوصیت نام آن‌را روی dlgopen تنظیم کنید.

۳ – به ماژول فرم و رویدادmnuopen Click_ بروید و رویدادهای زیر را تنظیم کنید:

```
Private Sub Form_Load()
        dlgopen.Filter = "BitMapFile (*.bmp)|*.bmp|AllFiles(*.*)|*.*"
        dlgopen.FilterIndex = 1
        dlgopen.InitDir = "c:\"
End Sub
Private Sub mnuopen_Click()
        dlgopen.DialogTitle = "Open"
        dlgopen.ShowOpen
        imgshow.Stretch = False
        imgshow.Picture = LoadPicture(.dlgopen.FileName)
End Sub
```

۴ – پس از انجام، تغییرات فوق را ذخیره نمایید برنامه را اجرا کرده و روی گزینه Open از منوی File کلیک کنید، کادرمحاوره Open باز می‌شود. در کادرمحاوره Open روی دکمه در لیست Files of type کلیک کنید تا لیست مربوطه باز شود. همان‌طور که ملاحظه می‌کنید توضیحات دو نوع فایل قابل مشاهده است؛ در صورتی که (*.bmp) BitMapFileانتخاب شود فقط فایل‌های با پسوند bmp و در صورتی که گزینه(*.*) AllFilesرا انتخاب کنید، تمام فایل‌ها را خواهید دید. یک فایل گرافیکی به‌دلخواه انتخاب کرده و آن‌را نمایش دهید.



Learn in English

Assigning Access Keys to Menu Item

You can improve keyboard access to menu commands by defining access keys and shortcut keys.

Access Keys

Access keys allow the user to open a menu by pressing the ALT key and typing a designated letter. Once a menu is open, the user can choose a control by pressing the letter (the access key) assigned to it. For example, ALT+E might open the Edit menu, and P might select the Paste menu item. An access-key assignment appears as an underlined letter in the menu control's caption.

To assign an access key to a menu control in the Menu Editor

1 – Select the menu item to which you want to assign an access key.

2 – In the **Caption** box, type an ampersand (&) immediately in front of the letter you want to be the access key.

واژه‌نامه

Access Key	کلید دسترسی
Assignment	مأموریت
Compiler	مترجم
Immediately	بی‌درنگ، بلافاصله
Improve	اصلاح کردن، بهبود دادن
Interpreter	مفسر
Object	شئ
Source	برنامه مبدأ
Structural Programming	برنامه‌نویسی ساخت‌یافته

خلاصه مطالب

- در ویژوال‌بیسیک از کادر محاوره Menu Editor برای ایجاد انواع منو استفاده می‌شود.
- از کنترل‌های نوار پیمایش افقی و عمودی برای مرور تصویر درکنترل‌های تصویر یا محتویات پنجره‌ها استفاده می‌شود.
- با استفاده از کنترل CommonDialog می‌توان انواع کادرهای محاوره باز و ذخیره کردن فایل، فونت، رنگ و غیره را ایجاد نمود.

آزمون نظری

۱ –کدام دستور برای ایجاد منوهای موضوعی به وسیله کلیک راست مناسب هستند؟

الف– MouseUp ب– PopupMenu ج– MouseMove د– Menu Editor

۲ – کـدام رویداد در کنترل‌های نوار پیمایش در زمان کلیک روی مکان خالی از نوار پیمایش اجرا می‌شود؟

الف– Change ب– Scroll ج– Click د– Move

۳ – در کنترل‌های نوار پیمایش کدام خصوصیت میزان تغییرات خصوصیت Value را در زمان کلیک روی دکمه‌های مثلثی کنترل تعیین می‌کند؟

الف– LargeChange ب– SmallChange ج– Scroll د– Change

۴ – کدام خصوصیت در کنترل کادرمحاوره می‌تواند نوع فایل‌هایی را که باید نمایش داده شوند تعیین کند؟

الف– File Name ب– Filter ج– Filter Index د– InitDir

۵ – برای نمایش یک منوی موضوعی از کدام دستور استفاده می‌شود؟

الف– Menu ب– MenuPopup

ج– Popup د–PopupMenu

۶ – کدام رویداد در کنترل‌های نوار پیمایش در هنگام درگ کردن دکمه مستطیل شکل اجرا می‌شود؟

الف– LargeChange ب– SmallChange

ج– Scroll د–MinScroll

۷ – کـدام رویـداد در کنترل‌های نوار پیمایش در زمان کلیـک روی دکمه‌های مثلثی شکل اجرا می‌شود؟

الف– Scroll ب– LargeChange ج– SmallChange د–Change

۸ – کدام متد نمی‌تواند یک کادرمحاوره را نمایش دهد؟

الف– ShowOpen ب– ShowFile ج– ShowSave د– ShowColor

9 - Which of the following characters is required to create an access key for Menu item?

a- * b- ? c- - d- &

۱۰– انواع منو در رابط‌های گرافیکی را همراه با کاربرد هر یک توضیح دهید.
۱۱– انواع کادرهای محاوره و کاربرد هر یک از آن‌ها را توضیح دهید.
۱۲– انواع کنترل‌های نوار پیمایش و نحوه استفاده از آن‌ها را توضیح دهید.
۱۳– انواع منو و کاربرد هر یک از آن‌ها را توضیح دهید.
۱۴– خصوصیت Filter در کادرمحاوره Open را توضیح دهید.

آزمون عملی

۱ – یک برنامه نقاشی مطابق این شکل‌ها به گونه‌ای طراحی کنید تا کاربر توانایی رسم خطوط و مستطیل‌های دلخواه خود را با رنگ‌های مورد نظر داشته باشد.

۲ – پروژه‌ای طراحی کنید که مطابق شکل‌های زیر توانایی محاسبه مساحت و محیط انواع اشـــکال هندسی را داشته باشد و به علاوه امکان تعیین واحد اندازه‌گیری نیز وجود داشته باشد.

واحد کاریازدهم

هدف جزیی

توانایی استفاده از انواع آرایه‌ها در ویژوال‌بیسیک

هدف‌های رفتاری

پس از مطالعه این واحد کار از فراگیر انتظار می‌رود که:

۱- توانایی ایجاد انواع آرایه‌های یک بعدی، با ابعاد ثابت و متغیر را داشته باشد.
۲- توانایی ذخیره‌سازی و بازیابی داده‌ها به‌وسیله آرایه‌های یک‌بعدی را داشته باشد.
۳- توانایی ارسال آرایه‌ها به رویه‌ها را داشته باشد.
۴- توانایی فراخوانی یک آرایه با تعداد آرگومان‌های نامعین را داشته باشد.
۵- توانایی ایجاد و استفاده از آرایه‌های دوبعدی را داشته باشد.
۶- نحوه استفاده از توابع Ubound و LBound را بداند.
۷- نحوه استفاده از توابع Split، Filter و Join را بداند.
۸- توانایی مرتب کردن اعضای یک آرایه با روش‌های زیر را داشته باشد.
الف ـ مرتب‌سازی با روش حبابی
ب ـ مرتب‌سازی با روش انتخابی
۹- توانایی جستجوی اطلاعات را در آرایه‌ها با روش خطی و دودویی داشته باشد.

کلیات

تاکنون با روش‌های مختلف ذخیره‌سازی اطلاعات در حافظه اصلی کامپیوتر آشنا شده‌اید که استفاده از انواع متغیرها و خصوصیات کنترل‌ها از نمونه‌های کاملاً مشخص آن است.

گاهی در برنامه‌نویسی‌های واقعی لازم است تعداد زیادی داده را مورد پردازش قرار دهیم که به ناچار باید به تعداد مورد نظر متغیر، تعریف کرد؛ اما این روش، همواره قابل اجرا نیست، مثلاً فرض کنید می‌خواهید برنامه‌ای را طراحی کنید که باید اسامی هزار نفر از دانشجویان یک دانشگاه را دریافت کند در چنین حالتی با توجه به دانسته‌های قبلی باید هزار متغیر با اسامی مختلف تعریف کنید. آیا این روش منطقی است؟ اگر تعداد داده‌ها باز افزایش پیدا کند، چطور؟ اگر بخواهید یک اسم را در میان مجموعه اسامی پیدا کنید، چه اتفاقی می‌افتد؟ این‌گونه عملیات با روش‌های معمول یا امکان‌پذیر نیست یا از نظر تکنیکی, منطقی نخواهد بود.

برای حل این مشکل و طراحی چنین برنامه‌هایی در تمام زبان‌های برنامه‌نویسی از مفهومی به نام آرایه (Array) استفاده می‌شود. یک آرایه در واقع یک سری از چندین متغیر با یک نام مشابه است که به‌وسیله یک اندیس (یک عدد صحیح مثبت) از یکدیگر متمایز می‌شوند.

استفاده از آرایه‌ها باعث می‌شود تا کدهای برنامه ساده‌تر و کوتاه‌تر شود زیرا شما می‌توانید با استفاده از انواع حلقه‌ها و شماره اندیس‌ها، به هر یک از اعضای آرایه دسترسی پیدا کنید. آرایه‌ها نیز مانند متغیرهای‌معمولی دارای نوع داده هستند و تمام اعضای یک‌آرایه از یک نوع داده هستند البته می‌توانید به‌وسیله استفاده از نوع داده Variant انواع مختلفی از داده‌ها را در اعضای یک آرایه ذخیره کنید.

استفاده از آرایه‌ها در برنامه‌های بزرگ اجتناب‌ناپذیر است و بدون استفاده از آن‌ها انجام عملیات مرتب‌سازی و جستجوی داده کار بسیار مشکلی خواهد بود.

ابعاد یک آرایه به‌وسیله دامنه پایینی و بالایی آن معین می‌شود و اعضای آرایه به‌طور پیوسته و پشت سر هم داخل این محدوده قرار می‌گیرند. ویژوال‌بیسیک برای هر یک از اعضای یک آرایه فضای جداگانه‌ای را در حافظه اختصاص می‌دهد بنابراین استفاده از آرایه‌هایی بزرگ‌تر از اندازه مورد نیاز، باعث اشغال حافظه بدون استفاده خواهد شد.

۱-۱۱ تعریف انواع آرایه در ویژوال‌بیسیک

در ویژوال‌بیسیک دو نوع آرایه وجود دارد: آرایه با ابعاد ثابت آرایه ایستا یا(Static Array)و آرایه با ابعاد متغیر (آرایه پویا یا Dynamic Array). ابتدا به نحوه تعریف آرایه‌ها با ابعاد ثابت می‌پردازیم:

۱-۱-۱۱ آرایه ایستا (Static Array)

برای تعریف آرایه با ابعاد ثابت می‌توانید از تمام روش‌هایی که تاکنون برای تعریف متغیرها به کار گرفته‌اید، استفاده کنید تنها تفاوتی که بین تعریف متغیر و آرایه وجود دارد تعیین ابعاد یک آرایه است. آرایه‌ها را می‌توانید به‌وسیله کلمات کلیدی Public، Private، Static و Dim در یک رویه یا بخش تعاریف ماژول فرم یا ماژول کد تعریف کنید.

برای تعریف یک آرایه با ابعاد ثابت می‌توانید یکی از روش‌های زیر را استفاده کنید:

[Dim | Public | Private | Static] نام آرایه (دامنه بالایی) As نوع داده

[Dim | Public | Private | Static] نام آرایه (دامنه پایینی To دامنه بالایی) As نوع داده

نکته با توجه به مکان تعریف آرایه و کاربرد آن می‌توانید یکی از کلمات کلیدی موجود در [ ] را انتخاب کنید.

مثلاً برای تعریف یک آرایه از نوع Integer و با تعداد ۱۵ عضو از فرمان زیر استفاده می‌شود:

```
Dim no (14) As Integer
```

در ویژوال‌بیسیک به‌طور پیش‌فرض اولین اندیس آرایه‌ها از شماره صفر آغاز می‌شود بنابراین در مثال قبل با توجه به مقدار دامنه بالایی، اندیس‌های آرایه از صفر تا ۱۴ در نظر گرفته می‌شوند و در نتیجه تعداد اعضا ۱۵ خواهد بود.

و در تعریف یک آرایه به‌صورت زیر:

```
Public counters(20) As Double
```

آرایه‌ای با تعداد ۲۱ عضو و از نوع Double در حافظه آدرس‌دهی خواهد شد. روش دوم در تعریف یک آرایه با ابعاد ثابت استفاده از دامنه بالایی و پایینی است مثلاً برای تعریف آرایه‌ای (با ۱۵ عضو) که اندیس اول آن از یک شروع شود و اندیس آخرین عضو در آن ۱۵ باشد از فرمان زیر استفاده می‌شود:

```
Dim counters (1 To 15) As Double
```

و در تعریف آرایه sums که به این صورت انجام شده است:

```
Private sums (100 To 120) As Variant
```

آرایه sums دارای اولین عضو با شماره اندیس ۱۰۰و آخرین عضو با شماره اندیس ۱۲۰خواهد بود به عبارت دیگر ۲۱ عضو خواهد داشت. برای آن‌که با نحوه کار آرایه‌ها بهتر آشنا شوید به ذکر مثالی در این رابطه می‌پردازیم:

مثال ۱: رویه‌ای بنویسید که ده عدد را به‌صورت تصادفی ایجاد کرده و در آرایه‌ای ذخیره کند، سپس آرایه را نمایش دهد.

به این منظور یک رویه با نام myrandom به صورت زیر بنویسید:

```
Sub myrandom( )
  Dim i As Integer, sngno(9) As Single
  Randomize
  For i = 0 To 9
     sngno(i) = Rnd
  Next i
  For i = 0 To 9
     Print , "number("; i + 1; ")="; sngno(i)
  Next i
End Sub
```

در رویه myrandom ابتدا آرایه‌ای با دامنه بالایی ۹ (۱۰عضو) تعریف شده است سپس به‌وسیله یک حلقه For که مقدار شمارنده آن (i) از صفر شروع می‌شود اولین عضو آرایه یعنی sngno(0) به‌وسیله تابع Rnd مقداردهی می‌شود برای نمایش اعدادی که در آرایه sngno ذخیره شده‌اند نیز از یک حلقه استفاده شده است. البته شما می‌توانید به جای For از حلقه‌های دیگر نیز استفاده کنید اما این کار با استفاده از حلقه For آسان‌تر خواهد بود. در رویه myrandom تابعی به نام Rnd وجود دارد، به وسیله این تابع می‌توان اعداد تصادفی بین صفر و یک را ایجاد کرد. این تابع یک عدد از نوعSingle را برمی‌گرداند و یک آرگومان اختیاری دارد که می‌تواند یک عدد از نوع Single یا یک عبارت عددی باشد. توجه داشته باشید که هربار تابع Rnd فراخوانی می‌شود یک عدد تصادفی تولید خواهد شد البته اگر برنامه مجدداً اجرا و تابع فراخوانی شود همان اعداد

به صورت تکراری به دست می‌آیند؛ برای جلوگیری از چنین حالتی و این‌که همواره اعداد تولید شده یکسانی به دست نیایند، می‌توانید قبل از تابع Rnd از تابع Randomize استفاده کنید. تابع Randomize دارای یک آرگومان اختیاری نیز بوده که می‌تواند یک عدد یا عبارت عددی باشد.

تمرین:
رویه مثال ۱ را به‌گونه‌ای تغییر دهید تا بزرگ‌ترین عضو آرایه را به‌دست آورده و همراه اعضای آرایه نمایش دهد. سپس در یک پروژه آن‌را فراخوانی کنید و نتیجه را مشاهده نمایید.

مثال ۲: رویه‌ای بنویسید که دو آرایه ۵ عضوی یک بعدی از اعداد تصادفی را ایجاد کرده و حاصل‌ضرب آن‌ها را محاسبه نماید و در آرایه دیگری ذخیره کند.

```
Sub mymatrix( )
  Dim i As Integer, no1(4) As Single
  Dim no2(4) As Single, no3(4) As Single
  Randomize
  For i = 0 To 4
      no1(i) = Rnd
      no2(i) = Rnd
  Next i
  For i = 0 To 4
     no3(i) = no1(i) * no2(i)
   Next i
End Sub
```

در این رویه از سه آرایه ۵ عضوی استفاده شده است و به‌وسیله اولین حلقه For اعضای دو آرایه no۱ و no۲ مقداردهی شده‌اند سپس با استفاده از حلقه For دوم حاصل‌ضرب اعضای متناظر دو آرایه در عضو متناظر آرایه حاصل‌ضرب یعنی no۳ قرار می‌گیرد.

علاوه بر آرایه‌های عددی می‌توانید داده‌های رشته‌ای را نیز به‌صورت آرایه ذخیره کنید. در واقع فرق زیادی بین آرایه‌های رشته‌ای و عددی وجود ندارد فقط در مورد آرایه‌های رشته‌ای طول هر عضو می‌تواند ثابت یا متغیر باشد.

تمرین:

پروژه‌ای طراحی کنید که با استفاده از یک آرایه، هر عددی از مبنای ۱۰ را به مبنای ۲ تبدیل کند.

مثال ۳: پروژه‌ای طراحی کنید که اسامی ۱۰نفر را دریافت کرده و در یک آرایه رشته‌ای ذخیره کند به علاوه بتوان اسامی ذخیره شده را روی یک فرم جداگانه مشاهده نمود.

به این منظور عملیات زیر را به ترتیب انجام دهید:

۱ – برنامه ویژوال‌بیسیک را اجرا کرده و یک پروژه از نوع Standard EXE ایجاد کنید سپس یک فرم مطابق شکل ۱–۱۱ و جدول ۱–۱۱ ایجاد کنید و کنترل‌های آن‌را مطابق جدول ۲–۱۱ روی فرم قرار دهید. در این فرم از یک کنترل کادر متن برای دریافت داده‌ها و از دکمه Add برای ذخیره‌سازی اسامی در آرایه استفاده می‌شود به علاوه با دکمه Show می‌توان اسامی ذخیره شده در آرایه را در یک فرم دیگر مشاهده کرد.

شکل ۱–۱۱

جدول ۱–۱۱ خصوصیات فرم

خصوصیت	مقدار
Name	frmarry
Caption	Array

جدول ۲–۱۱ خصوصیات کنترل‌ها

کنترل / خصوصیت	Command Button	Command Button	Command Button	Text Box	Label
Name	cmdadd	cmdshow	cmdexit	txtname	lblen
Caption	Add&	Show&	Exit&	————	Enter Name:

۲ – یک ماژول کد به پروژه اضافه کنید و در بخش تعاریف آن آرایه‌ای را با نام strname با ۱۰عضو که هر عضو آن نیز توانایی دریافت ۲۰کاراکتر را داشته باشد به صورت زیر تعریف کنید:

```
Public strname(9) As string* 2 0
```

رویداد Click کنترل دکمه Add را به این صورت تنظیم کنید:

```
Private Sub cmdadd_Click( )
    Static i As Integer
    If i < 10 Then
        strname(i) = txtname.Text
        i = i + 1
    Else
        MsgBox «Array is full»
    End If
End Sub
```

در این رویداد از یک متغیر i به‌عنوان شمارنده اندیس آرایه به‌صورت Static استفاده شده است تا به‌وسیله مقدار این متغیر بتوان در هر مرحله یک عضو در آرایه را با نامی‌که کاربر در کنترل کادر متن می‌نویسد پر کرد. اگر این متغیر به‌صورت محلی تعریف شود اسامی، همواره در اولین عضو آرایه ذخیره می‌شوند و در نتیجه، نامی که قبلاً در عضو اول آرایه ذخیره شده است از بین می‌رود. در ادامه اجرای رویداد، یک فرمان If مقدار i را کنترل می‌کند تا مقدار i از دامنه بالایی آرایه (یعنی ۹) تجاوز نکند در صورتی که مقدار i کنترل نشود در زمان رسیدن به مقدار ۱۰, چون بالاترین مقدار اندیس آرایه ۹ است پیام خطای Subscript out of range نمایش داده می‌شود. بنابراین اگر (i<10) باشد آنگاه محتویات خصوصیت Text کنترل کادرمتن در یکی از اعضای آرایه ذخیره می‌شود و سپس مقدار i یک واحد افزایش می‌یابد تا مرحله بعد, اندیس عضو بعدی آرایه آماده باشد، اما اگر کاربر ۱۰ نام را وارد کند در هنگام ورود نام یازدهم با پیامی‌که به‌وسیله یک تابع MsgBox نمایش داده می‌شود از کامل شدن روند عملیات ورود داده مطلع می‌شود زیرا در رویداد Click دکمه فرمان Add نتیجه بررسی شرط (i < 10) نادرست بوده و در نتیجه تابع MsgBox فراخوانی می‌شود.

۴ – یک فرم جدید با نام frmdisplay و عنوان Display مطابق شکل ۲–۱۱ به پروژه اضافه کرده سپس یک دکمه با نام cmdback و عنوان &Back روی آن قرار دهید. از این فرم برای نمایش اسامی ذخیره شده در آرایه استفاده می‌شود.

۵ – رویداد Click دکمه Show را در فرم frmarray به صورت زیر تنظیم کنید:

```
Private Sub cmdshow_Click( )
    frmarray.Hide
    frmdisplay.Show
End Sub
```

شکل ۲-۱۱

در این رویداد با استفاده از متد Hide فرم frmarray مخفی شده و با استفاده از متد Show فرم frmdisplay نمایش داده می‌شود.

۶ – رویداد دکمه cmdbackرا در فرم frmdisplay به‌صورت زیر تنظیم کنید:

```
Private Sub cmdback_Click( )
        unload frmdisplay
        frmarray.Show
End Sub
```

در این رویداد با استفاده از دستور unload فرم frmdisplay بسته شده و با متد Show امکان بازگشت به فرم اول فراهم می‌شود.

۷ – پس از تنظیم رویداد دکمه فرمان cmdback در فرم frmdisplay دستورات زیر را در رویداد Activate همین فرم به‌صورت زیر تنظیم کنید:

```
Private Sub Form_Activate( )
    Dim j
    For j = 0 To 9
        Print "NAME ( "; j ; " )= "; strname( j )
     Next j
End Sub
```

دستورات در این رویداد باعث خواهند شد تا پس از فعال شدن فرم محتویات آرایه به وسیله یک حلقه نمایش داده شود.

۸ – پروژه و فرم frmarray را با نام Array و فرم frmdisplay را با نام Display ذخیره کنید سپس برنامه را اجرا نمایید و تعداد ۱۰ نام را وارد کرده و بعد از ورود هر نام، دکمه Add را کلیک کنید و بعد از پایان ورود داده‌ها به‌وسیله دکمه Show محتویات آرایه را مشاهده نمایید در پایان با کلیک روی دکمه Back به فرم اول بازگردید.

۹ – روی دکمه Exit کلیک کنید و به پنجره ویژوال‌بیسیک بازگردید.

تمرین:

برنامه قبل را به‌گونه‌ای تغییر دهید که در صورت عدم استفاده از تمام اعضای آرایه (خالی ماندن بعضی از اعضا) در هنگام نمایش اطلاعات در فرم cmddisplay, فقط تا آخرین اندیسی از آرایه که حاوی نام است, نمایش داده شود مثلاً اگر کاربر ۴ نام را وارد کرده است فقط همان ۴ نام نمایش داده شوند و از نمایش عناصر بعدی آرایه یعنی اندیس‌های بزرگ‌تر از ۳ خودداری شود.

در این‌جا لازم است که به ذکر نکته مهمی در رابطه با شماره اندیس اولین عضو در آرایه بپردازیم. تاکنون وقتی یک آرایه را با ذکر مقدار اندیس بالایی آن تعریف می‌کنید شماره اندیس اولین عضو در آرایه از صفر شروع می‌شود، اما گاهی لازم است که این مقدار را با توجه به نیاز تغییر دهید، با استفاده از فرمان Option Base در بخش تعاریف می‌توانید مقدار اندیس اولین عضو را در آرایه‌ها تعیین کنید. شکل کلی فرمان به این صورت است:

Option Base n

که n می‌تواند صفر یا یک باشد در صورت استفاده از مقدار صفر یا عدم استفاده از فرمان فوق اندیس آغازین در آرایه‌های برنامه صفر خواهد بود و اگر بخواهید اندیس آغازین در آرایه‌های برنامه از یک شروع شود مقدار n را در فرمان مزبور ۱ انتخاب کنید.

نکته
- از این فرمان فقط یک بار و در بخش تعاریف یکی از ماژول‌ها استفاده کنید.
- برای تعیین دامنه پایینی آرایه‌ها به جای استفاده از فرمان Option Base بهتر است در تعریف آرایه‌ها دامنه پایینی و بالایی آرایه را تعیین کنید.

به‌عنوان مثال فرض کنید می‌خواهیم یک آرایه عددی با ۱۰عضو را به‌وسیله اعداد تصادفی مقداردهی کنیم (این مسأله قبلاً حل شده است اما از آن، برای بررسی عملکرد دستورOption Base استفاده می‌شود).

```
Sub myrandom( )
  Dim i As Integer, sngno(10) As Single
  Randomize
  For i = 1 To 10
    sngno( i ) = Rnd
  Next i
End Sub
```

اگر دستور Option Base 1 استفاده شود در رویه myrandom جدید و در تعریف آرایه sngnoبه جای عدد ۹ عدد ۱۰ استفاده می‌شود و مقدار پایانی در حلقه For نیز از ۹ به ۱۰ تغییر می‌یابد و مقدار شروع شمارنده نیز از صفر به ۱ تبدیل می‌شود.

توجه داشته باشید که در ذخیره‌سازی مقادیر هر دو حالت، مشابه هم عمل می‌شود؛ فقط در نحوه تعریف آرایه‌ها و استفاده از حلقه باید دقت کافی داشته باشید.

تمرین:

پروژه‌ای طراحی کنید که ده نمره درس‌های یک دانش‌آموز را در یک آرایه ذخیره کند سپس مجموع و معدل نمرات وی را محاسبه کرده و نمایش دهد (اندیس آرایه از عدد ۱ شروع شود).

۲–۱–۱۱ آرایه پویا (Dynamic Array)

گاهی اوقات ممکن است تعداد داده‌ها نامشخص باشد در نتیجه نمی‌توان تعداد اعضا آرایه را در زمان طراحی برنامه تعیین کرد بنابراین لازم است ابعاد آرایه در زمان اجرا تنظیم شود.

به این منظور ویژوال‌بیسیک نوع دیگری از آرایه‌ها را با نام آرایه پویا در اختیار شما قرار می‌دهد. ابعاد آرایه‌ای از این نوع را می‌توانید در زمان اجرای برنامه با توجه به نیازتان مکرراً تغییر دهید.

برای تعریف یک آرایه دینامیک، بهتر اســت مانند آرایه‌های ثابت عمل‌کرده اما از ذکر ابعاد آرایه خودداری کنید.

به‌عنوان مثال به دستور زیر توجه کنید:

```
Dim  dynnumber ( )  AS  Integer
```

اما در صورت اســتفاده از آرایه در این مرحله پیام خطا نمایش داده خواهد شد. برای قابل اســتفاده شــدن آرایه‌های پویا، پس از تعریف آن باید ابعاد آن را با استفاده از دستور ReDim تعیین کنید. به‌عنوان مثال پس از تعریف آرایه dynnumber از نوع Integer ابعاد آن به صورت زیر تعیین می‌شود:

```
ReDim dynumber  (10)
```

اگر ۰ Option Base باشد آرایه دارای یازده عضو و اگر ۱ Option Base باشد آرایه دارای ۱۰ عضو (از ۱ تا ۱۰) خواهد بود.

نکته

- در صورت استفاده از دستور ReDim مقادیر موجود در تمام اعضای آرایه از بین خواهد رفت بنابراین در استفاده مجدد از دستور ReDim با دقت کافی اقدام کنید.
- در صورتــی کــه بخواهید مقادیر موجود در آرایه در زمان تغییر ابعاد آن حفظ شــوند از کلمه کلیدی Preserve همراه با دستور ReDim استفاده کنید.
- آرایه‌های پویا را نیز می‌توانید به‌صورت Public، Private یا Static تعریف کنید.

به‌عنوان مثال به دستورات زیر توجه کنید: (با فرض این که Option Base 1 است).

```
Dim myarray( ) As Integer, i As Integer
ReDim myarray(5)
Next i
    ReDim myarray(10)
For i = 1 To 10
    Print myarray(i)
Next i
```

با اســتفاده از تعریف آرایه پویا آرایه myarray با ۵ عضو تعریف شــده اســت ســپس به‌وسیله یک حلقه For مقادیر ۱ تا ۵ در آرایه قرار گرفته‌اند. پس از حلقه For اول مجدداً دســتور ReDim به‌کار گرفته‌شــده اســت تا تعداد اعضای آرایه دو برابر شود پس از تغییر

ابعاد آرایه، حلقه For دوم مقادیر موجود در آرایه را نمایش می‌دهد اما استفاده دوباره از دستور ReDim، تمام مقادیر قبلی در آرایه را از بین می‌برد، در نتیجه فقط مقادیر صفر توسط حلقه نمایش داده می‌شود.

حال اگر به جای دستور ReDim myarray (10) ازدستور ReDim Preserve myarray (10) استفاده کنید، حلقه For دوم پس از نمایش مقادیر ۱ تا ۵ برای اعضای قبلی, مقدار صفر را هم برای ۵ عضو جدید که اضافه شده‌اند، نمایش می‌دهد.

تمرین:

یک پروژه جدید طراحی کنید و دستورات فوق را در رویداد Click یک دکمه فرمان قرار داده و نتیجه را در دو حالت بحث شده بررسی کنید.

نکته

• در صورت کاهش ابعاد یک آرایه به‌وسیله دستور ReDim مقادیر مربوط به اعضای حذف شده از بین می‌روند.

• به‌وسیله دستور ReDim نیز می‌توانید یک آرایه پویا Dynamic تعریف کنید شکل کلی این دستور برای تعریف یک آرایه پویا به‌صورت زیر است:

ReDim نام آرایه (دامنه بالایی) As نوع داده

مثلاً دستور زیر یک آرایه با ۲۰ عضو و از نوع رشته‌ای تعریف می‌کند.

```
ReDim fam (19) As String
```

• در صورت استفاده از دستور ReDim برای تغییر ابعاد یک آرایه ثابت در هنگام اجرای برنامه، پیام خطای Array already dimensioned نمایش داده می‌شود.

تمرین:

می‌خواهیم پروژه‌ای که در مثال ۳ طراحی شده است به‌گونه‌ای تغییر کند تا تعداد اسامی با توجه به نیاز و درخواست کاربر دریافت و در آرایه ذخیره شود. به این منظور پروژه Array را باز کنید و عملیات زیر را به ترتیب انجام دهید:

۳-۱-۱۱ دستور Erase

از این دستور برای حذف اعضا در آرایه‌های پویا و از بین بردن محتویات اعضا در آرایه‌های ثابت استفاده می‌شود. شکل کلی دستور Erase به‌صورت زیر است:

Erase نام آرایه

اگر این دستور روی یک آرایه پویا اجرا شود تمام مقادیر آن از بین خواهد رفت و شما می‌توانید مجدداً با استفاده از دستور ReDim از آرایه استفاده کنید اگر پس از حذف اعضای یک آرایه پویا، بدون استفاده از دستور ReDim سعی در دستیابی به اعضای آن داشته باشید پیام خطای Subscript out of range نمایش داده می‌شود.

اگر این دستور روی یک آرایه ثابت اجرا شود مقادیر موجود در آرایه حذف خواهند شد، اما فضاهای مربوط به اعضای آرایه در حافظه از بین نخواهند رفت.

به عنوان مثال به دستورات زیر توجه کنید (با فرض این که Option Base ۱ است):

```
Dim myarray1( ) As Integer, myarray2(5) As Integer
Dim i As Integer
ReDim myarray1(5)
For i = 1 To 5
   myarray1 (i) = i
   myarray2 (i) = i
Next i
Erase myarray1, myarray2
ReDim myarray1(5)
For i = 1 To 10
   Print myarray1(i)
   Print myarray2(i)
Next i
```

در این مجموعه دستورات از یک آرایه پویا (myarray1) و یک آرایه‌ای با ابعاد ثابت (myarray2) استفاده شده است. در اولین حلقه For، آرایه‌های myarray1 و myarray2 به ترتیب مقداردهی شده‌اند سپس به‌وسیله دستور Erase، آرایه myarray1 از حافظه کاملاً پاک شده اما آرایه myarray2 فقط مقادیر خود را از دست می‌دهد و در نتیجه حلقه For دوم فقط مقادیر صفر را برای هر دو آرایه نمایش می‌دهد. البته اگر دستور ReDim mayrray1(5) پس از حذف آرایه مزبور استفاده نشود آرایه در حلقه For قابل شناسایی نبوده و پیام خطای Subscript out of range نمایش داده می‌شود.

تمرین:

پروژه Array را به‌گونه‌ای تنظیم کنید که شرایط زیر را داشته باشد:

الف– امکان افزایش اسامی پس از پرشدن آرایه وجود داشته باشد.

ب– در صورت لزوم بتوان اعضای آرایه را حذف کرده و اسامی جدید در آن ذخیره نمود.

۲–۱۱ آرایه‌های چند بعدی

تاکنون با نحوه تعریف و چگونگی استفاده از آرایه‌های یک بعدی آشنا شدید اما در برنامه‌نویسی پروژه‌های واقعی، گاهی اوقات آرایه‌های یک بعدی نیز گره‌گشا نیستند و برنامه‌نویسی را با مشکلات متعددی روبه‌رو می‌کنند.

آرایه‌های دو بعدی از دو اندیس برای شناسایی اعضای خود استفاده می‌کنند. اندیس اول شماره سطر و اندیس دوم شماره ستون متناظر با آرایه را مشخص می‌کنند. برای روشن شدن بهتر موضوع به آرایه زیر توجه کنید, این آرایه دارای ۴ سطر و ۳ ستون است

جدول ۳-۱۱

شماره ستون / شماره سطر	0	1	2
0	$a_{0,0}$	$a_{0,1}$	$a_{0,2}$
1	$a_{1,0}$	$a_{1,1}$	$a_{1,2}$
2	$a_{2,0}$	$a_{2,1}$	$a_{2,2}$
3	$a_{3,0}$	$a_{3,1}$	$a_{3,2}$

در نوع از آرایه‌ها برای دسترسی به هر یک از عناصر آرایه از دو اندیس استفاده می‌شود مثلاً عنصر a۲,۱ , عضوی است که در سطر شماره ۲ و ستون شماره ۱ قرار دارد.

برای تعریف آرایه دو بعدی می‌توانید یکی از این روش‌ها را به کار ببرید:

[Dim| Public| Private| Static] (دامنه بالایی ستون‌ها، دامنه بالایی سطرها) نام آرایه

[Dim| Public| Private| Static] نام آرایه (دامنه بالایی ستون‌ها، دامنه بالایی سطرها) As نوع داده

[Dim|Public|Private|Static] نام آرایه (دامنه بالایی ستون‌ها To دامنه پایینی ستون‌ها، دامنه بالایی سطرها To دامنه پایینی سطرها) As نوع داده

نکته با توجه به مکان تعریف و کاربرد آرایه می‌توانید یکی از کلمات کلیدی موجود در [ ] را انتخاب کنید.

به‌عنوان مثال فرض کنید می‌خواهید نمرات سه درس دانش‌آموزان یک کلاس ۵ نفره را به‌صورت یک آرایه دو بعدی بنویسید.

	شیمی	فیزیک	ریاضی
	۲	**۱**	۰
دانش‌آموز اول	۱۶	۱۴	۱۷
دانش‌آموز دوم	۸	۱۰	۱۲
دانش‌آموز سوم	۱۵	۱۸	۲۰
دانش‌آموز چهارم	۱۹	۱۳	۱۴
دانش‌آموز پنجم	۱۱	۲۰	۱۷

شماره ستون / **شماره سطر**

اکنون می‌خواهیم این آرایه دو بعدی را تعریف کنیم بنابراین می‌توان یکی از موارد زیر را برای انجام این‌کار استفاده کرد:

```
Dim  sng student (4,2)  As  Single
Dim  sng student (0  To  4, 0 To 2) As  Single
```

نکته توجه داشته باشید که قوانین دستور Option Base برای آرایه‌های دو بعدی نیز صادق است.

مثال ۴: رویه‌ای بنویسید که آرایه دو بعدی ۴×۴ را با اعداد تصادفی مقداردهی کرده سپس اعضای آن را نمایش دهد (با فرض Option Base 0).

```
Sub mymatrix( )
  Dim A(4, 4) As Integer, i As Integer, j As Integer
  Randomize
  For i = 0 To 3
      For j = 0 To 3
         A(i, j) = Int(100 * Rnd + 1)
    Next j
  Next i
  For i = 0 To 3
      For j = 0 To 3
          Print "   "; "A("; i; ","; j; ")="; A(i, j)
      Next j
      Print
   Next i
  End Sub
```

رویه فرعی فوق در صورت اجرا، خروجی مشابه شکل زیر را خواهد داشت:

A(0,0)=0 A(0,1)=20 A(0,2)=17 A(0,3)=9

A(1,0)=2 A(1,1)= 0 A(1,2)=52 A(1,3)=53

A(2,0)=69 A(2,1)=44 A(2,2)=0 A(2,3)=31

A(3,0)=28 A(3,1)=87 A(3,2)=3 A(3,3)=0

در این رویه ابتدا آرایه دو بعدی A را تعریف کرده‌ایم سپس با استفاده از دو حلقه تو در تو و تابع Rnd عناصر آرایه مقداردهی شده‌اند. حلقه For اول به‌وسیله شمارنده i شماره سطر و حلقه دوم به وسیله شمارنده j شماره ستون را در آرایه کنترل می‌کنند بنابراین با هر بار اجرای حلقه اول، حلقه دوم سه بار تکرار می‌شود تا اعضای هر سطر را جهت مقداردهی به ترتیب آماده کند. به عنوان مثال زمانی که مقدار i برابر با صفر است حلقه دوم مقادیر j را به ترتیب روی مقادیر ۰، ۱، ۲ و ۳ تنظیم می‌کند. با خاتمه اجرای حلقه دوم، حلقه اول مقدار i را یک واحد افزایش می‌دهد و حلقه دوم مجدداً از j=۰ شروع شده و اعضای سطر دوم نیز به ترتیب مورد دسترسی قرار می‌گیرند. در پایان پس از مقداردهی آرایه، با استفاده از دو حلقه For تودرتو اعضای آرایه نمایش داده می‌شوند

تمرین:
پروژه‌ای طراحی کنید که بتواند نام, نام خانوادگی و نمره سه درس، پنج دانش‌آموز را در آرایه ذخیره کند و در ضمن بتوان هر لحظه اطلاعات دانش‌آموزان را مشاهده کرد.

۳-۱۱ توابع UBound و LBound

از این توابع برای دستیابی به دامنه پایینی و بالایی یک آرایه استفاده می‌شود. تابع LBound می‌تواند با دریافت نام یک آرایه، دامنه پایینی آن را به‌صورت یک عدد از نوع Long بازگرداند. به عبارت دیگر به‌وسیله این تابع می‌توانید شماره اولین اندیس را در آرایه به دست آورید. شکل کلی این تابع به این صورت است:

LBound (arrayname ,dimension)

تابع LBound یک آرگومان اجباری و یک آرگومان اختیاری دارد, آرگومان arrayname در واقع نام آرایه‌ای است که می‌خواهید اندیس آغازین آن را به دست آورید. آرگومان دوم (dimension) در آرایه‌های دو و چند بعدی استفاده می‌شود و بعدی از آرایه که می‌خواهید اندیس آغازین آن را پیدا کنید به‌صورت یک عدد صحیح تعیین می‌شود. مقدار ۱ برای بعد اول (سطر), ۲ برای بعد دوم (ستون) و الی آخر. در صورت عدم استفاده از این آرگومان, مقدار پیش‌فرض ۱ خواهد بود.

به‌عنوان مثال به این دستورات توجه کنید:

```
Dim A( 1 To 100, 10 To 13 ) As Integer
Dim B(10) As Integer
Print LBound( A, 1)
Print LBound( A, 2)
Print LBound( B)
```

در دستورات قبل یک آرایه دو بعدی (A) و یک آرایه یک بعدی (B) تعریف شده‌اند. نتیجه اجرای اولین و دومین دستور Print به ترتیب ۱ و ۱۰ خواهد بود و در صورت اجرای دستور آخر با توجه به عدم استفاده از آرگومان اختیاری و یک بعدی بودن آرایه مقدار صفر نمایش داده می‌شود. توجه داشته باشید که فرض بر این باشد که : Option Base ۰ است.

تابع UBound می‌تواند با دریافت نام یک آرایه دامنه بالایی آن را به‌صورت یک عدد از نوع Long بازگرداند به عبارت بهتر عددی را که در تعریف آرایه به‌عنوان دامنه بالایی آرایه تعریف کرده‌ایم در اختیار شما قرار می‌دهد. شکل کلی این تابع به‌صورت زیر است:

UBound (arrayname ,dimension)

این تابع نیز یک آرگومان اجباری و یک آرگومان اختیاری دارد و عملکردی را مشابه آنچه در تابع LBound دارند به عهده می‌گیرند. به‌عنوان مثال به دستورات زیر توجه کنید:

```
Dim A( 1 To 100,  10 To 13 )  As  Integer
Dim B( 10 )  As  Integer
Print  UBound( A,  1)
Print  UBound( A,  2)
Print  UBound( B )
```

اولین و دومین دستور Print به ترتیب مقادیر ۱۰۰ و ۱۳ را نمایش می‌دهند و دستور آخر نیز مقدار ۱۰ را نمایش خواهد داد.

البته باید توجه داشت که دستور Option Base تأثیری روی تابع UBound نمی‌گذارد. به‌عنوان مثال به این دستورات توجه کنید:

```
Dim  A(5) As Integer
For  i  =  LBound(A)  To  UBound(A)
     Print  i
Next  i
```

در دستورات قبل اگر Option Base 0 باشد حلقه For از مقدار صفر تا ۵ را نمایش می‌دهد و در واقع آرایه ۶ عضو خواهد داشت ولی اگر Option Base 1 باشد حلقه For از مقدار ۱ تا ۵ را نمایش می‌دهد و در نتیجه آرایه ۵ عضو خواهد داشت پس همان‌طور که دیدید دستور Option Base روی تابع UBound تأثیر نمی‌گذارد.

۴-۱۱ تابع Split

به‌وسیله این تابع می‌توان محتویات یک متغیر رشته‌ای را به‌صورت زیر رشته‌های جداگانه‌ای در یک آرایه یک بعدی ذخیره کرد. مقدار بازگشتی این تابع یک آرایه یک بعدی است. شکل کلی این تابع به‌صورت زیر است

Split (expression, delimiter, limit, compare)

این تابع دارای یک آرگومان اجباری اســت. expression می‌تواند یک عبارت رشته‌ای باشــد. آرگومان delimiter یک آرگومان اختیاری اســت و به‌طور پیش‌فرض یک کاراکتر فاصله " "اســت. در واقع این آرگومان کاراکتری را که برای جدا کردن زیر رشــته‌ها در آرگومان expression مورد استفاده قرار می‌گیرد معین می‌کند.

آرگومان limit دومین آرگومان اختیاری اســت و تعداد زیر رشــته‌های مورد نظر را که باید بازگشــت داده شوند تعیین می‌کند در صورت عدم استفاده از این آرگومان مقدار آن ۱- خواهد بود که به معنی بازگشت دادن تمام زیررشته‌هاست.

آرگومــان compare نیز اختیاری بــوده و تعیین می‌کند که آیا تابع بین حروف بزرگ و کوچک آرگومان delimiter با کاراکترهای expression تفاوت قائل شــود یا خیر. مقادیری که این آرگومان می‌تواند دریافت کند در جدول ۴-۱۱ ارایه شده‌اند.

جدول ۴-۱۱ مقادیر مربوط به آرگومان Compare

توضیح	ثابت عددی	ثابت رشته‌ای
تابع بین حروف کوچک و بزرگ تفاوت قائل می‌شود.	۰	vbBinaryCompare
تابع بین حروف کوچک و بزرگ تفاوت قائل نمی‌شود.	۱	vbTextCompare

نکته مقدار پیش‌فرض برای آرگومان compare صفر است.

برای روشن شدن بهتر مطلب به رویه رویداد زیر توجه کنید:

```
Private Sub cmdsplit_Click ()
    Dim strname As String, strresult() As String
    Dim i As Integer
    strname = "microsoft visual basic 6"
    strresult = Split(strname)
    (For i = 0 To UBound(strresult
        Print strresult(i)
```

```
Next i
End Sub
```

در دستورات فوق دستور Split براساس فضاهای خالی « » موجود در رشته strname, کلمات موجود در متغیر رشته‌ای strname را به‌صورت جداگانه در آرایه strresult ذخیره می‌کند که به‌وسیله یک حلقه For مطابق شکل ۳–۱۱ نمایش داده می‌شود.

شکل ۳–۱۱

اکنون فرض کنید تابع Split را به‌صورت زیر تغییر دهید:

```
strresult = Split ( strname , "A" )
```

در این صورت کاراکتری که برای جدا کردن زیررشته‌ها استفاده می‌شود کاراکتر A خواهد بود. اما همان‌طور که در شکل ۴–۱۱ مشاهده می‌کنید رشته strname بدون توجه به کاراکتر A بدون تغییر باقی می‌ماند چون در این دستور از آرگومان compare استفاده نشده است مقدار پیش فرض صفر است در نتیجه تابع بین کاراکتر A در آرگومان دوم با کاراکتر a در رشته strname تفاوت قائل شده است.

شکل ۴–۱۱

حال اگر دستور قبل را به‌صورت(strresult = Split (strname,»A»,1 تغییر دهید خروجی برنامه به‌صورت شکل ۵–۱۱ خواهد بود یعنی تابع بین کاراکتر A در آرگومان دوم با کاراکتر a در رشته strname تفاوتی قائل نمی‌شود و strname به سه زیر رشته تقسیم می‌شود.

در صورتی که بخواهید تعداد معینی از زیررشته‌ها را جدا کنید می‌توانید از آرگومان سوم در تابع Split استفاده کنید. مثلاً اگر مثال قبل را به این صورت تغییر دهید:

```
strresult = Split (strname,"ic",2,1)
```

خروجی برنامه مطابق شکل ۶-۱۱ خواهد بود.

شکل ۵-۱۱

شکل ۶-۱۱

نکته در صورتی که آرگومان limit در تابع Split, ۱ باشد تمام رشته expression به‌صورت یک زیر رشته بازگشت داده خواهد شد.

تمرین:
پروژه‌ای طراحی کنید که یک رشته و کاراکتر دلخواه را دریافت کند و رشته را براساس کاراکتر موردنظر به چند زیررشته تقسیم کرده و نمایش دهد.

۵-۱۱ تابع Join

عملکرد این تابع برعکس تابع Split است. به‌وسیله این تابع می‌توانید اعضای یک آرایه رشته‌ای را به‌صورت یک مقدار رشته‌ای واحد درآورید. شکل کلی این تابع به‌صورت زیر است:

Join (sourcearray, delimiter)

این تابع دارای یک آرگومان اجباری و یک آرگومان اختیاری است. آرگومان sourcearray، نام آرایه رشته‌ای مورد نظر است که می‌خواهید مقادیر موجود در آن را به‌صورت یکپارچه در یک متغیر رشته‌ای ذخیره کنید.

آرگومان delimiter یک آرگومان اختیاری است که کاراکتر جداکننده را بین کلمات در رشته‌ای که تابع بازگشت می‌دهد تعیین می‌کند. در صورت عدم استفاده از این آرگومان، کاراکتر فضای خالی " " مورد استفاده قرار می‌گیرد.

به‌عنوان مثال به دستورات زیر توجه کنید:

```
Private Sub cmdjoin_Click()
  Dim strname(3) As String, strresult As String
  strname(0) = "microsoft"
  strname(1) = "visual"
  strname(2) = "basic"
  strname(3) = "6"
 strresult = Join(strname)
  Print
  Print , strresult
End Sub
```

در صورت اجرای دستورات فوق خروجی مشابه شکل ۷–۱۱ به دست می‌آید همان‌طور که می‌بینید مقادیر موجود در آرایه strname به‌وسیله تابع Join در متغیر strresult به‌صورت یک رشته واحد ذخیره شده‌اند و از کاراکتر فاصله خالی " " برای ایجاد فاصله بین مقادیر آرایه استفاده شده است.

اکنون تابع Join را در مثال قبل به‌صورت زیر تغییر دهید:

```
strresult = Join ( strname , " # " )
```

در این صورت خروجی دستورات به‌صورت شکل ۸–۱۱ خواهد بود.

شکل ۸–۱۱

شکل ۷–۱۱

تمرین:
پـروژه‌ای طراحی کنید که تعدادی کلمه را به‌صورت جداگانه دریافت کند سـپس با استفاده از تابع Join آن‌ها را به یک رشته واحد تبدیل نموده و نمایش دهد.

۶-۱۱ نحوه ارسال آرایه‌ها به رویه‌ها

تاکنون با نحوه تعریف آرایه و چگونگی استفاده از آن‌ها آشنا شده‌اید اما گاهی اوقات لازم اسـت تا آرایه‌ها را به یک رویه فرعی یا تابع ارسال کنید. در این‌جا با ذکر یک مثال با چگونگی انجام این کار آشنا می‌شوید.

مثال ۵: می‌خواهیم تابعی بنویسـیم که با دریافت یک آرایه عددی از نوع Integer با ۱۰ عضو، کوچک‌ترین عضو را پیدا کرده و بازگرداند.

ابتدا به تعریف تابع می‌پردازیم، در اینجا لازم است آرگومان ورودی به صورت آرایه استفاده شود بنابراین در هنگام تعریف آرگومان از دو پرانتز ( ) استفاده می‌شود.

```
Public Function minelement(myarray( ) As Integer) As Integer
  Dim i As Integer, min As Integer
  min = myarray(0)
  For i = 1 To 9
     If min > myarray(i) Then min = myarray(i)
  Next i
  minelement = min
End Function
```

همان‌طـور کـه در تعریف تابع می‌بینید برای تعریف یک آرگومـان از نوع آرایه فقط کافی است دو پرانتز در ابتدای نام آرایه ذکر کنید.

برای فراخوانی این تابع نیز می‌توانید از دستور زیر استفاده کنید:

```
Print  minelement (myarray ( ) )
```

نکته آرایه‌هـا به‌طور پیش فرض به‌صورت ارسـال بـا مرجع (Call By Reference) به رویه ارسال می‌شوند.

۷–۱۱ روش‌های مرتب‌سازی آرایه‌ها

در برنامه‌نویسی واقعی معمولاً لازم است تا اطلاعات دریافت شده یا نتیجه محاسبات را به‌صورت مرتب شده در اختیار کاربران قرار دهید. به این منظور می‌توان از آرایه‌ها استفاده کرد.

تاکنون روش‌های متعددی برای مرتب کردن داده‌های موجود در یک آرایه طراحی و معرفی شده‌اند که هر یک نقاط ضعف و قدرتی نیز دارند. از انواع روش‌های معروف در مرتب‌سازی اطلاعات می‌توان به روش مرتب‌سازی حبابی (Bubble Sort)، روش مرتب‌سازی Shell Sort، روش مرتب‌سازی سریع (Quick Sort)، روش مرتب‌سازی انتخابی (Selection Sort) و مرتب‌سازی به روش درج (Insertion Sort) اشاره کرد. بعضی از روش‌های نامبرده دارای الگوریتم پیچیده‌ای هستند که از حوصله این کتاب خارج است بنابراین دو روش مرتب‌سازی حبابی و مرتب‌سازی انتخابی (Selection Sort) ارائه می‌شود.

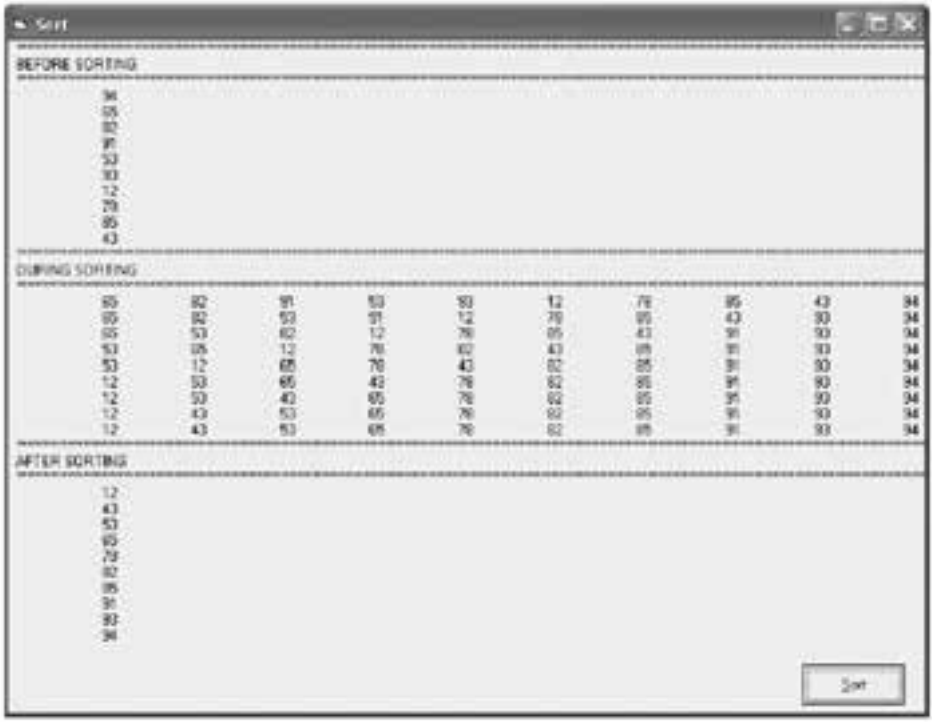
۱–۷–۱۱ روش مرتب‌سازی حبابی (Bubble Sort)

این روش ساده‌ترین روش مرتب‌سازی است و هر عضو از آرایه با اعضای بعدی آرایه مقایسه می‌شود و در صورت نیاز, عمل تعویض صورت می‌گیرد. در این روش چون هر عضو با سایر اعضای بعد از آن مقایسه می‌شود در آرایه‌های بزرگ زمان زیادی برای مرتب شدن داده‌ها مصرف می‌شود بنابراین توصیه می‌شود در صورت کوچک بودن ابعاد یک آرایه از این روش استفاده کنید. در زیر دستوراتی را می‌بینید که می‌تواند یک آرایه عددی از نوع صحیح با ۱۰عضو را که قبلاً پر شده است به روش حبابی و به‌صورت صعودی مرتب کند (با فرض این‌که Option Base 1 است).

```
Private  Sub cmdsort_Click()

    Dim  i As  Integer

    Dim  j As  Integer

    Dim  temp As  Integer
    For  i  =  1  To  9
       For  j  =  1  To  9
        If  myarray(j)  >  myarray(j  +  1)  Then
```

```
            temp = myarray(j)
            myarray(j) = myarray( j + 1)
            myarray(j + 1) = temp
        End If
    Next j
Next i
End Sub
```

در ایـــن رویداد عمل مرتب‌ســـازی با دو حلقه که n – ۱ بار تکرار می‌شـــوند (n تعداد اعضای آرایه است) انجام می‌شود. در واقع به‌وسیله یک If در داخل حلقه دوم مقدار هر عضو با عضو بعدی مقایســـه می‌شود و در صورت بزرگ‌تر بودن عضوی که اندیس آن j اســـت مقدار آن با مقدار عضوی که اندیس آن j+1 اســـت تعویض می‌شود. با ادامه روند عملیات، این کار آنقدر تکرار می‌شـــود تا آرایه مرتب شود؛ می‌توانید روند انجام عملیات را در شکل ۹–۱۱ مشاهده کنید: در ابتدا، مقادیر موجود در آرایه را قبل از مرتب شدن و در انتها، همان آرایه را به‌صورت مرتب شـــده می‌بینید در بخش میانی نیز اعضای آرایه را در هر بار اجرای حلقه اول (حلقه با شمارنده i) مشاهده می‌کنید. اگر به روند تغییر مکان اعداد توجه کنید نحوه مرتب‌سازی را کاملاً درک خواهید کرد.



شکل ۹–۱۱

تمرین:

پـــروژه‌ای طراحی کنید که ده عدد را از کاربر دریافت کند و پس از ذخیره‌ســـازی در یک آرایه، آرایه را به‌صورت نزولی مرتب کرده و نمایش دهد.

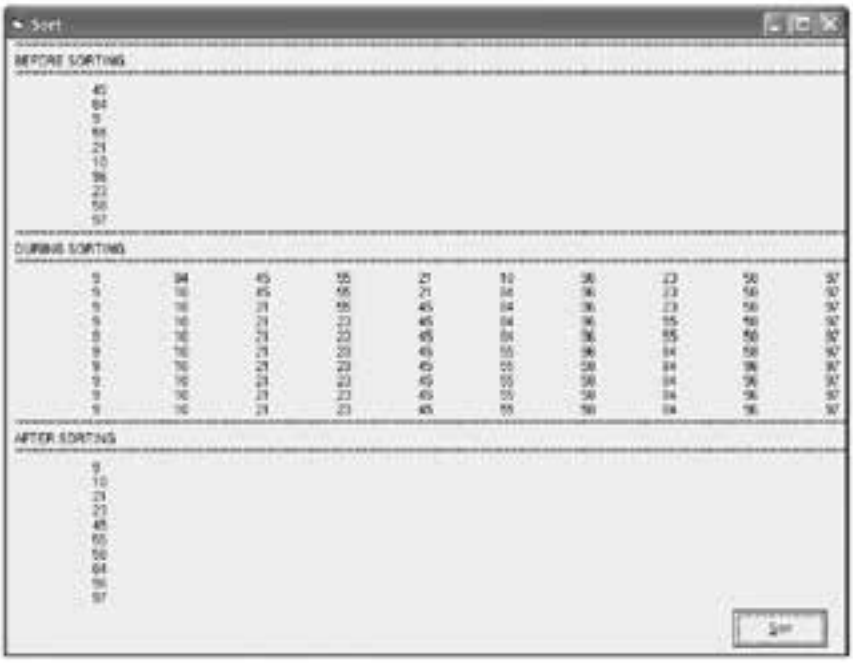
۲–۷–۱۱ روش مرتب‌سازی انتخابی

در این روش ابتدا کوچک‌ترین عضو آرایه تعیین و با عضو اول آرایه جابه‌جا می‌شود. در مرحله بعد کوچک‌ترین عضو آرایه در بین عضو دوم تا آخرین عضو آرایه پیدا شده و با عضو دوم آرایه جابه‌جا می‌شود و به همین ترتیب الی آخر تا آرایه مرتب شود.

```
Private Sub cmdsort_Click()
    Dim length As Integer
    Dim index As Integer
    Dim min As Long
    length = UBound(myarray)
    For i = 1 To length
        min = myarray(i)
        index = i
        For j = i + 1 To length
            If min > myarray(j) Then
                min = myarray(j)
                index = j
            End If
        Next j
    If index < > i Then
        temp = myarray(i)
        myarray(i) = myarray(index)
        myarray(index) = temp
      End If
      Next i
End Sub
```

در این رویداد با استفاده از دو حلقه For تودرتو عملیات مرتب‌سازی انجام می‌شود در ابتدا نخستین حلقه For اولین عضو آرایه را به عنوان کوچک‌ترین مقدار در متغیر min ذخیره می‌کند. سپس حلقه For دوم این مقدار را با سایر اعضا یعنی اعضای دوم تا آخرین عضو مقایسه می‌کند اگر مقدار عضوی از مقدار min کمتر باشد مقدار کوچک‌تر را در متغیر min و اندیس آن‌را در متغیر index ذخیره می‌کند و پس از خاتمه اجرای این حلقه اگر مقدار کوچک‌تری پیدا شده باشد (index <> i) آنگاه با استفاده از یک دستور If جای عضو اول با عضوی کوچک‌تر که اندیس آن در متغیر index ذخیره شده است عوض می‌شود.

در این مرحله با رسیدن Next i اجرای دستورات به ابتدای حلقه اول منتقل می‌شود. در این مرحله همان عملیات برای عضو دوم آرایه اجرا خواهد شد و به همین شکل آرایه مرتب می‌شود (شکل ۱۰-۱۱).



شکل ۱۰-۱۱

۸-۱۱ روش‌های جستجوی داده‌ها در آرایه‌ها

گاهی اوقات لازم است تا داده‌ای را میان مجموعه‌ای از اطلاعات موجود جستجو

کرده و مورد دستیابی قرار دهیم. یکی از دلایل استفاده از آرایه‌ها جستجوی سریع‌تر و راحت‌تر داده‌ها در میان انبوهی از اطلاعات است.

جستجوی داده در یک آرایه مانند مرتب‌سازی از روش‌های مختلفی امکان‌پذیر است که از مهم‌ترین آن‌ها می‌توان روش جستجوی خطی و روش جستجوی دودویی را نام برد.

۱–۸–۱۱ روش جستجوی خطی (Linear Search)

این روش یکی از ساده‌ترین روش‌های جستجو است در این روش برای پیدا کردن یک مقدار، مقدار مربوطه را یک به یک با اعضای آرایه مورد مقایسه قرار داده و در صورت پیدا شدن اطلاعات مورد نظر جستجو خاتمه می‌یابد و اگر مقدار مورد نظر در هیچ یک از اعضای آرایه پیدا نشود در آن صورت نتیجه جستجو منفی خواهد بود و مقدار مربوطه در آرایه وجود نخواهد داشت.

به‌عنوان مثال فرض کنید می‌خواهیم یک عدد را در آرایه‌ای با ۲۰ عضو جستجو کنیم. تابع mysearch می‌تواند با دریافت یک عدد و آرایه مربوطه، عدد مربوط را در آرایه جستجو کند و در صورت وجود عدد در آرایه، مقدار Yes و در غیر این صورت مقدار No را برگرداند.

```
Function mysearch(myarray() As Integer,mynumber As Integer) _ As  Boolean
  Dim  i  As  Integer
  For  i  =  0  To  19
       If  mynumber  =  myarray(i)  Then
            mysearch  =  True
            Exit  Function
       End  If
       mysearch  =  False
   Next  i
  End  Function
```

در این دستورات با استفاده از حلقه For و یک دستور If تک‌تک اعضای آرایه با مقدار متغیر mynumber مقایسه می‌شوند و در صورت پیدا شدن مقدار مشابه، مقدار درست (True) برگشت داده می‌شود و پس از آن به‌وسیله Exit Function خروج از تابع اتفاق می‌افتد.

اما اگر تمام اعضای آرایه بررسی شوند و مقداری برابر با mynumber پیدا نشود با بازگشت دادن مقدار نادرست (False) این مسأله را مشخص می‌کند.

به‌عنوان مثالی دیگر فرض کنید می‌خواهیم یک نام را در یک آرایه رشته‌ای با ۲۰ عضو مورد جستجو قرار دهیم، در این‌صورت تابع mysearch به این صورت در می‌آید:

```
Function mysearch(myarray( ) As String, myname As String) As Boolean
  Dim i As Integer
  For i =  0 To 19
      If StrComp(myarray(i), myname, 1) Then
          mysearch = True
          Exit Function
      End If
      mysearch = False
   Next i
End Function
```

در تابع فوق نیز نحوه انجام عملیات مانند تابع قبل است و فقط برای پیدا کردن نام مورد نظر (myname) در آرایه از تابع StrComp استفاده می‌شود.

تمرین:

پروژه‌ای طراحی کنید که ده عدد را که به‌صورت تصادفی تولید شده‌اند در آرایه قرار دهد و به کاربر نشان دهد سپس کاربر عدد دلخواهی را در یک کنترل کادر متن وارد کند و با کلیک روی دکمه فرمان Search به روش جستجوی خطی به جستجوی عدد موردنظر بپردازد.

۲-۸-۱۱ روش جستجوی دودویی (Binary Search)

روش جستجوی خطی در آرایه‌های بزرگ، بسیار کند و وقت‌گیر است، روش جستجوی دو دویی با استفاده از این راه‌حل ساده، تعداد مقایسه‌ها را به مقدار زیادی کاهش می‌دهد.

در این روش ابتدا آرایه را مرتب کرده سپس عضو میانی آرایه با مقدار مورد جستجو، مقایسه می‌شوند، اگر مقدار مورد جستجو کوچک‌تر از عضو میانی باشد جستجو در

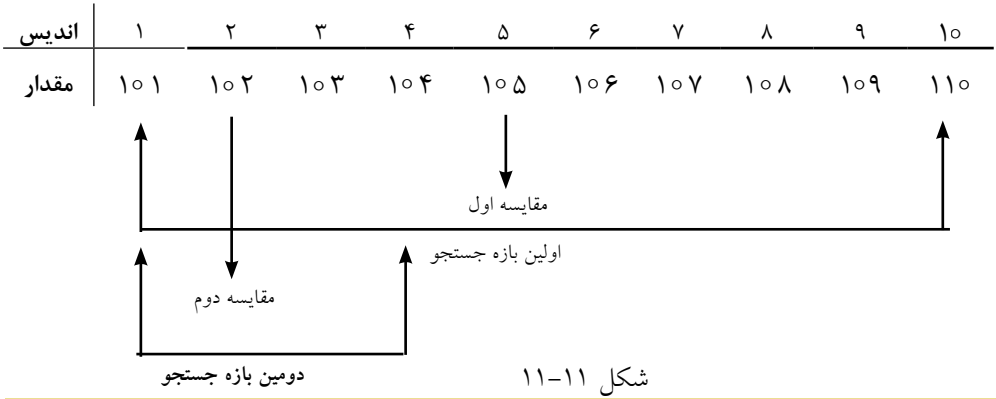
قسمت پایینی عضو میانی صورت می‌گیردو اگر مقدار مورد جستجو بزرگ‌تر از عضو میانی باشد جستجو در قسمت بالایی عضو میانی انجام می‌گیرد سپس همین اعمال برای نیمه پایینی و یا بالایی در آرایه انجام می‌شود و بدین صورت تا زمان پیدا شدن مقدار مورد نظر یا با نصف شدن تعداد اعضای باقیمانده عملیات ادامه می‌یابد.

بنابراین می‌توانید تابع Binary Search را به این صورت بنویسید:

```
Public Function mysearch(myarray() As Integer,mynum As Integer) As Boolean
    Dim i As Integer, intlow As Integer
    Dim inthigh As Integer, intmid As Integer
    intlow = LBound(myarray)
    inthigh = UBound(myarray)
  Do While (inthigh >= intlow)
    intmid = Int((intlow + inthigh) / 2)
    If mynum = myarray(intmid) Then
      mysearch = True
      Exit Function
  ElseIf myarray(intmid) < mynum Then
    intlow = intmid + 1
  Else
    inthigh = intmid - 1
  End If
  Loop
  mysearch = False
End Function
```

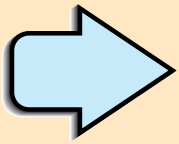
ایـــن تابـــع با دریافت یک آرایه و عدد موردنظر برای جســـتجو تعریف می‌شـــود پس از تعریـــف متغیرهای مورد نظر، مقدار اندیس بالایی و پایینی در آرایه را به‌وســـیله توابع UBound، LBound در متغیرهای inthigh و intlow قرار می‌دهد. سپس یک حلقه Do while عملیات جستجو را انجام می‌دهد. ابتدا مقدار اندیس میانی در آرایه محاسبه می‌شود برای این کار اندیس پایینی با بالایی جمع شـــده و مقدار صحیح تقســـیم آن‌ها بر عدد ۲ توسط تابع Int به دست می‌آید.

فرض کنید که آرایه دارای ۲۰ عضو است و اندیس پایینی ۱ است. با این شرایط مقدار intmid برابر با ۵ خواهد بود. پس از محاسبه مقدار اندیس عضو میانی به‌وسیله فرمان if تساوی عضو میانی myarray (5) با عدد مورد جستجو mynum بررسی می‌شود، در صورت مساوی بودن آن‌ها با یکدیگر مقدار True بازگشت داده خواهد شد و به‌وسیله فرمان Exit Function اجرای برنامه به محل فراخوانی منتقل می‌شود اما اگر تساوی برقرار نباشد شرط موجود در ElseIf بررسی می‌شود. اگر در این مرحله مقدار عضو myarray (5) کوچک‌تر از مقدار mynum باشد چون آرایه مرتب است بنابراین مقدار اندیس پایینی به‌وسیله فرمان intlow = intmid +1 به مقدار بعد از اندیس عضو میانی (یعنی ۶) تغییر پیدا خواهد کرد و حلقه در اجرای دوباره دستورات، عناصر موجود در بین اندیس۶ و ۱۰را مورد جستجو قرار می‌دهد؛ اما اگر شرط موجود در ElseIf غلط باشد بنابراین عنصر مورد جستجو در نیمه پایینی آرایه قرار دارد و با استفاده از بخش Else مقدار اندیس بالایی یکی‌کمتر از اندیس میانی (یعنی ۴) خواهد شد و در اجرای دوباره حلقه، عناصر موجود در بین اندیس ۱ و ۴ مورد جستجو قرار می‌گیرند، به این ترتیب با کوچک‌تر شدن بازه جستجو، در صورتی که مقدار مورد نظر در آرایه کشف نشود مقادیر inthigh و intlow به‌گونه‌ای تغییر می‌کنند که نتیجه شرط inthigh > = intlow، نادرست شود که در نتیجه اجرای حلقه خاتمه یافته و مقدار False بازگشت داده می‌شود مثلاً فرض کنید که می‌خواهیم عدد ۲۰را در آرایه ۱۰ عضوی (مقدار اعضای آن از ۱۰۱ تا ۱۰ هستند) جستجو کنیم. عملکرد تابع به این صورت نمایش داده می‌شود:



شکل ۱۱-۱۱

تمرین:
پروژه تمرین قبل را با روش جستجوی دودویی انجام دهید.



Learn in English

Array Variables

Much of the time, you just want to assign a single value to a variable you've declared.

A variable containing a single value is a scalar variable.

Other times, it's convenient to assign more than one related value to a single variable.

Then you can create a variable that can contain a series of values.

This is called an array variable. Array variables and scalar variables are declared in the same way,

except that the declaration of an array variable uses parentheses ( ) following the variable name.

In the following example, a single–dimension array containing 11 elements is declared:

```
Dim A(۱۰) As Integer
```

واژه‌نامه

Array	آرایه
Binary Search	جستجوی دودویی
Bound	مرز، محدوده
Bubble Sort	مرتب‌سازی حبابی
Convenient	راحت، مناسب
Delimiter	حائل
Dimension	اندازه، بعد
Dynamic	پویا
Element	عنصر
Erase	پاک‌کردن
Expression	عبارت
Join	متصل کردن
Limit	حد، اندازه
Linear Search	جستجوی خطی
Random	تصادفی
Related	مربوط
Scalar	عددی
Selection Sort	مرتب‌سازی انتخابی
Split	از هم جدا کردن
Subscript	اندیس، زیرنویس

خلاصه مطالب

- از آرایه‌ها برای ذخیره‌سازی و نگهداری آسان‌تر داده‌ها، مرتب‌سازی و جستجوی سریع اطلاعات استفاده می‌شود.
- آرایه‌ها به دو دسته کلی، آرایه‌ها با ابعاد ثابت و آرایه‌های پویا یا با ابعاد متغیر تقسیم می‌شوند.
- به‌وسیله فرمان Option Base می‌توان شماره اولین عنصر آرایه را تعیین کرد.
- در آرایه‌های پویا (Dynamic) تعداد اعضای آرایه با توجه به تعداد مورد نیاز معین می‌شوند اما در آرایه‌های ثابت تعداد اعضای آرایه بعد از تعریف آن قابل تغییر نیست.
- برای قابل استفاده شدن یک آرایه پویا پس از تعریف آرایه از دستور ReDim برای تعیین ابعاد آن استفاده می‌شود.
- در صورتی که بخواهید مقادیر موجود در یک آرایه پویا در زمان تغییر ابعاد آن از بین

نرود از کلمه کلیدی Preserve استفاده کنید.

• به‌وسیله فرمانErase می‌توان آرایه‌های پویا را حذف کرد.

• مقادیر موجود در یک آرایه با ابعاد ثابت به‌وسیله فرمانErase از بین خواهند رفت اما فضاهای مربوط به اعضای آرایه در حافظه باقی می‌مانند.

• هنگام ارسال آرایه‌ها به رویه‌ها به‌طور پیش‌فرض از حالت ارسال با مرجع استفاده می‌شود.

• روش‌های مرتب‌سازی آرایه‌ها عبارتند از : مرتب‌سازی حبابی، مرتب‌سازی به روشShell Sort، روش مرتب‌سازی Quick Sort، روش مرتب‌سازی انتخابی (Selection Sort) و روش مرتب‌سازی Insertion Sort.

• برای‌جستجوی‌اطلاعات دریک‌آرایه می‌توان از روش جستجوی خطی یا دودویی استفاده کرد.

• توابع UBound و LBound به ترتیب دامنه بالایی و پایینی آرایه مورد نظر را محاسبه می‌کنند.

• تابع Split می‌تواند یک عبارت رشته‌ای را به‌صورت زیر رشته‌های جداگانه در یک آرایه یک بعدی ذخیره کند.

• تابع Join برخلاف تابع Split، می‌تواند مقادیر رشته‌ای موجود در یک آرایه رشته‌ای را به‌صورت یک عبارت رشته‌ای واحد درآورد.

آزمون نظری

۱ –در ویژوال‌بیسیک اندیس اول یک آرایه به‌طور پیش‌فرض از چه شماره‌ای آغاز می‌شود؟

الف– ۱–　　ب– صفر　　ج– ۱　　د– ۲

۲ – پس از تعریف یک آرایه پویا به‌وسیله کدام دستور آرایه قابل استفاده می‌شود؟

الف– Dim　　ب– Static　　ج– Erase　　د– ReDim

۳ – به‌وسیله کدام فرمان می‌توان شماره اندیس اولین عضو آرایه را تعیین کرد؟

الف– Option Compare Text　　ب– Option Explicit

ج– Option Base　　د– Option Keyword

۴ – آرایه (۵ ,۱۰) No چند عضو دارد؟ (Option Base 1)

الف– ۳۶　　ب–۵۰　　ج– ۴۵　　د– ۴۰۰

۵ – فرمان Erase می‌تواند آرایه‌های ..... را حذف کند.

الف– پویا　　ب– با ابعاد ثابت

ج– رشته‌ای　　د– گزینه‌های الف و ب صحیح هستند.

۶ – اگر (۱۰) name باشد حاصل عبارت (name) UBound چیست؟

الف– ۸　　ب– ۹　　ج–۱۰　　د– ۱۱

۷ – آرایه (1 To 5, 3 To 6) Color چند عضو دارد؟

الف–۲۰　　ب– ۲۵　　ج–۳۰　　د– ۳۵

۸–کدام تابع می‌تواند اعضای یک‌آرایه رشته‌ای را به یک عبارت رشته‌ای واحد تبدیل کند؟

الف– Split　　ب– Join　　ج– Ubound　　د– Preserve

۹-در صورتی که بخواهیم در زمان تغییر ابعاد یک آرایه پویا مقادیر قبلی آرایه از بین نرود از کدام کلمه کلیدی همراه با دستور ReDim استفاده می‌کنیم؟

الف– Dim　　ب– ParamArray　　ج– Para　　د– Preserve

۱۰ – کدام تابع می‌تواند یک عبارت رشته‌ای را به‌صورت کلمات جداگانه در یک آرایه ذخیره کند؟

الف– Split　　ب– Join　　ج– Preserve　　د– Filter

۱۱ -Which of the following answer can be use to create a single dimension array With 5 elements?

a- Dim no (5) As Integer　　b- Dim no (4) As Integer

c- Dim no [5] As Integer　　d- Dim no [4] As Integer

۱۲- آرایه را تعریف کنید و انواع آن را نام ببرید.
۱۳ - دستور Erase، Split و Join را با ذکر مثال توضیح دهید.
۱۴ - نحوه تعریف و استفاده از آرایه‌های استاتیک و پویا را با ذکر مثال بیان کنید.
۱۵ - توابع UBound و LBound را با ذکر مثال توضیح دهید.
۱۶ - نحوه مرتب سازی آرایه‌ها با روش حبابی و انتخابی را بیان کنید.
۱۷ - کاربرد دستور Option Base را با ذکر مثال توضیح دهید.
۱۸ - نحوه ارسال آرایه‌ها به رویه‌های تابعی و فرعی را بیان کنید.
۱۹ - انواع روش‌های جستجوی داده‌ها در آرایه‌ها را توضیح دهید.

آزمون عملی

۱ – رویه‌ای بنویسید که یک آرایه ۵×۵ را با اعضای تصادفی ایجاد کرده سپس مجموع هر سطر و هر ستون آرایه را به‌طور جداگانه به همراه خود آرایه نمایش دهد.
۲ – رویه‌ای بنویسید که دو آرایه دلخواه را دریافت کرده و آرایه حاصل‌ضرب آن دو را به همراه دو آرایه نمایش دهد.
۳ – رویه‌ای بنویسید که دو آرایه یک بعدی دلخواه را دریافت کرده و سپس آرایه حاصل جمع، حاصل‌ضرب و حاصل تفریق آن دو را در آرایه‌های جداگانه‌ای ذخیره کند.
۴ – رویه‌ای بنویسید که یک آرایه عددی با ۲۰۰ عضو را به‌صورت نزولی مرتب کرده و نمایش دهد.

واحد کاردوازدهم

هدف جزیی

توانایی استفاده از جلوه‌های گرافیکی و چاپ در ویژوال بیسیک

هدف‌های رفتاری

پس از مطالعه این واحد کار از فراگیر انتظار می‌رود که:

۱- مفاهیم مربوط به سیستم مختصات را بداند.

۲- توانایی تغییر مختصات را به منظور انجام ترسیمات داشته باشد.

۳- توانایی به‌کارگیری متدهای گرافیکی زیر را داشته باشد:

PSet , Line , Circle , Point , Cls , Print , TextHeight , TextWidth

۴- توانایی استفاده از خصوصیات گرافیکی زیر را داشته باشد:

CurrentX , CurrentY , AutoRedraw , DrawMode , DrawStyle , DrawWidth , FillStyle

۵- توانایی استفاده از توابع QBColor و RGB را داشته باشد.

۶- توانایی استفاده از امکانات چاپ، خصوصیات و متدهای آن را در برنامه‌ها داشته باشد.

۷- توانایی استفاده از امکانات چندرسانه‌ای و کنترل MCI و MAM

۸- توانایی استفاده از شئ Picture و رویه‌های SavePicture و LoadPicture

تاکنون چگونگی طراحی و ساخت رابط‌های گرافیکی را با استفاده از فرم‌ها و بعضی از کنترل‌هـا آموختید؛ اما گاهی اوقـات وجود فرم‌ها و کنترل‌ها بدون جلوه‌های گرافیکی محیط نرم‌افزار را خسته‌کننده و غیر قابل استفاده می‌کند. با استفاده از جلوه‌های گرافیکی نظیر رنگ، نمودارهای گرافیکی و تصاویر متحرک می‌توانید به کاربر در مشاهده و درک بهتر گزارشات و نتیجه محاسبات کمک کنید.

ویژوال‌بیسـیک قابلیت بالایی در اسـتفاده از جلوه‌های گرافیکی در اختیار شـما قرار می‌دهد و می‌توانید این جلوه‌های گرافیکی را با دو روش ایجاد کنید: کنترل‌های گرافیکی مثل Line ,Shape و متدهای گرافیکی مثل Line , PSet,Circle و ...

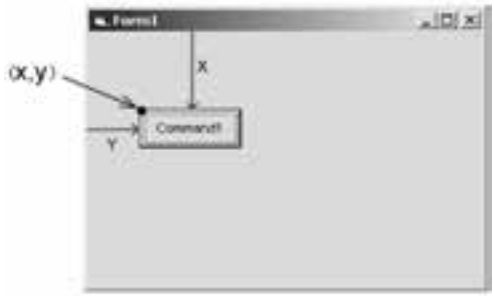
۱-۱۲ مفهوم سیستم مختصات در ویژوال‌بیسیک

در ویژوال بیسیک نیز مانند ریاضیات و هندسه برای انجام هر نوع ترسیمی‌از سیستم مختصات استفاده می‌شود، هم‌چنین از دو بعد یا محور مختصات برای تعیین موقعیت ترسیمات استفاده می‌شود (شکل ۱-۱۲).

شکل ۱-۱۲

مختصـات هر نقطه به‌صـورت (X,Y) تعیین می‌شـود مقدار X موقعیـت نقطه را در طـول محـور X و مقـدار Y موقعیـت نقطه را در طول محـور Y بیان می‌کنـد که مقدار شـروع در هر یک از محورها صفر اسـت. دقت داشـته باشـید که محورهای مختصات در ویژوال‌بیسـیک بـا محورهـای مختصـات در ریاضیات متفاوت اسـت. در شـکل ۱-۱۲ نحـوه قرار گرفتن محورها بر نقطه مبنا قابل مشـاهده اسـت. مختصات نقطه مبنا (۰,۰) است که در گوشه بالا و چپ شئ مربوطه که معمولاً فرم است قرار دارد.

وقتی کنترلی را جابه‌جا کرده یا اندازه آن را تغییر می‌دهید, از سیستم مختصات فرمی‌که کنترل در آن قرار دارد, استفاده می‌کنید. در واقع کنترل‌ها و هرگونه ترسیمات گرافیکی از سیستم مختصات شئ که در آن رسم می‌شوند, تبعیت می‌کنند. اگر کنترلی در روی فرم قرار گیرد موقعیت و اندازه کنترل از سیستم مختصات فرم پیروی می‌کند و اگر خطی در یک جعبه تصویر PictureBox رسم شود، موقعیت و اندازه خط از سیستم مختصات کنترل جعبه تصویر استفاده می‌کند.



شکل ۲-۱۲ تعیین موقعیت یک کنترل به وسیله سیستم مختصات

برای تعریف موقعیت ترسیمات گرافیکی به‌وسیله محورها از واحدهای اندازه‌گیری استفاده می‌شود که به آن مقیاس می‌گویند. در ویژوال‌بیسیک هر محور در سیستم مختصات می‌تواند مقیاس خاص خود را داشته باشد.

۲-۱۲ تغییر سیستم مختصات

می‌توانید سیستم مختصات یک شئ خاص مثل فرم را به‌وسیله خصوصیت Scale یا متد Scale روی مقادیر مورد نظرتان تنظیم کنید. برای انجام این کار می‌توانید از مقیاس پیش‌فرض استفاده کرده یا یکی از مقیاس‌های استاندارد را انتخاب کنید یا این که یک مقیاس جدید را ایجاد کنید. با تغییر مقیاس سیستم مختصات می‌توانید اندازه و موقعیت ترسیمات گرافیکی را روی فرم با توجه به نیازتان به آسانی تنظیم کنید.

هر فرم یا بعضی از کنترل‌ها مانند PictureBox چندین خصوصیت Scale، نظیر ScaleMode، ScaleWidth، ScalHeight، ScaleTop، ScaleLeft و یک متد Scale دارند که به‌وسیله آن‌ها می‌توانید سیستم مختصات خود را تعریف کنید. مقیاس پیش‌فرض twip است.

همان‌طور که قبلاً هم اشاره کردیم هر twip ۵۶۷ برابر با یک سانتی‌متر است.

برای انتخاب یک مقیاس استاندارد می‌توانید یکی از مقادیر موجود در جدول ۱-۱۲ را برای خصوصیت ScaleMode فرم یا کنترل مورد نظر خود در نظر بگیرید.

جدول ۱-۱۲ مقادیری که خصوصیت ScaleMode کسب می‌کند.

ثابت رشته‌ای	ثابت عددی	توضیح (نوع مقیاس)
vbUser	۰	مقادیر تعریفی کاربر در خصوصیات ScaleHeight، ScaleWidth، ScaleTop، ScaleLeft استفاده می‌شوند.
vbTwips	۱	twip
vbPoints	۲	Point (۷۲Point = ۱Inch)
vbPixels	۳	Pixel (یک pixel کوچک‌ترین واحد نمایشی در صفحه نمایش یا چاپگر است و تعداد آن‌ها در هر اینچ به مقدار وضوح تصویر بستگی دارد.)
vbCharacters	۴	Character
vbInches	۵	Inch
vbMillimeters	۶	Milimeter
vbCentimeters	۷	Centimeter

اگر بخواهید مختصات نقطه مبنا را تغییر دهید یا مقیاس جدید را در یک کنترل یا فرم ایجاد کنید می‌توانید از خصوصیات ScaleHeight، ScaleWidth، ScaleTop و ScaleLeft استفاده کنید. خصوصیات ScaleLeft و ScaleTop با دریافت مقادیر عددی، مختصات نقطه مبنا را در کنترل و فرم معین می‌کنند. مقدار پیش‌فرض برای این دو خصوصیت صفر است. مقدار این خصوصیات را می‌توانید از طریق پنجره خصوصیات تغییر دهید و یا با استفاده از نوشتن کد در رویدادها و رویه‌های مورد نظر این کار را انجام دهید. شکل کلی استفاده از این خصوصیات به‌صورت زیر است:

```
object. ScaleLeft  = value
object. ScaleTop  = value
```

منظور از object نام یک فرم یا کنترل است و استفاده از آن اختیاری است و در صورتی‌که از آن استفاده نشود فرمی که فوکوس دارد در نظر گرفته خواهد شد. value یک مقدار عددی است که مختصات نقطه مبنا را تعیین می‌کند در صورت

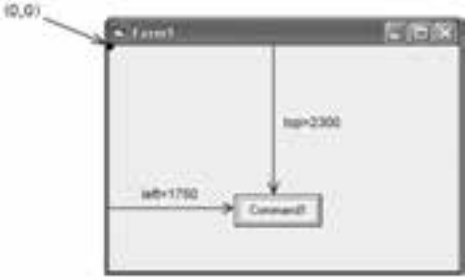
عدم استفاده از value می‌توانید مقدار فعلی مختصات نقطه مبنا را به‌دست آورید. به‌عنوان مثال فرض کنید در یک فرم همراه با یک کنترل دکمه فرمان این مقادیر تنظیم شده است:

```
Form1. ScaleMode = 1
Command1.Top = 2300
Command1.Left = 1750
```

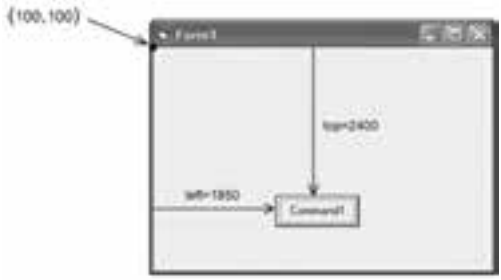
اگر بخواهیم در این فرم مختصات نقطه مبنا را (100, 100) قرار دهیم در این صورت خصوصیات مربوطه را به‌صورت زیر تنظیم می‌کنیم:

```
Form1.ScaleTop = 100
Form1.ScaleLeft = 100
```

در این صورت مقدار خصوصیت ScaleMode در Form۱ به‌طور خودکار روی مقدار صفر قرار می‌گیرد و در ضمن خصوصیت Top و Left در دکمه فرمان به‌طور خودکار به ترتیب روی مقادیر ۲۴۰۰ و ۱۸۵۰تنظیم می‌شوند و همان‌طور که از مقایسه مقادیر در دو حالت مشاهده می‌کنید مقادیر (x,y) برای کنترل دکمه فرمان با توجه به نقطه مبنای جدید (۱۰۰ , ۱۰۰) تنظیم می‌شود. این دو حالت را می‌توانید در شکل‌های ۳-۱۲ و ۴-۱۲ مشاهده کنید.



شکل ۳-۱۲



شکل ۴-۱۲

به علاوه به‌وسیله این دو خصوصیت می‌توانید از مقدار مختصات نقطه مبنا مطلع شوید در رویه بعد، پس از تغییر مقدار این دو خصوصیت، مقادیر جدید در یک کادر پیغام نمایش داده می‌شوند.

```
Private Sub  cmdshow_Click ()
   Form1. ScaleLeft = 150
   Form1. ScaleTop = 180
   MsgBox  " ScaleLeft = " + Str ( ScaleLeft ) + " ScaleTop = " + Str ( ScaleTop )
End Sub
```

نکته در صورت عدم استفاده از بخش object، فرمی که فوکوس دارد در نظر گرفته خواهد شد.

عـلاوه بر تغییـر مختصات نقطه مبنا، می‌توانید مقیاس سیسـتم مختصات را نیز تغییر دهیـد. خصوصیـات ScaleWidth و ScaleHeight واحد اندازه‌گیری در محورهای X و Y را تعیین می‌کنند. مقدار این خصوصیات را می‌توانید از طریق پنجره خصوصیات یا با نوشتن کد مناسب تغییر دهید. شکل کلی استفاده از این خصوصیات به‌صورت زیر است:

```
object. ScaleHeight  = value

object. ScaleWidth  = value
```

منظور از object نام یک فرم یا کنترل اسـت و value یک مقدار عددی اسـت که واحد اندازه‌گیری را در محورها تعیین می‌کند. در صورتی که از object اسـتفاده نشود، فرمی‌که فوکوس دارد در نظر گرفته خواهد شد و در صورت عدم استفاده از بخش value می‌توانید مقادیر ذخیره شده در این دو خصوصیت را به‌دست آورید.

به‌عنوان مثال به رویه زیر توجه کنید:

```
Private  Sub  cmdscale_Click( )

Form1.ScaleWidth  =  1000

Form1.ScaleHeight  =  500

End  Sub
```

با اجرای رویه فوق معیار اندازه‌گیری در محور افقی (X)، یک هزارم($\frac{۱}{۱۰۰۰}$)عرض داخلی (Width) فـرم و در محـور عمودی (Y)، یک پانصـدم ($\frac{۱}{۵۰۰}$) ارتفاع داخلی (Height) فرم خواهد بود.

نکته خصوصیات ScaleWidth و ScaleHeight واحدها را با توجه به ابعاد داخلی فرم یا شـئ مربوطه تعیین می‌کنند. این ابعاد شـامل حاشـیه‌ها یا منوها یا نوار عنوان نمی‌شـوند. این دو خصوصیت همواره در رابطه با بخش قابل ترسیم داخل فرم یا شئ مربوطه تعریف می‌شوند.

به‌عنوان مثال یک فرم همراه با یک دکمه فرمان با مشخصات زیر را ایجاد کنید:

```
Form1.BorderStyle  =  None
```

```
Form1.Height = 3000
Form1.Width = 4200
Form1.ScaleMode = twip
Command1.Height = 400
Command1.Width = 1200
```

اگــر تنظیمات فوق را به‌ترتیب انجام دهید ســپس مقــدار خصوصیات ScaleHeight و ScaleWidth را ملاحظه کنید خواهید دید که مقدار این دو خصوصیت مانند مقدار ارتفاع و عرض فرم اســت زیرا فرم شــما بدون حاشیه و نوار عنوان اســت و تمام فضای فرم، فضای قابل دســترس محسوب می‌شود. اما اگر مقدار خصوصیت BorderStyle را Sizeable قرار دهیــد مقدار دو خصوصیــت ScaleHeight و ScaleWidth مقادیر کمتری خواهند بود زیرا بخشــی از ابعاد فرم به نوار عنوان و حاشــیه‌ها داده شده است که جزء فضاهای قابل دســترس نیســتند. هم‌چنین معیار اندازه‌گیری در محورها توسط خصوصیت ScaleMode تعیین می‌شود که از نوع twip است.

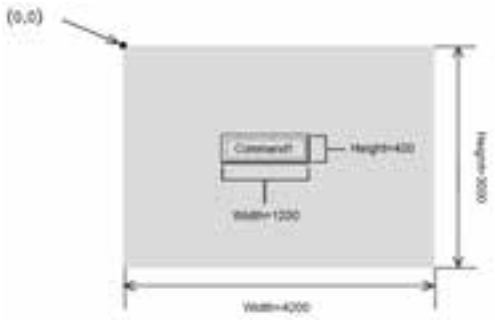
اکنــون خصوصیات ScaleHeight و ScaleWidth فرم را روی ۵۰۰ و ۱۰۰۰تنظیم کنید با تغییر یکی از این خصوصیات، خصوصیت ScaleMode به‌طور خودکار روی مقدار User-0 تنظیم می‌شــود پس از تغییــر دو خصوصیت فوق، خصوصیــات Height و Width کنترل Command1 را مشــاهده کنید. همان‌طور که می‌بینید مقــدار آن‌ها به ترتیب به ۶۶/۶۶۷ و ۲۸۵/۷۱۴ تغییر کرده اســت. در واقع برای محاســبه هر یک از این مقادیر به این صورت عمل شده است (شکل ۵-۱۲ و ۶-۱۲).

مقدار خصوصیت Command1.Height

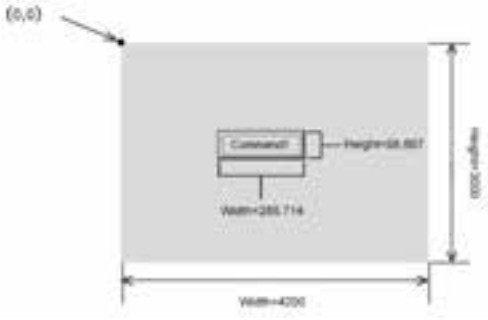
$$۴۰۰ \times \frac{۵۰۰}{۳۰۰۰} = ۶۶/۶۶۷$$

مقدار خصوصیت Command1.Width

$$۱۲۰۰ \times \frac{۱۰۰۰}{۴۲۰۰} = ۸۵/۷۱۴$$



شکل ۵-۱۲



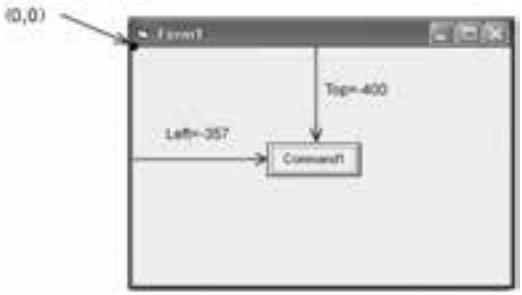
شکل ۶-۱۲

نکته • تنظیم هر یک از خصوصیات Scale، مقدار خصوصیت ScaleMode را به‌طور خودکار روی ° تنظیم می‌کند.

• انتخاب مقیاس بزرگ‌تر از صفر برای خصوصیت ScaleMode مقدار خصوصیات ScaleHeight و ScaleWidth را به‌طور خودکار روی مقادیر جدید تنظیم می‌کند و مقدار خصوصیات ScaleTop و ScaleLeft را روی صفر تنظیم می‌کند.

هر چهار خصوصیت Scale می‌توانند مقادیر اعشاری و حتی منفی داشته باشند. در صورت استفاده از اعداد منفی برای خصوصیات ScaleWidth و ScaleHeight جهت محورها در سیستم مختصات تغییر می‌کند، مثلاً برای فرم و کنترلی با مشخصات زیر فرمی مطابق شکل ۷–۱۲ خواهید داشت:

```
Form1.ScaleWidth  = - 1000
Form1.ScaleHeight  = - 1000
Form1.ScaleLeft  = 0
Form1.ScaleTop  = 0
Command1.Left  = - 357
Command1.Top  = - 400
```



شکل ۷–۱۲

به‌عنوان آخرین مثال در رابطه با دو خصوصیت ScaleWidth و ScaleHeight به این رویه توجه کنید:

```
Private Sub cmdscale_Click( )
   Form1.ScaleMode  = 1
   Form1.Width  = 4200
   Form1 height  = 3000
   Print "Form1.Width  = "  Form1.Width, "Form1.Height  = "  Form1.Height
   Print "Form1.ScaleWidth  = " Form1.ScaleWidth
   Print "Form1.ScaleHeight  = " Form1.ScaleHeight
   Print Form1.ScaleWidth  = 1000
   Print Form1.ScaleHeight  = 1500
```

```
Print "Form1.Width = " Form1.Width, "Form1.Height = " Form1.Height;
    Print "Form1.ScaleWidth = " Form1.ScaleWidth
    Print "Form1.ScaleHeight = " Form1.ScaleHeight
End Sub
```

در این رویه رویداد ابتدا نوع مقیاس اندازه‌گیری twip در نظر گرفته شده است سپس عرض و ارتفاع فرم مقداردهی می‌شود. اولین دستور Print ابتدا عرض و ارتفاع فرم یعنی ۴۲۰۰و ۳۰۰۰ و دو فرمان بعدی نیز مقادیر ۴۰۸۰ و ۲۵۹۵ را برای خصوصیات ScaleWidth و ScaleHeight نمایش می‌دهد. در واقع این دو خصوصیت ابعاد فرم را بدون در نظر گرفتن حاشیه و نوار عنوان فرم نمایش می‌دهند. اما در خطوط بعدی مقدار دو خصوصیت ScaleWidth و ScaleHeight روی ۱۰۰۰و ۱۵۰۰تنظیم می‌شود در این حالت خصوصیت ScaleMode نیز به‌طور خودکار روی صفر تنظیم می‌شود. پس از مقداردهی دو خصوصیت ScaleWidth و ScaleHeight، سه دستور Print دیگر اجرا می‌شوند اولین Print همان ابعاد قبلی یعنی ۴۲۰۰و ۳۰۰۰ را برای ابعاد فرم نمایش می‌دهند اما دو دستور Print بعد مقادیر جدید دو خصوصیت ScaleWidth و ScaleHeight یعنی ۱۰۰۰ و ۱۵۰۰ را نمایش خواهند داد.

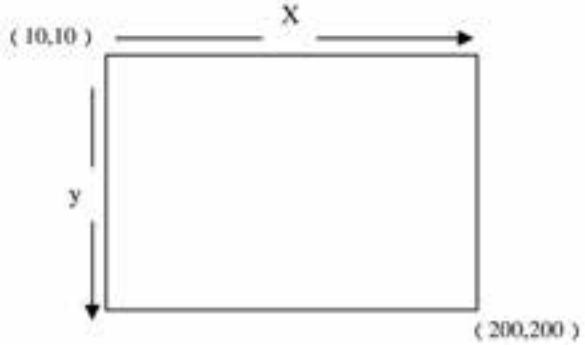
روش دیگری که در رابطه با تغییر سیستم مختصات می‌توان به‌کار برد استفاده از متد Scale است. در واقع متد Scale راه‌حل مناسب و آسان‌تری برای تنظیم سیستم مختصات می‌باشد. شکل کلی متد Scale به‌صورت زیر است:

```
object. Scale (x1,y1) - ( x2,y2)
```

مقادیر عددی x۱,y۱ مختصات گوشه بالایی و سمت چپ شئ و مقادیر عدد x۲,y۲ مختصات گوشه پایینی و سمت راست شئ را مشخص می‌کنند.

object در واقع نام شئ است که می‌خواهید سیستم مختصات آن را تعیین کنید و در صورت عدم استفاده از این قسمت، فرمی که فوکوس دارد به‌عنوان شئ مربوطه در نظر گرفته می‌شود. به‌عنوان مثال دستور زیر سیستم مختصات را درفرم به‌صورت‌شکل ۸-۱۲ درمی‌آورد.

```
Form1. Scale (10,10)-(200,200)
```



شکل ۸-۱۲

در واقع فرمان فوق چهار خصوصیت Scale را به این صورت تنظیم می‌کند:

```
ScaleWidth =190
ScaleHeight = 190
ScaleTop = 10
ScaleLeft = 10
```

۱۲–۳ خصوصیات و متدهای گرافیکی

در این‌جا لازم است تا چگونگی انجام انواع ترسیمات گرافیکی را بیاموزید. تاکنون این کار را با استفاده از کنترل‌های گرافیکی انجام می‌دادید اما کنترل‌های معرفی شده همواره نیازهای گرافیکی را برطرف نمی‌کنند و در پاره‌ای از مواقع نیز کار را با مشکلات متعدد روبه‌رو می‌سازند. استفاده از متدها و تنظیم خصوصیات گرافیکی نیاز گرافیکی شما را در پروژه‌های برنامه‌نویسی برآورده می‌کند.

۱۲–۳–۱ متد PSet

به‌وسیله این متد می‌توانید نقاط مورد نظر خود را در مکان‌های مناسب قرار دهید شکل کلی این متد به‌صورت زیر است:

```
object. PSet    Step  (x,y) , color
```

منظور از object، شیئی است که نقطه روی آن رسم می‌شود. اگر از ذکر آن خودداری کنید فرمی‌که فوکوس را در اختیار دارد مکان رسم نقطه است. کلمه کلیدی Step نیز اختیاری است و در صورت استفاده از آن هنگام رسم نقطه، مکان ترسیم با توجه به موقعیت جاری در شیئی که در آن ترسیم انجام شده، انتخاب می‌شود.

x , y مقادیر عددی از نوع Single هستند که مختصات محل ترسیم نقطه را مشخص می‌کنند. علاوه بر این می‌توانید رنگ نقطه مورد نظر را به‌وسیله بخش color تعیین کنید در صورت عدم استفاده از این قسمت، رنگی که در خصوصیت ForeColor شیئی که نقطه در آن رسم می‌شود (به عنوان مثال فرم) در نظر گرفته خواهد شد. جدول مربوط به مقادیر رنگ‌ها در جدول ۲–۱۲ آورده شده است. می‌توانید از ثابت‌های رشته‌ای یا از ثابت‌های عددی در مبنای ۱۶ استفاده کنید.

جدول ۲-۱۲ مقادیر رنگ در ویژوال بیسیک

توضیح	ثابت عددی (مبنای ۱۶)	ثابت رشته‌ای
سیاه	&H0	vbBlack
قرمز	&HFF	vbRed
سبز	&HFF00	vbGreen
زرد	&HFFFF	vbYellow
آبی	&HFF0000	vbBlue
بنفش	&HFF00FF	vbMagenta
فیروزه‌ای	&HFFFF00	vbCyan
سفید	&8HFFFFFF	vbWhite

به‌عنوان مثال این دستورات یک نقطه به رنگ آبی روی فرم نمایش می‌دهد:

```
Form1.ForeColor = vbBlue

PSet ( 1000, 200 )
```

اکنون دستور زیر را در نظر بگیرید:

```
PSet ( 500 , 700 ) , vbCyan
```

این دستور نقطه‌ای را با رنگ فیروزه‌ای در مختصات ۵۰۰ و ۷۰۰ رسم می‌کند حال اگر بلافاصله دستور زیر اجرا شود:

```
PSet Step ( 500 , 700 ) , vbGreen
```

مختصات نقطه مربوطه با توجه به مختصات نقطه رسم شده قبلی محاسبه می‌شود بنابراین دو نقطه روی یکدیگر قرار نخواهند گرفت.

مثال ۱: می‌خواهیم پروژه‌ای طراحی‌کنیم که ده‌نقطه به‌صورت‌تصادفی روی یک فرم ترسیم کند.

به این منظور عملیات زیر را به ترتیب انجام دهید:

۱ – برنامه ویژوال‌بیسیک را اجرا کرده و یک پروژه از نوع Standard EXE ایجاد کنید.

۲ – نام فرم frmgraphic و عنوان آن‌را روی Geraphic تنظیم کنید سپس یک دکمه فرمان با نام cmdpset و عنوان &Pset مطابق شکل ۹-۱۲ روی فرم قرار دهید.

شکل ۹-۱۲

۳ – رویداد click دکمه PSet را به صورت زیر تنظیم کنید:

```
Private Sub cmdpset_Click( )
   Dim i As Integer
   Randomize
   For i = 1 To 10
         PSet (Int(Rnd * 1000 ), Int(Rnd * 2000 ))
   Next i
End Sub
```

۴ – پروژه و فرم را با نام Point ذخیره کنید سپس برنامه را اجرا و روی دکمه PSet کلیک کنید تا ۱۰ نقطه به صورت تصادفی روی فرم ترسیم شوند (شکل ۹–۱۲).

۵ – از برنامه خارج شوید و به پنجره ویژوال‌بیسیک بازگردید.

تمرین:

پروژه Point را به‌گونه‌ای تغییر دهید که نقاط با رنگ‌های مختلف به طور پیوسته روی فرم نمایش داده شوند.

۱۲–۳–۲ متد Line

به‌وسیله متد Line می‌توانید انواع خطوط و مستطیل‌های توپر و توخالی را رسم کنید. شکل کلی این متد به صورت زیر است:

```
object. Line  Step  (x1,y1) - ( x2,y2 ) , color, B F
```

عملکرد گزینه‌های object و Step مانند متد PSet است. مقادیر y۱ و x۱ مختصات نقطه ابتدای خط و مقادیر y1 و x1 مختصات نقطه انتهایی را در خط تعیین می‌کنند. به‌وسیله بخش Color نیز می‌توانید رنگ خط را مشخص کنید.

استفاده از حرف B یک مستطیل خالی و استفاده از حرف F به همراه حرف B (BF) یک مستطیل توپر ایجاد می‌کند، البته استفاده از این دو کاراکتر اختیاری است. به عنوان مثال به دستورات زیر توجه کنید:

```
Line (500,800 ) – (1500 , 1500 )
Line (1000 , 2500) – ( 200,3500 ),vbBlue
```

```
Line Step (100 ,250) – ( 1400,3500) ), vbGreen

Line (410,870) – ( 1500,1800), vbBlue, B

Line (410,2500) – ( 2000,3700), vbRed, BF
```

در این مجموعه دستورات، دستور اول یک خط با رنگی که در خصوصیت ForeColor فرم تعیین شده است ترسیم می‌کند، دستور دوم یک خط با رنگ آبی و دستور سوم نیز یک خط با رنگ سبز ترسیم می‌کنند اما در دستور سوم به دلیل استفاده از Step نقطه شروع از انتهای نقطه‌ای که خط دوم تعیین می‌کند استفاده می‌شود. دستور چهارم و پنجم نیز به ترتیب یک مستطیل توخالی با رنگ آبی و یک مستطیل توپر با رنگ قرمز رسم می‌کنند. در شکل ۱۰-۱۲ و ۱۱-۱۲ نتیجه اجرای این دستورات را مشاهده می‌کنید.

شکل ۱۰-۱۲

شکل ۱۱-۱۲

۱۲-۳-۳ متد Circle

به‌وسیله این متد می‌توانید انواع دایره، بیضی و کمان را رسم کنید. شکل کلی این متد به‌صورت زیر است: object. Circle step (x,y) , radius , color , start , end , aspect

بخش object و step و color مانند توضیحات ارایه شده در متد PSet است. مقادیر عددی x و y از نوع Single بوده و مختصات مرکز دایره یا بیضی را با توجه به مقدار ScaleMode تعیین می‌کند.

مقدار عددی radius نیز از نوع Single است و مقدار شعاع دایره را براساس مقدار ScaleMode معین می‌کند. مقادیر عددی start و end از نوع Single و اختیاری بوده و موقعیت شروع و خاتمه کمان را برای ترسیم معین می‌کند. مقدار مجاز برای این دو مقدار از $2\pi -$ رادیان تا 2π رادیان است. در صورت عدم استفاده از این دو مقدار، کمان ترسیمی از صفر تا 2π رادیان در نظر گرفته می‌شود.

نکته • جهت ترسیم کمان، خلاف جهت حرکت عقربه‌های ساعت است. مقدار عددی aspect از نوع Single است و نسبت دو قطر عمودی و افقی را در بیضی معین می‌کند اگر این مقدار ۱ باشد دایره و در غیر این صورت برای مقادیر بزرگ‌تر از ۱، بیضی‌های عمودی و برای مقادیر کوچک‌تر از ۱، بیضی‌های افقی ایجاد می‌شود.

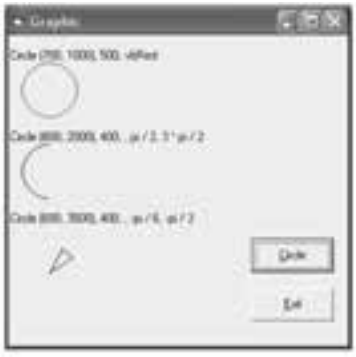
• منظور از عدد π, ثابت ۳/۱۴۱۵۹۲۶۵۳۵۸۹۷۹ است.

• برای رسم قطاع‌های بیضی یا دایره همراه با خطوط شعاع آن‌ها از مقادیر منفی استفاده کنید.

به عنوان مثال به دستورات زیر توجه کنید:

```
Cricle (700 , 1000 ),500, vbRed

Cricle (600,2000), 400, Pi/2,3* Pi/2

Cricle (600,3500), 400, -Pi/6, -Pi/3

Cricle (1700,1200), 500, vbRed,,,2

Cricle (1700,2500), 500, vbRed,,,0.2

Cricle (1000,4000), 1500,, -0..001, Pi/2,0.4
```

نتیجه اجرای این دستورات را در شکل‌های ۱۲-۱۲ و۱۳-۱۲ مشاهده می‌کنید.

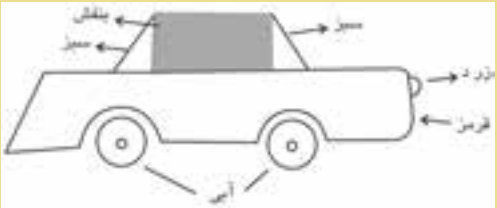


شکل ۱۲-۱۲

شکل ۱۳-۱۲

تمرین:

پروژه‌ای طراحی کنید که شکل ۱۴-۱۲ را روی یک فرم ترسیم کند.



شکل ۱۴-۱۲

۱۲-۳-۴ متد Point

این متد با دریافت مختصات یک نقطه، شماره رنگ آن را به‌صورت یک عدد صحیح از نوع Long بازمی‌گرداند. شکل کلی این متد به‌صورت زیر است:

```
object. Point (x,y)
```

به‌عنوان مثال فرض کنید که با استفاده از متد Line، یک مستطیل توپر با رنگ قرمز رسم شده است:

```
Line (500,500)-(2000,2500) , vbRed , BF

Print . Point (700,800)
```

در دستور دوم مختصات نقطه‌ای را که در مستطیل قرمز قرار دارد، به متد Point می‌دهد و در نتیجه مقدار ۲۵۵ که بیانگر رنگ قرمز است توسط متد Point بازگشت می‌یابد و نمایش داده می‌شود.

۱۲-۳-۵ خصوصیات CurrentX و CurrentY

به‌وسیله این دو خصوصیت می‌توانید موقعیت جاری مکان نما در صفحه ترسیمات را تغییر دهید. خصوصیت CurrentX مختصات مکان‌نما را در جهت محور X و خصوصیت CurrentY مختصات مکان‌نما را در جهت محور Y تعیین می‌کنند. شکل کلی نحوه استفاده از این دو خصوصیت به‌صورت زیر است:

```
object. CurrentX  = x

object. CurrentY  = y
```

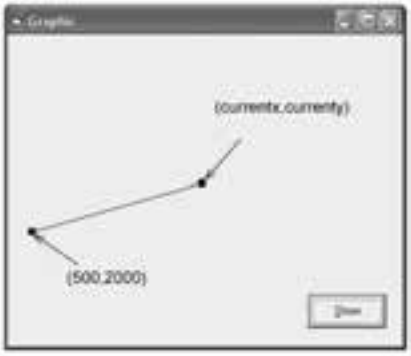
مقادیر x و y مقادیر عددی هستند که موقعیت مکان‌نما را در سطح شئ که ترسیمات گرافیکی روی آن انجام می‌گیرد مشخص می‌کنند. اگر مقادیر x و y استفاده نشود می‌توان مختصات فعلی مکان‌نما را به‌دست آورد. به دستورات زیر توجه کنید:

```
Form1.CurrentX  =  2000

Form1.CurrentY  = 1000

PSet  (CurrentX,  CurrentY)
```

در این دستورات ابتدا مختصات مکان‌نما به نقطه (۱۰۰۰ و ۲۰۰۰) تغییر پیدا می‌کند سپس به‌وسیله مقدار این دو خصوصیت نقطه‌ای در همین مختصات رسم می‌شود. به عنوان مثالی دیگر به رویه زیر توجه کنید:

```
Private Sub cmdshow_Click( )
   Form1.CurrentX = ScaleWidth / 2
   Form1.CurrentY = ScaleHeight / 2
   Line (CurrentX, CurrentY)-(500, 2000)
End Sub
```

با استفاده از خصوصیات ScaleWidth و ScaleHeight ابعاد فرم به‌دست آمده و با تقسیم آن‌ها بر عدد ۲ مختصات نقطه میانی در فرم محاسبه شده است و فرمان Line، خطی را از این نقطه تا نقطه (۵۰۰ و ۲۰۰۰) رسم می‌کند (شکل ۱۵-۱۲).



شکل ۱۵-۱۲

تمرین:
پروژه‌ای طراحی کنید که مختصات مرکز یک دایره و شعاع و رنگ آن‌را دریافت کرده و دایره موردنظر را رسم کند.

۱۲-۳-۶ متد Cls

شکل کلی این متد به صورت زیر است:

```
object. Cls
```

این متد صفحه را کاملاً پاک کرده و مکان نما را به مختصات (۰ و ۰) انتقال می‌دهد. قبلاً با نحوه استفاده از این متد آشنا شده‌اید. به‌عنوان مثال در رویه زیر ابتدا یک مستطیل توپر با رنگ بنفش و یک دایره با رنگ قرمز رسم می‌شود و بعد از رسم آن‌ها یک کادر پیغام (MessageBox) با جمله Choose ok to clear this background نمایش داده می‌شود که در صورت فشرده شدن دکمه

فرمان OK در کادر پیغام، سطح فرم پاک می‌شود و مقدار صفر برای دو خصوصیت CurrentX و CurrentY در کادر پیغام دیگری به نمایش درمی‌آید که نشان‌دهنده عملکرد متد Cls در تغییر موقعیت جاری مکان‌نماست.

```
Private Sub cmdcls_Click( )
   Dim pi, msg
   pi = 3.14159265358979
   Line (200, 150)-(850 ,600) , vbMagenta, BF
   Circle (1600 ,1800 ), 400 , vbRed
   msg = "Choose OK to clear this background"
   MsgBox msg
   Cls
MsgBox "CurrentX=" + Str(CurrentX) + "CurrentY=" + Str(CurrentY)
End Sub
```

مثال ۲: پروژه‌ای طراحی کنید که به وسیله آن کاربر بتواند اشکال هندسی مانند نقطه, خط و انواع مستطیل را رسم کند. به این منظور مراحل بعد را انجام دهید:

۱ – یک پروژه از نوع Standard EXE ایجاد کنید که شامل یک فرم با عرض و ارتفاع ۸۰۰۰ و ۶۰۰۰ باشد. سپس نام و عنوان آن را به‌ترتیب روی Frmdrawing و Drawing تنظیم کنید و خصوصیت Appearance آن را روی Flat- ◦ تنظیم کنید.(شکل ۱۶–۱۲).

۲ – در این مرحله بخش تعاریف فرم و رویدادهای ماوس را به صورت زیر تنظیم کنید:

```
Option Explicit
Dim intx As Integer, inty As Integer
Private Sub Form_MouseDown(Button As Integer,Shift As_
            Integer , X As Single, Y As Single)
If Button = vbLeftButton Then
```

```
        Pset (X,Y)

        intx = X

        inty = Y

      End If

End Sub

Private Sub Form_MouseUp(Button As Integer,Shift As Integer , X As Single, Y As Single)

   If Button = vbLeftButton Then

      Select Case Shift

        Case vbCtrlMask: Line (intx, inty)-(X,Y)

        Case vbShiftMask: Line (intx, inty)-(X,Y),  , B

        Case vbAltMask: Line (intx, inty)-(X, Y),  , BF

      End Select

   End If

End Sub

Private Sub Form_MouseMove(Button As Integer,Shift As_

                Integer , X As Single, Y As Single)

   If Button = vbRightButton Then

      PSet (X,Y)

   End If

End Sub
```

شکل ۱۶-۱۲

در رویداد MouseDown با استفاده از دستور If مقدار آرگومان Button بررسی می‌شود. اگر این مقدار برابر با vbLeftButton باشد به این معنی است که کلید سمت چپ ماوس فشرده شده است و با استفاده از دستور PSet یک نقطه در موقعیتی که اشاره‌گر ماوس قرار دارد، ترسیم می‌شود و همین مقادیر یعنی X و Y در دو متغیر عمومی ذخیره می‌شوند تا برای رویداد MouseUp مورد استفاده قرار گیرند. متد PSet می‌تواند یک نقطه در مختصات x و y با رنگ color ایجاد کند، color می‌تواند یکی از ثابت‌های رنگ ویژوال‌بیسیک باشد.

شکل کلی استفاده از این دستور به صورت PSet (x,y),color می‌باشد. استفاده از آرگومان color اختیاری است.

از رویداد MouseUp برای ترسیم انواع خطوط و مستطیل‌های توپر و توخالی استفاده شده است. در این رویداد ابتدا رها شدن کلید سمت چپ ماوس به وسیله دستور If کنترل می‌شود، سپس با استفاده از یک دستور Select Case فشرده شدن کلیدهای Ctrl، Shift و Alt در زمان رها شدن کلید چپ ماوس بررسی می‌شود و اگر در زمان رها شدن کلید سمت چپ ماوس، کلید Ctrl نگه داشته شود، متد Line (intx,inty)-(X,Y) در اولین Case اجرا شده و یک خط بین نقطه با مختصات (intx,inty) یعنی محل اشاره‌گر در زمان فشرده شدن کلید چپ ماوس و نقطه با مختصات (X,Y) یعنی محل اشاره‌گر در زمان رها شدن کلید چپ ماوس رسم می‌شود. اگر در زمان رها شدن کلید چپ ماوس، کلید Shift نگه داشته شود، متد Line (intx,inty)-(X ,Y, B) در Case دوم اجرا شده و یک مستطیل بین نقاط (intx,inty) و (X,Y) رسم می‌کند. پارامتر B در متد Line سبب رسم یک مستطیل

خواهد شد. اما اگر در زمان رها شدن کلید چپ ماوس، کلید Alt نگه داشته شود، متد Line (intx,inty)-(X,Y),,BF در Case آخر اجرا شـــده و یک مســـتطیل توپر بین نقاط مربوطه رسم خواهد شد.

متد Line می‌تواند یک خط را با دریافت موقعیت نقاط ابتدا و انتهای آن ترســـیم کند. شـــکل کلی این متد به صورت Line (x1,y1)-(x2,y2),color است کـــه x2,x1 فـاصله نقاط ابتدا و انتهای خط از ســـمت چپ فرم و y۱ و y۲ فاصله نقاط ابتدا و انتهای خط از بالای فرم می‌باشـــد, آرگومان color رنگ خط ترســـیمی را مشخص می‌کند. استفاده از این آرگومان اختیاری است.

برای رسم یک مستطیل توخالی از کاراکتر B در متد Line به صورت Line(x1,y1)-(x2,y2),color,B اســـــتفاده کـنید و برای رسـم یـک مستطیل توپر از عـبارت BF در متد Line به صـورت Line (x1,y1)-(x2,y2),color,BF استفاده کنید.

آخریـــن رویداد در ماژول فرم، رویداد MouseMove فرم اســـت کـــه در این رویداد با بررســـی آرگومان Button فشرده شدن دکمه راســـت ماوس بررسی می‌شود. اگر در زمان حرکت اشـــاره‌گر ماوس کلید راست ماوس نگه داشته شـــود (Right Drag) متد PSet اجرا شده و یک نقطه در محل اشاره‌گر ترسیم خواهد کرد.

۳ – پروژه را با نام drawing ذخیره کرده و سپس آن را اجرا کنید.

۴ – روی فرم، عمل کلیک انجام دهید. مشاهده می‌کنید که نقاط کوچکی با رنگ سیاه روی فرم ایجاد خواهد شد.

۵ – روی فـــرم عمل درگ انجام دهید و هم‌زمان کلید Ctrl را پایین نگه‌دارید، ســـپس کلید چپ ماوس را رها کنید. مشـــاهده می‌کنید که یک خط با رنگ سیاه ترسیم می‌شود. همین عمل را با کلیدهای Shift و Alt انجام دهید و نتیجه را بررسی کنید.

۶ – در ایـــن مرحلـــه با کلید راســـت ماوس عمل درگ را انجام دهیـــد، می‌بینید که با حرکت ماوس نقاط ترسیم می‌شوند.

۷ – از برنامه خارج شده و به پنجره ویژوال‌بیسیک بازگردید.

۷-۳-۱۲ متد Print

تاکنون بارها از این متد استفاده کرده‌اید. این متد می‌تواند هرگونه اطلاعات اعم از متن, مقادیر عددی، مقادیر مربوط به خصوصیات و نظایر آن‌ها را روی فرم نمایش دهد. شکل کلی این متد به صورتی است که در ادامه می‌آید:

```
object. Print     outputlist
```

object نام شیئی است که اطلاعات روی آن نمایش داده می‌شوند. outputlist اطلاعاتی است که توسط متد Print روی شیء object نمایش داده می‌شود. استفاده از outputlist و object اختیاری است و در صورتی که outputlist استفاده نشود یک خط خالی نمایش داده خواهد شد.

نکته:

- استفاده از کاراکتر " ; " در متد Print سبب می‌شود تا اطلاعات به صورت چسبیده به هم و بدون فاصله از هم نمایش داده شوند.
- استفاده از کاراکتر " , " در متد Print سبب می‌شود تا اطلاعات با فاصله ۱۴ ستون از یکدیگر نمایش داده شوند.

به‌عنوان مثال, رویداد زیر عبارت Visual Basic 6 را به شکل‌های مختلف نمایش می‌دهند:

```
Private Sub cmdprint_Click( )
   Dim str1 As String
   Dim str2 As String
   Dim str3 As String
   str1 = "Visual"
   str2 = "Basic"
   str3 = "6"
   Print str1; str2; str3
```

```
    Print str1, str2, str3

    Print str1;

    Print str2,

    Print str3
End Sub
```

نتیجه اجرای این دستورات به‌صورت زیر خواهد بود:

```
Visual Basic 6

Visual        Basic    6

VisualBasic      6
```

شکل ۱۷-۱۲

برای نمایش فضاهای خالی توسط متد Print، می‌توانید از تابع Spcace استفاده کنید. مثلاً دستور زیر قبل از نمایش کلمه BASIC، ۵ فضای خالی ایجاد می‌کند.

```
Print Spcace (5) ; " BASIC "
```

در ضمن می‌توانید به‌وسیله تابع Tab اطلاعات خود را در ستون‌های مورد نظر نمایش دهید. شکل کلی تابع Tab به این صورت است:

```
Tab (n)
```

که در آن n یک مقدار عددی است که شماره ستون مورد نظر را معین می‌کند. مثلاً فرمان زیر، کلمه VISUAL را از ستون دوم به بعد و کلمه BASIC را از ستون دهم به بعد نمایش می‌دهد.

```
Print    Tab (2) ; " VISUAL " ; Tab (10) ; " BASIC"
```

اما اگر این فرمان به‌صورت زیر استفاده شود:

```
Print    Tab (2) ; " VISUAL " ; Tab (5) ; " BASIC "
```

کلمه VISUAL از ستون دوم خط جاری نمایش داده می‌شود اما چون پس از نمایش کلمه VISUAL، مکان نما در ستون هشتم قرار می‌گیرد و Tab دوم به ستون پنجم اشاره می‌کند بنابراین کلمه BASIC در ستون پنجم خط بعد نمایش داده خواهد شد.

نکته در صورتی که مقدار n در تابع Tab عددی منفی باشد، ستون شماره ۱ در نظر گرفته خواهد شد.

۸-۳-۱۲ متدهای TextWidth و TextHeight

متد TextWidth عرض متن مورد نظر و متد TextHeight ارتفاع متن مورد نظر برای نمایش را معین می‌کنند. شکل کلی این دو متد به‌صورت زیر است:

```
object. TextWidth ( string )

object. TextHeight ( string )
```

string یک عبارت رشته‌ای است. هر دو متد یک مقدار عددی از نوع Single را بازمی‌گردانند مثلاً برای نمایش عبارت VISUAL BASIC در وسط صفحه می‌توانید به این صورت عمل کنید:

```
Private Sub cmdtext_Click( )
   CurrentX = (ScaleWidth - TextWidth ("VISUAL BASIC")) / 2
   CurrentY = (ScaleHeight - TextHeight ("VISUAL BASIC")) / 2
   Print "VISUAL BASIC"
End Sub
```

نتیجه اجرای این رویه را در شکل ۱۸-۱۲ مشاهده کنید:

در این رویه به‌وسیله دو متد TextWidth و TextHeight و محاسبه عرض و ارتفاع متن و با کم کردن این مقادیر از عرض و ارتفاع فرم که به‌وسیله خصوصیات ScaleWidth و ScaleHeight به‌دست می‌آیند و تقسیم مقدار به‌دست آمده بر عدد ۲، محل نمایش متن را به‌گونه‌ای که در وسط صفحه قرار گیرد محاسبه شده‌اند. مزیت این کار این است که حتی اگر ابعاد فرم تغییر یابد و رویه مجدداً اجرا شود متن مورد نظر با توجه به ابعاد جدید باز

شده و در وسط صفحه قرار می‌گیرد.

شکل ۱۸-۱۲

۹-۳-۱۲ خصوصیت AutoRedraw

شاید تاکنون در هنگام استفاده از دستورات گرافیکی به این نکته دقت کرده‌اید که در بعضی از مواقع ترسیمات انجام شده در روی فرم از بین می‌رود یا در اثر قرار گرفتن پنجره‌های دیگر روی پنجره‌ای که ترسیمات در آن انجام شده است ترسیمات شما مخدوش می‌شود. به‌عنوان مثال یک فرم با یک دکمه فرمان به گونه‌ای طراحی کنید که با فشردن دکمه فرمان یک مستطیل توپر در روی فرم رسم شود سپس برنامه را اجرا کرده و روی دکمه فرمان کلیک کنید و بعد پنجره برنامه را به حداقل اندازه برسانید (به‌وسیله دکمه‌کنترلی Minimize) و مجدداً روی آیکن پنجره در نوار وظیفه کلیک کنید تا پنجره برنامه به حالت قبل بازگردد. در این صورت اثری از شکل رسم شده مشاهده نخواهد شد.

فرم‌ها و بعضی از کنترل‌ها مانند PictureBox خصوصیتی به نام AutoRedraw دارند که در حالت پیش‌فرض مقدار این خصوصیت False است اگر این خصوصیت روی مقدار True تنظیم شود در هنگام تغییر مکان یا تغییر اندازه فرم و نظایر آن، ترسیمات موجود از بین نمی‌روند و با نمایش مجدد فرم یا کنترل نمایش داده می‌شوند.

در مثالی که در این قسمت انجام دادید خصوصیت AutoRedraw را برای فرم برنامه روی مقدار True تنظیم کنید و برنامه را مجدداً اجرا کنید و روی دکمه فرمان کلیک کنید. فرم را به حالت Minimize درآورده و دوباره آن را روی دسک‌تاپ نمایش دهید. مشاهده می‌کنید در این حالت مشکل قبلی از بین رفته است.

شکل کلی استفاده از این خصوصیت به این صورت است:

```
object. AutoRedraw  = Boolean
```

object یک مقدار اختیاری است که می‌تواند نام فرم یا کنترل باشد و در صورت عدم استفاده از آن، نام فرمی‌که فوکوس دارد استفاده می‌شود.

مقدار Boolean یک مقدار منطقی است که می‌تواند True یا False باشد. در صورت عدم استفاده از مقدار Boolean، مقدار خصوصیت بازگشت داده می‌شود. مقدار پیش‌فرض این خصوصیت False است.

تمرین:

از خصوصیت AutoRedraw در پروژه‌هایی که تاکنون طراحی کرده‌اید استفاده کرده و نتیجه را با حالت قبل از استفاده این خصوصیت بررسی کنید.

۱۰-۳-۱۲ خصوصیت DrawMode

به‌وسیله این خصوصیت می‌توان شکل ظاهری ترسیماتی (از نظر رنگ) را که به‌وسیله متدهای گرافیکی نظیر Circle، Line، PSet و غیره انجام می‌شوند تعیین کرد. شکل کلی نحوه استفاده از این خصوصیت به‌صورت زیر است:

```
object. DrawMode  = value
```

مقدار value نیز اختیاری بوده و شکل ظاهری ترسیمات را مشخص می‌کند. درصورت عدم استفاده از این مقدار، مقدار خصوصیت بازگشت داده می‌شود. مقادیری که بخش value می‌تواند کسب کند در جدول ۳-۱۲ آورده شده‌اند.

توجه داشته باشید که در جدول ۳-۱۲منظور از رنگ pen، رنگی است که متد گرافیکی برای رسم شکل گرافیکی از آن استفاده می‌کند. در واقع این خصوصیت به ویژوال‌بیسیک می‌گوید که چگونه رنگ یک pixel نمایشی روی صفحه را تعیین کند به‌عبارت دیگر ویژوال‌بیسیک بر اساس رنگ فعلی pixel و رنگ نقطه‌ای که در این pixel به‌وسیله متد گرافیکی ایجاد می‌شود, ترکیب رنگی مناسب را با توجه به مقدار خصوصیت ScaleMode تعیین می‌کند. به‌عنوان مثال دستورات زیر نحوه استفاده از این خصوصیت را نشان می‌دهند:

```
Form1.DrawMode  =  vbInvert

Form1.BackColor  =  vbRed

Line (500 , 500  ) - (2000 , 2500) vbGreen,  BF
```

در حالت عادی این دستورات یک مستطیل توپر به‌وسیله متد Line با رنگ سبز رسم می‌کنند اما چون مقدار خصوصیت DrawMode روی vbInvert تنظیم شده، از رنگ معکوس رنگ قرمز (یعنی رنگ فعلی نقاطی که مستطیل به‌وسیله آن‌ها نمایش داده می‌شود) استفاده می‌شود و مستطیل با این رنگ دیده خواهد شد. بنابراین اگر رنگ دیگری نیز در دستور Line استفاده شود تغییری در رنگ مستطیل ایجاد نمی‌شود.

جدول ۳-۱۲ مقادیر مربوط به خصوصیت DrawMode

توضیح	ثابت عددی	ثابت رشته‌ای
رنگ سیاه	۱	vbBlackness
عکس حالت 15-vbMergePen	۲	vbNotMergePen
ترکیب رنگ بر اساس رنگ زمینه و رنگ معکوس pen	۳	vbMaskNotPen
عکس حالت 13-CopyPen	۴	vbNotCopyPen
ترکیب رنگ‌های عمومی با رنگ pen و رنگ معکوس، رنگی که نمایش داده شده است.	۵	vbMaskPenNot
رنگ معکوس، رنگی که نمایش داده شده است.	۶	vbInvert
ترکیب رنگ‌ها بر اساس رنگ pen یا رنگی که نمایش داده شده است.	۷	vbXorPen
عکس حالت 9-MaskPen	۸	vbNotMaskPen
ترکیب رنگ‌های عمومی با رنگ pen و رنگی که نمایش داده شد.	۹	vbMaskPen
عکس حالت 7-vbXorPen	۱۰	vbNotXorPen
ترسیمی انجام نمی‌شود.	۱۱	vbNop
ترکیب رنگ معکوس، رنگ pen و رنگی‌که نمایش داده شده است.	۱۲	vbMergeNotePen
به‌طور پیش‌فرض رنگ pen و اگر رنگ pen تعیین نشود رنگ ForeColor استفاده می‌شود.	۱۳	vbCopyPen
ترکیب رنگ pen و رنگ معکوس، رنگی‌که نمایش داده شده است.	۱۴	vbMergePenNot
ترکیب رنگ pen و رنگی که نمایش داده شده است.	۱۵	vbMergePen
رنگ سفید	۱۶	vbWhiteness

۳۴۸

نکته مقدار پیش‌فرض خصوصیت DrawMode، 13-vbCopyPen است که سبب می‌شود از رنگی که در متد گرافیکی تعیین می‌شود استفاده شود و در صورتی که رنگ توسط متد تعیین نشود از رنگی که در خصوصیت ForeColor تعیین شده استفاده شود.

۱۲-۳-۱۱ خصوصیت DrawStyle

به‌وسیله این خصوصیت می‌توان نوع و حالت خطوط را در ترسیمات گرافیکی تعیین کرد. شکل کلی نحوه استفاده از این خصوصیت به‌صورت زیر است:

object. DrawStyle = value

مقدار value اختیاری بوده و نوع و حالت ترسیمات را مشخص می‌کند. در صورت عدم استفاده از این مقدار، مقدار خصوصیت بازگشت داده می‌شود. مقادیری که value می‌تواند کسب کند در جدول ۴-۱۲ ارایه شده است.

جدول ۴-۱۲ مقادیر مربوط به خصوصیت DrawStyle

حالت نمایشی	ثابت عددی	ثابت رشته‌ای
————	۰	vbsolid
— —	۱	vbDash
------	۲	vbDot
— . — .	۳	vbDashDot
— .. —	۴	vbDashDotDot
خط نامریی	۵	vbInvisible
لبه بیرونی حاشیه بر لبه بیرونی شکل منطبق است.	۶	vbInsideSolid

رویداد زیر، حالت‌های مختلف ترسیم را برای مقادیر صفر تا ۴ مورد استفاده قرار می‌دهد.

```
Private Sub cmdshow_Click( )
   DrawStyle = vbSolid
   Line (500, 200)-(5000, 200), vbBlue
   DrawStyle = vbDash
   Circle (900,1000), 400, vbRed
```

```
    DrawStyle = vbDot
    Line (2000, 500)-(5000, 1500), vbBlue, B
    DrawStyle = vbDashDot
    Circle (900, 2500),400, vbRed
    DrawStyle = vbDashDotDot
    Line (2000 ,2000)-( 5000-3000), vbBlue, B
End Sub
```

شکل ۱۹–۱۲

۱۲–۳–۱۲ خصوصیت DrawWidth

به‌وسیله این خصوصیت می‌توانید ضخامت خطوط را برای ترسیمات گرافیکی معین کنید. شکل کلی نحوه استفاده از این خصوصیت به‌صورت زیر است:

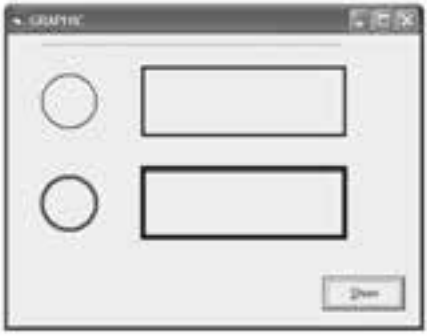
object. DrawWidth = size

مقدارsize اختیاری بوده و یک عبارت عددی است که اندازه قلم را برای ترسیمات معین می‌کند و در صورت عدم استفاده از این مقدار، مقدار خصوصیت بازگشت داده می‌شود. مقدار size می‌تواند مقداری بین ۱ تا ۳۲۷۶۷ باشد.

در این رویداد حالت‌های مختلف ترسیم را با اندازه‌های مختلف قلم مشاهده می‌کنید (شکل ۲۰–۱۲).

```
Private Sub cmdshow_Click( )
    DrawStyle = vbDot
    DrawWidth = 1
    Line (500, 150)-(5000, 150), vbBlue
    DrawWidth = 2
```

```
    Circle (900, 1000), 400, vbRed
    DrawWidth = 3
    Line (2000, 500)-(5000, 1500), vbBlue, B
    DrawWidth = 4
    Circle (900, 2500), 400, vbRed
    DrawWidth = 5
    Line (2000, 2000)-(5000, 3000), vbBlue, B
End Sub
```



شکل ۲۰-۱۲

نکته در صورتی که خصوصیت DrawWidth روی عدد بزرگ‌تر از یک تنظیم شود مقادیر ۱ تا ۴ برای خصوصیت DrawStyle در زمان اجرای متدهای گرافیکی عملکردی از خود نشان نمی‌دهند و مانند مقدار vbSolid عمل می‌کنند.

۱۲-۳-۱۳ خصوصیت FillStyle

به‌وسیله این خصوصیت می‌توانید ترسیماتی نظیر دایره، بیضی و مستطیل را با حالت‌های مختلف پر کنید. شکل کلی نحوه استفاده از این خصوصیت به‌صورت زیر است:

object. FillStyle = number

مقدار number اختیاری بوده و عددی صحیح است که حالت موردنظر را برای پر کردن ترسیمات معین می‌کند. در صورت عدم استفاده از این بخش، مقدار خصوصیت بازگشت داده می‌شود. مقادیر مجاز برای number در جدول ۵-۱۲ ارایه شده است:

جدول ۵-۱۲ مقادیر مربوط به خصوصیت FillStyle

ثابت رشته‌ای	ثابت عددی	توضیح
vbFsSolid	۰	داخل شکل با رنگی که در خصوصیت FillColor تعیین شده پر می‌شود.
vbFStransparent	۱	داخل شکل با رنگ زمینه پر می‌شود.
vbHorizontalLine	۲	داخل شکل با خطوط افقی پر می‌شود.
vbVerticalLine	۳	داخل شکل با خطوط عمودی پر می‌شود.
vbUpwardDiagonal	۴	داخل شکل با خطوط مایل پر می‌شود (از چپ به راست).
vbDownwardDiagonal	۵	داخل شکل با خطوط مایل پر می‌شود (از راست به چپ).
vbCross	۶	داخل شکل با خطوط عمودی و افقی پر می‌شود (حالت شطرنجی).
vbDiagonalCross	۷	داخل شکل با خطوط عمودی و افقی مایل پر می‌شود(حالت شطرنجی مایل).

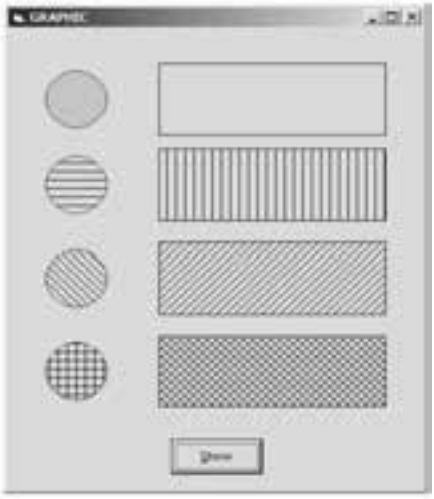
در این رویداد حالت‌های مختلف ترسیم را برای مقادیر متفاوتی از خصوصیت FillStyle نشان می‌دهد:

```
Private Sub cmdshow_Click( )
   DrawStyle = vbSolid
   DrawWidth = 1
   FillColor = vbGreen
   FillStyle = 0
   Circle (900, 1000), 400, vbRed
   FillColor = vbBlack
   FillStyle = 1
   Line (2000, 500)-(5000, 1500), vbBlue, B
   FillStyle = 2
   Circle (900,2200), 400, vbRed
   FillStyle = 3
   Line (2000, 1700)-(5000, 2700), vbBlue, B
   FillStyle = 4
   Circle (900,3500), 400, vbRed
   FillStyle = 5
```

```
    Line  (2000, 3000)-(5000, 4000),  vbBlue,  B
    FillStyle  =  6
    Circle  (900, (4800, 400,  vbRed
    FillStyle  =  7
    Line  (2000, 4300)-(5000,5300),  vbBlue,  B
End  Sub
```

در این رویداد پس از تعیین مقادیر مورد نظر برای خصوصیات DrawStyle و DrawWidth، مقدار خصوصیت FillColor روی سبز تنظیم شده است. خصوصیت FillCo-or رنگ قلم را برای ترسیماتی که به‌وسیله خصوصیت FillStyle ایجاد می‌شود معین می‌کند.

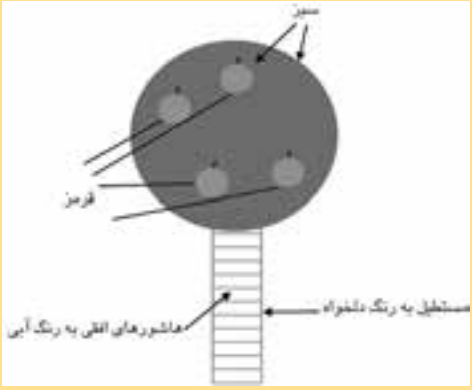
به‌عنوان مثال در دایره اول چون مقدار خصوصیت FillStyle، صفر است دایره با رنگ FillColor یعنی سبز پر می‌شود. در مورد سایر مقادیر FillStyle نیز خصوصیت FillColor رنگ خطوط عمودی، افقی، مایل و غیره که شکل را پر می‌کند تعیین می‌کند. نتیجه اجرای این رویداد را می‌توانید در شکل ۲۱-۱۲ مشاهده کنید.



شکل ۲۱-۱۲

تمرین:

پروژه‌ای طراحی کنید که این شکل را روی یک فرم نمایش دهد:



شکل ۲۲-۱۲

۱۲-۴ تابع QBcolor

این تابع با دریافت یک عدد بین صفر و ۱۵، یک عدد از نوع Long را که بیانگر رنگ معادل عدد دریافتی است بازمی‌گرداند. شکل کلی این تابع به‌صورت زیر است:

QBColor (color)

آرگومان color یک عدد از نوع صحیح است که براساس جدول ۶-۱۲ قابل استفاده خواهند بود.

جدول ۶-۱۲ مقادیر قابل استفاده برای آرگومان color

رنگ	مقدار عددی	رنگ	مقدار عددی
خاکستری	۸	سیاه	۰
آبی روشن	۹	آبی	۱
سبز روشن	۱۰	سبز	۲
فیروزه‌ای روشن	۱۱	فیروزه‌ای	۳
قرمز روشن	۱۲	قرمز	۴
بنفش روشن	۱۳	بنفش	۵
زرد روشن	۱۴	زرد	۶
سفید روشن	۱۵	سفید	۷

به‌عنوان مثال در دستورات زیر با استفاده از تابع QBColor، ۵۰۰۰ نقطه با رنگ‌های متفاوت نمایش داده خواهد شد.

```
Dim i As Integer
Dim XPos As Single, YPos  As Single
DrawWidth = 4
Randomize
For i = 1 To 5000
    XPos = Rnd * ScaleWidth
    YPos = Rnd * ScaleHeight
    PSet (XPos, YPos), QBColor(Rnd * 15)
Next i
```

۱۲-۵ تابع RGB

در ویژوال‌بیسیک تابع دیگری به نام RGB وجود دارد که می‌تواند ترکیبات رنگی را با توجه به نیاز ایجاد کند. این تابع می‌تواند با دریافت سه مقدار عددی برای سه رنگ اصلی تمام ترکیبات مورد نظر را ایجاد کند.

شکل کلی این تابع به‌صورت زیر است:

RGB (red , green , blue)

این تابع سه آرگومان اجباری دارد که می‌توانند اعداد صحیح از صفر تا ۲۵۵ را کسب کنند. آرگومان red مقدار رنگ قرمز، آرگومان green مقدار رنگ سبز و آرگومان blue مقدار رنگ آبی را معین می‌کنند. مقدار بازگشتی این تابع یک عدد از نوع Long است که بیانگر ترکیب رنگی درخواستی است. مقادیر سه آرگومان فوق برای رنگ‌های استاندارد در جدول ۷-۱۲ ارایه شده است.

جدول ۷-۱۲ مقادیر سه رنگ اصلی برای رنگ‌های استاندارد

رنگ	مقدار آرگومان red	مقدار آرگومان green	مقدار آرگومان blue
سیاه	۰	۰	۰
آبی	۰	۰	۲۵۵
سبز	۰	۲۵۵	۰
فیروزه‌ای	۰	۲۵۵	۲۵۵
قرمز	۲۵۵	۰	۰
بنفش	۲۵۵	۰	۲۵۵
زرد	۲۵۵	۲۵۵	۰
سفید	۲۵۵	۲۵۵	۲۵۵

به‌عنوان مثال دستورات زیر پانصد دایره با ابعاد و مختصات و رنگ‌های تصادفی ایجاد می‌کند.

```
Dim i As Integer
Dim XPos As Single, YPos As Single
DrawWidth = 4
```

```
Randomize
For i = 1 To 500
XPos = Rnd * ScaleWidth
   YPos = Rnd * ScaleHeight
    (Circle (XPos, YPos), Rnd * 800, RGB(Rnd * _ 255, Rnd * 255, Rnd * 255
Next i
```

۱۲-۶ شئ چاپگر (Printer Object)

تاکنون کلیه عملیاتی که انجام داده‌اید روی فرم و صفحه نمایش انجام شـده است اما گاهی اوقات لازم اسـت تا اطلاعات مورد نیاز خود را به‌وسـیله چاپگر روی کاغذ چاپ کنید. ویژوال‌بیسـیک در این زمینه نیز امکانات لازم را مهیا کرده اسـت. شـما با استفاده از شـئ چاپگر، علاوه بر انجام عملیات چاپ می‌توانید به‌وسـیله خصوصیات این شـئ عملیات چاپ را به نحو مناسـبی مدیریت کنید. در این بخش به معرفی متدهای چاپ و معرفی خصوصیات شئ چاپگر می‌پردازیم.

۱۲-۶-۱ متدهای چاپ

تاکنون متدهای مختلفی را برای نمایش اطلاعات و ترسیمات آموختید. در این متدها می‌توانید به جای پارامتر object از شـئ چاپگر اسـتفاده کنید؛ بنابراین به آسانی می‌توانید از متدهای Circle، Line، PSet، Scale، TextHeight، TextWidth و Print اسـتفاده کنید. فقط کافی است برای معـرفی شئ چاپـگر از کلـمه Printer استفاده کنید. به‌عـنوان مـثال نتایج حاصل از اجرای دستـورات زیر روی کاغذ چاپ خواهند شد.

```
Printer. Print  " IN THE NAME OF GOD. "
(Printer. Line (100,100) – (800,800
Printer. Circle (150,150), 500
```

متد EndDoc

این متد سبب می‌شود تا عملیات چاپ متوقف شده و تا زمانی که چاپگر آماده چاپ شود اطلاعات مربوط به چاپ، روی دیسک یا حافظه کامپیوتر ذخیره می‌شود. شکل کلی

نحوه استفاده از این متد به‌صورت زیر است:

Printer. EndDoc

متد KillDoc

این متد می‌تواند در زمان چاپ اطلاعات، عملیات چاپ را خاتمه دهد. شکل کلی نحوه استفاده از این متد به‌صورت زیر است:

Printer. KillDoc

متد NewPage

متد Newpage می‌تواند عملیات چاپ صفحه جاری را خاتمه داده و چاپگر، چاپ را از صفحه بعدی انجام دهد. شکل کلی نحوه استفاده از این متد در ادامه می‌آید:

Printer. NewPage

۲-۶-۱۲ خصوصیات شئ چاپگر

تنظیمات مربوط به چاپگرها نیز مانند اشیای دیگر به‌وسیله تعدادی از خصوصیت‌ها قابل دستیابی و تغییر هستند. در این بخش مهم‌ترین خصوصیات شئ چاپگر را مورد بررسی قرار می‌دهیم. بعضی از خصوصیات نیز قبلاً توضیح داده شده‌اند مانند: Dra- Style، DrawMode، CurrentY، CurrentX، FillStyle، FillColor، DrawWidth، خصوصیات مربوط به قلم‌ها (Fonts) و ...

خصوصیت ColorMode

به‌وسیله این خصوصیت می‌توان نوع چاپگر را از نظر چاپ رنگی یا سیاه سفید تعیین کرد. شکل کلی نحوه استفاده از این خصوصیت به‌صورت زیر است:

Printer. ColorMode = value

value یک ثابت عددی یا رشته‌ای است که نوع چاپ را معین می‌کند. این مقدار می‌تواند یکی از مقادیر موجود در جدول ۸-۱۲ باشد. در صورت عدم استفاده از بخش value، مقدار فعلی خصوصیت بازگشت داده خواهد شد.

جدول ۸-۱۲ مقادیر مربوط به خصوصیت ColorMode

توضیح	ثابت عددی	ثابت رشته‌ای
چاپ سیاه سفید	۱	vbPRCMMonochrome
چاپ رنگی	۲	vbPRCMColor

نکته استفاده از چاپ رنگی یا سیاه سفید به امکانات چاپگر بستگی دارد.

خصوصیت Copies

به‌وسیله این خصوصیت می‌توانید تعداد نسخه‌هایی که چاپگر چاپ می‌گیرد تعیین کنید. شکل کلی نحوه استفاده از این خصوصیت به‌صورت زیر است:

Printer. Copies = number

Number یک عبارت عددی از نوع صحیح است که تعداد نسخه‌ها را برای چاپ معین می‌کند و در صورت عدم استفاده از آن، مقدار فعلی خصوصیت بازگشت داده خواهد شد.

خصوصیت DeviceName

این خصوصیت نام دستگاه چاپگر پیش‌فرض را باز می‌گرداند. نام چاپگرها در زمان نصب آن‌ها از طریق برنامه Control Panel توسط کاربر تعیین می‌شود. مثلاً اگر یک چاپگر EPSON LQ 300 را با نام myprinter و به‌صورت پیش‌فرض نصب کرده باشید فرمان زیر نام چاپگر یعنی myprinter را نمایش می‌دهد.

Print Printer. DeviceName

خصوصیت DriverName

این خصوصیت نام راه‌انداز (driver) دستگاه چاپگر پیش‌فرض را باز می‌گرداند. به‌عنوان مثال اگر چاپگر EPSON LQ 500 را با نام myprinter نصب کرده باشید فرمان زیر نام راه‌انداز نصب شده یعنی EPSON LQ 500 را نمایش خواهد داد.

Print Printer. DriverName

خصوصیت Orientation

به‌وسیله این خصوصیت می‌توانید جهت انجام عملیات چاپ را روی صفحه کاغذ تعیین کنید. عملیات چاپ می‌تواند به‌صورت portrait یا landscape باشد. در شکل ۲۳–۱۲ تفاوت این دو حالت نمایش داده شده است.

شکل ۲۳–۱۲

شکل کلی نحوه استفاده از این خصوصیت به‌صورت زیر است:

Printer. Orientation = value

value یک ثابت عددی یا رشته‌ای است که جهت چاپ اطلاعات را روی کاغذ معین می‌کند. این مقدار می‌تواند یکی از مقادیر موجود در جدول ۹-۱۲ باشد. در صورت عدم استفاده از مقدار value، مقدار فعلی خصوصیت بازگشت داده خواهد شد.

جدول ۹-۱۲ مقادیر مربوط به خصوصیت Orientation

توضیح	ثابت عددی	ثابت رشته‌ای
چاپ به‌صورت portrait	۱	vbPRORPortrait
چاپ به‌صورت landscape	۲	vbPRORLandscape

خصوصیت Page

این خصوصیت شماره صفحه در حال چاپ را در اختیار برنامه قرار می‌دهد. شکل کلی نحوه استفاده از این خصوصیت به این صورت است:

Printer.Page

خصوصیت PaperSize

به‌وسیله این خصوصیت می‌توانید نوع و ابعاد کاغذ چاپ را تنظیم کنید. شکل کلی نحوه استفاده از این خصوصیت به‌صورت زیر است:

Printer.PaperSize = value

value یک ثابت عددی یا رشته‌ای است که ابعاد کاغذ را معین می‌کند. این مقدار می‌تواند یکی از مقادیر موجود در جدول ۱۰-۱۲ باشد. در صورت عدم استفاده از مقدار value، مقدار فعلی خصوصیت بازگشت داده خواهد شد.

جدول ۱۰-۱۲ مقادیر مربوط به اندازه کاغذ در خصوصیت PageSize

ثابت رشته‌ای	ثابت عددی	ابعاد کاغذ
vbPRPSLetter	۱	۸ ۱/۲ ×۱۱ Inch
vbPRPSLetterSmall	۲	۸ ۱/۲ ×۱۱ Inch (اندازه کوچک)
vbPRPSTabloid	۳	۱۱ × ۱۷ Inch
vbPRPSLedger	۴	۱۷ ×۱۱ Inch
vbPRPSLegal	۵	۸ ۱/۲ ×۱۴ Inch
vbPRPSStatement	۶	۵ ۱/۲ × Inch۸
vbPRPSExecutive	۷	۷ ۱/۲ ×Inch
vbPRPSA3	۸	A3 (۲۹۷× mm)
vbPRPSA4	۹	A4 (×۲۹۷ mm)
vbPRPSA۴Small	۱۰	A4 (اندازه کوچک)
vbPRPSA5	۱۱	A5 (۱۴۸×mm)

خصوصیت Port

به‌وسیله این خصوصیت می‌توان نام پورت مربوط به چاپگری را که چاپ به آن ارسال می‌شود به‌دست آورد. شکل کلی نحوه استفاده از این خصوصیت به این صورت است:

Printer.Port

نکته چاپگرها معمولا از پورت‌های موازی LPT1 و LPT2 استفاده می‌کنند.

خصوصیت PrintQuality

این خصوصیت کیفیت وضوح چاپ را در چاپگر معین می‌کند. شکل کلی نحوه استفاده از این خصوصیت به صورت زیر است:

Printer.PrintQuality = value

value یک ثابت عددی یا رشته‌ای است که میزان وضوح چاپ را معین می‌کند در صورت عدم استفاده از مقدار value مقدار فعلی خصوصیت بازگشت داده خواهد شد.

مقدار value می‌تواند یکی از مقادیر موجود در جدول ۱۱-۱۲ باشد.

جدول ۱۱-۱۲ مقادیر مربوط به کیفیت چاپ در خصوصیت PrintQuality

توضیح	ثابت عددی	ثابت رشته‌ای
چاپ با کیفیت Draft	-۱	vbPRPQDraft
چاپ با کیفیت پایین	-۲	vbPRPQLow
چاپ با کیفیت متوسط	-۳	vbPRPQMedium
چاپ با کیفیت بالا	-۴	vbPRPQHigh

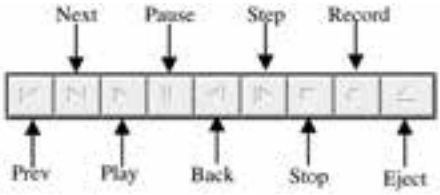
۱۲-۷ چندرسانه‌ای (Multimedia)

یکی از ویژگی‌های دیگر زبان برنامه‌نویسی ویژوال‌بیسیک توانایی استفاده از امکانات صوتی و تصویری در آن است. با استفاده از کنترل‌های ارائه شده در زبان برنامه‌نویسی ویژوال‌بیسیک می‌توان انواع فایل‌های ویدئویی، صوتی و موسیقی را مورد استفاده قرار داد.

۱۲-۷-۱ کنترل MCI (Media Control Interface)

با استفاده از این کنترل می‌توان انواع فایل‌های صدا، موسیقی، ویدئویی و سی‌دی‌های صوتی را پخش کرد.

این کنترل به‌طور پیش‌فرض در جعبه ابزار مشاهده نمی‌شود. برای اضافه کردن این کنترل گزینه Components را از منوی Project انتخاب کنید سپس در کادرمحاوره Components که نمایش داده می‌شود کادر علامت Microsoft Multimedia Control 6.0 را انتخاب کنید و روی دکمه OK کلیک کنید. در این مرحله کنترل MCI به کنترل جعبه ابزار اضافه می‌شود. کنترل MCI شامل ۹ دکمه است که وظایف مختلفی را به عهده دارند. عملکرد این دکمه‌ها در شکل ۱۲-۲۴ و مطابق جدول ۱۲-۱۴ قابل مشاهده است.



شکل ۱۲-۲۴

مثال ۳: می‌خواهیم پروژه‌ای طراحی کنیم که به‌وسیله آن بتوان فایل‌های ویدئویی از نوع avi را مشاهده کرد برای این کار عملیات بعد را به ترتیب انجام دهید:

۱- یک پروژه از نوع StandardEXE ایجاد کنید سپس خصوصیت‌های فرم را مطابق جدول ۱۲-۱۲ تنظیم کنید.

جدول ۱۲-۱۲ خصوصیات فرم

خصوصیت	مقدار
Name	frmmultimedia
Caption	Multimedia

۲ - یک کنترل منو و کادر تصویر با نام PicVideo مطابق شکل ۲۵-۱۲ روی فرم قرار دهید.

شکل ۲۵-۱۲

۳ - کنترل MCI را به جعبه ابزار اضافه کنید. در جعبه ابزار روی آیکن MMControl دابل‌کلیک کنید تا کنترل MCI روی فرم قرار گیرد، سپس خصوصیت Name آن‌را روی مقدار mcivideo و اندازه و موقعیت آن‌را مطابق شکل ۲۵-۱۲ تنظیم نمایید.

۴ - به ماژول فرم بروید و دستورات زیر را در رویداد Load فرم تایپ کنید.

```
mcivideo.DeviceType = "aviVideo"
mcivideo.hWndDisplay = picvideo.hWnd
```

دستور اول با استفاده از خصوصیت DeviceType کنترل MCI را برای نمایش فایل‌ها با قالب‌بندی از نوع AVI آماده می‌کند. با استفاده از این خصوصیت می‌توان انواع رسانه‌های مالتی‌مدیا را برای استفاده تنظیم کرد.این‌خصوصیت می‌تواند مقادیر ارائه‌شده در جدول ۱۳-۱۲ را کسب کند.

جدول ۱۳-۱۲

مقدار	توضیح
AVIVideo	پخش فایل‌های صوتی تصویری AVI
CDAudio	پخش سی‌دی صوتی
DigitalVideo	پخش فایل‌های ویدئویی دیجیتال
Videodisc	پخش دیسک‌های ویدئویی
WaveAudio	پخش فایل‌های صوتی از نوع WAV

دستور دوم در این رویداد نیز سبب می‌شود تصویر فایل ویدئویی در کنترل جعبه تصویر Picvideo نمایش داده شود.

۵ – رویداد کلیک گزینه mnuopen را به صورت زیر تنظیم کنید:

```
Private Sub mnuopen_Click()
 Dim strpath As String
 strpath = InputBox("Enter Path and Filename :", "Input Data")
 mcivideo.FileName = strpath
 mcivideo.Command = "Open"
End Sub
```

در این رویداد ابتدا با استفاده از یک کادر ورود داده، مسیر و نام فایل ویدئویی دریافت شده و در متغیر strpath ذخیره می‌شود سپس محتویات strpath در خصوصیت Filename کنترل MCI قرار می‌گیرد تا کنترل MCI بتواند به فایل ویدئویی دسترسی پیدا کند سپس خصوصیت Command کنترل MCI روی مقدار Open تنظیم می‌شود تا کنترل آماده پخش فایل ویدئویی شود. این خصوصیت می‌تواند فرمان‌های مختلفی را برای اجرا در کنترل مشخص کند. این فرمان‌ها در جدول ۱۴-۱۲ قابل مشاهده است.

جدول ۱۴-۱۲

مقدار	توضیح
Open	ابزار مورد نیاز برای کنترل MCI را باز می‌کند.
Close	ابزار مورد نیاز برای کنترل MCI را می‌بندد.
Play	فایل ویدئویی، صدا یا موسیقی را پخش می‌کند.
Pause	عملیات پخش یا ضبط را موقتاً قطع می‌کند.
Stop	عملیات ضبط یا پخش را قطع می‌کند.
Back	از موقعیت جاری کمی به عقب حرکت می‌کند.
Step	از موقعیت جاری کمی به جلو حرکت می‌کند.
Prev	به ابتدای شیار (Track) جاری حرکت می‌کند. اگر ۳ ثانیه از شیار جاری پخش شده باشد به ابتدای شیار قبلی باز می‌گردد.
Next	یک ترک به جلو حرکت می‌کند.
Record	عملیات ضبط را آغاز می‌کند.
Eject	سی‌دی را به درایو وارد یا از آن خارج می‌کند.

۶ – پروژه و فرم را با نام playvideo ذخیره کرده سپس برنامه را اجرا کنید.

۷ – روی گزینه Open از منوی File کلیک کرده، نام و مسیر فایل ویدئویی خود را که دارای پسوند avi است در کادر ورود داده تایپ کنید و روی دکمه OK کلیک نمایید تا به فرم برنامه بازگردید.

۸ – در این مرحله بعضی از دکمه‌های کنترل MCI فعال می‌شوند روی دکمه پخش (Play) کلیک کنید تا فایل ویدئویی مورد نظرتان در کنترل جعبه تصویر نمایش داده شود سپس عملکرد سایر دکمه‌های کنترل MCI را بررسی نمایید.

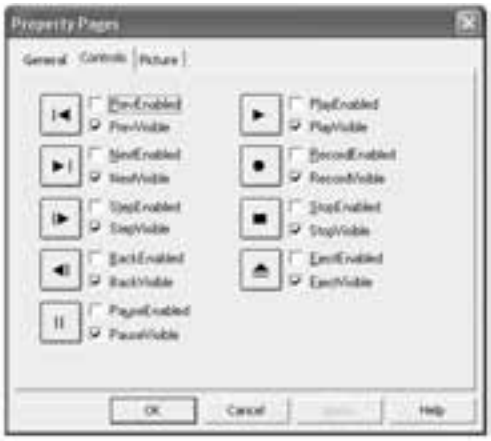
۹ – از برنامه خارج شده و به پنجره ویژوال‌بیسیک بازگردید.

تمرین:

پروژه playvideo را به‌گونه‌ای تغییر دهید که به وسیله آن بتوان فایل‌های صوتی با پسوند WAV را پخش کرد.

نکته در کنترل MCI با استفاده از خصوصیت Orientation می‌توان کنترل را به صورت افقی (mciOrientHorz) یا عمودی (mciOrientVert) تنظیم نمود.

نکته کلیه دکمه‌های موجود درکنترل MCI را می‌توان با استفاده از صفحه‌های خصوصیت (Property Pages) مرئی یا مخفی، فعال یا غیرفعال نمود. به منظور دسترسی به صفحه‌های خصوصیت در پنجره خصوصیات روی دکمه ... روبه‌روی خصوصیت Custom کلیک کنید سپس زبانه Controls را انتخاب کنید (شکل ۲۶-۱۲).



شکل ۲۶-۱۲

۱۲-۷-۲ کنترل Microsoft Active Movie (MAM)

با استفاده از این کنترل می‌توان انواع فایل‌های صوتی و موسیقی مانند WAV و MIDI را پخش کرد و فایل‌های تصویری مانند AVI، MPEG و MOV را نمایش داد.

این کنترل به طور عادی در جعبه ابزار مشاهده نمی‌شود. برای اضافه کردن این کنترل گزینه Components را از منوی Project انتخاب کنید سپس در کادرمحاوره Components که نمایش داده می‌شود کادر علامت Microsoft Active Movie Control را انتخاب کنید و روی دکمه OK کلیک کنید تا کنترل MAM به جعبه ابزار اضافه شود. این کنترل نیز مانند کنترل MCI دکمه‌های متعددی را شامل می‌شود که وظایف متفاوتی را بر عهده دارند (شکل ۲۷-۱۲).

شکل ۲۷-۱۲

در ایـــن کنترل به طور پیش‌فرض فقط دکمه‌های پخـــش (Play) و توقف (Stop) قابل اســـتفاده می‌باشد. به منظور دسترسی به سایر دکمه‌ها می‌توانید پس از انتخاب کنترل، در پنجـــره خصوصیـــات روی دکمه ... روبه‌روی خصوصیت Custom کلیک کنید. در کادر محـــاوره Property Pages روی زبانه Controls کلیک کنید (شـــکل ۲۸-۱۲). با اســـتفاده از کادرهای علامت موجود در این بخش می‌توان دکمه‌ها و سایر بخش‌های کنترل را مخفی یا قابل مشاهده کرد.

شکل ۲۸-۱۲

مثـــال ۴: می‌خواهیم پروژه‌ای طراحی کنیم که به وســـیله آن بتـــوان فایل‌های صوتی، موسیقی و ویدئویی را پخش کرد. برای این کار عملیات زیر را به ترتیب انجام دهید:

۱ - یک پروژه از نوع Standard EXE ایجاد کنید. خصوصیت‌های فرم را مطابق جدول ۱۵-۱۲ تنظیم کنید.

جدول ۱۵-۱۲

خصوصیت	مقدار
Name	frmmpeg
Caption	Movie

۲ - یک کنترل منو مطابق شکل ۲۹-۱۲ روی فرم قرار دهید سپس کنترل MAM را به جعبـــه ابـــزار اضافه کنید. در جعبه ابزار روی آیکن Active Movie دابل کلیک کنید تا کنترل روی فرم قرارگیرد سپس خصوصیت Name آنرا روی mammovie تنظیم کنید.

۳ - کنترل mammovie را انتخاب کرده و خصوصیت AutoStart آن را روی مقدار True تنظیم کنید تا در زمانی‌که فایل صوتی یا تصویری باز می‌شود کنترل MAM آنرا پخش کند.

شکل ۲۹-۱۲

۴ – خصوصیت ShowPositionControls را در کنترل mammovie روی مقدار True تنظیم کنید تا چهار دکمه دیگر نیز قابل مشاهده و استفاده شوند.

۵ – رویداد Click گزینه Open را به صورت زیر تنظیم کنید:

```
Private Sub mnuopen_Click()
  Dim strpath As String
  strpath = InputBox("Enter Path and Filename :", "Input Data")
  mammovie.FileName = strpath
End Sub
```

در این رویداد پس از دریافت نام و مسیر فایل موسیقی یا ویدئویی و ذخیره‌سازی آن در متغیر strpath، محتویات این متغیر در دستور سوم محتویات این متغیر در خصوصیت FileName کنترل mammovie قرار داده می‌شود تا آن‌را پخش کند.

۶ – پروژه و فرم را با نام activemovie ذخیره کرده سپس برنامه را اجرا کنید.

۷ – با استفاده از گزینه Open در منوی File، مسیر و نام یک فایل mpg را تعیین کنید. سپس در هنگام پخش فایل مورد نظرتان، عملکرد دکمه‌ها و سایر بخش‌ها در کنترل mammovie را بررسی نمایید.

۸ – از برنامه خارج شده و به پنجره ویژوال‌بیسیک بازگردید.

تمرین:

پروژه activemovie را برای انواع دیگر فایل‌ها آزمایش کنید.

۸–۱۲ شئ تصویر (Picture)

با استفاده از شئ تصویر نیز می‌توانید یک تصویر را روی فرم یا در کنترل کادر تصویر نمایش دهید. به منظور ایجاد یک شئ تصویر از نوع داده Picture استفاده می‌شود. با شئ تصویر می‌توان انواع فایل‌های گرافیکی مانند BMP، GIF، ICO و JPG را نمایش داد.

مثال ۵: می‌خواهیم رویه‌ای بنویسیم که بتواند یک تصویر را با استفاده از شئ تصویر نمایش دهد.

به این منظور می‌توان از یک شئ تصویر همراه با تابع LoadPicture استفاده کرد به این ترتیب رویه به صورت زیر تعریف می‌شود:

```
Public Sub showpicture(strpath As String)
    Dim objpicture As Picture
    Set objpicture = LoadPicture(strpath)
    Set picshow.Picture = objpicture
End Sub
```

این رویه با یک آرگومان (strpath) برای دریافت مسیر و نام فایل تعریف شده است سپس داخل رویه، یک شئ از نوع Picture با نام objpicture با دستور Dim تعریف شده است در مرحله بعد با استفاده از دستور Set و تابع LoadPicture تصویر موردنظر در شئ تصویر objpicture بارگذاری می‌شود و در پایان با دستور شئ تصویر در کنترل کادر تصویر picshow نمایش داده می‌شود.

برای فراخوانی رویه ShowPicture می‌توان به صورت زیر عمل نمود:

```
Call showpicture (" C:\WinXP\Coffee Bean.Bmp")
```

اگر لازم باشد تصویر روی فرم نمایش داده شود می‌توان از متد PaintPicture استفاده کرد. به عنوان مثال دستور سوم در رویه showpicture را می‌توان به این صورت تغییر داد تا تصویر روی فرم نمایش داده شود.

```
PaintPicture objpic, 20, 50, 3000, 4000
```

این دستور شئ تصویر را در مختصات X = ۲۰ و Y =۵۰ با عرض 3000 و ارتفاع 4000 روی فرم نمایش می‌دهد.

نکته • از متد PaintPicture می‌توان در کنترل کادر تصویر و شئ چاپگر نیز استفاده کرد.
• متد LoadPicture از انواع فایل‌های گرافیکی مانند BMP، ICO، GIF، JPG و ... پشتیبانی می‌کند.

تمرین:
پروژه‌ای طراحی کنید که با استفاده از آن بتوان هر تصویر دلخواهی را با استفاده از یک کادر محاوره Open مشاهده کرد به علاوه امکان بزرگ یا کوچک کردن تصویر موردنظر نیز وجود داشته باشد.

مثال ۶ : می‌خواهیم پروژه‌ای طراحی کنیم که به وسیله آن بتوان ترسیمات رسم شده روی فرم را در یک فایل ذخیره کرد. برای این کار عملیات زیر را به ترتیب انجام دهید:

۱ – یک پروژه از نوع Standard EXE ایجاد کنید و فرم و کنترل‌های آن‌را مطابق شکل ۱۲–۳۰ و جدول ۱۲–۱۶ و ۱۲–۱۷ تنظیم کنید.

شکل ۱۲–۳۰

جدول ۱۲–۱۶ خصوصیات فرم

خصوصیت	مقدار
Name	frmdrawing
Caption	Save Picture

جدول ۱۲–۱۷ خصوصیات کنترل‌ها

کنترل / خصوصیت	Command Button	Command Button	Picture Box
Name	cmddrawline	cmdsave	picdrawing
Caption	&Line	&Save	_____

۲ – رویداد Click دکمه Line را به صورت زیر تنظیم کنید.

```
Private Sub cmddrawline_Click()

ScaleMode = vbPixels
```

```
AutoRedraw = True
Line (10, 10)-(200, 200), vbRed
End Sub
```

در این رویداد عملیات رسم یک خط انجام می‌گیرد.

۳ – دستور زیر را در رویداد Click دکمه Save تایپ کنید:

```
SavePicture Image, "C:\drawing.bmp"
```

رویه فرعی SavePicture می‌تواند ترسیمات انجام شده روی فرم یا کنترل کادر تصویر را در یک فایل گرافیکی از نوع BMP ذخیره کند. آرگومان اول در این رویه فرعی خصوصیت Image فرم یا کنترل کادر تصویری است که ترسیمات روی آن انجام شده است و آرگومان دوم مسیر و نام فایلی است که ترسیمات در آن ذخیره می‌شود.

۴ – پروژه و فرم را با نام savepicture ذخیره کرده سپس آن‌را اجرا کنید و ابتدا روی دکمه Line و سپس Save کلیک کنید. در پایان به مسیر C:\ بروید و فایل drawing.bmp را با برنامه نقاشی ویندوز باز کرده و محتویات آن‌را بررسی کنید.

۵ – از برنامه خارج شوید و به پنجره ویژوال‌بیسیک بازگردید.

تمرین:

پروژه‌ای طراحی کنید که به‌وسیله آن بتوان هر نوع فایل گرافیکی مانند JPG، GIF، CUR، ICO و ... را در فایل جدیدی با قالب‌بندی BMP ذخیره کرد.

Learn in English

Circle Method

Draws a circle, ellipse, or arc on an object.

Syntax

object.**Circle** [*Step*] (x, y), radius, [*color, start, end, aspect*]

The **Circle** method syntax has the following object qualifier and parts:

Part	Description
object	Optional. Object expression that evaluates to an object in the Applies To list. If object is omitted, the **Form** with the focus is assumed to be object.
Step	Optional. Keyword specifying that the center of the circle, ellipse, or arc is relative to the current coordinates given by the **CurrentX** and **CurrentY** properties of object.
(x, y)	Required. **Single** values indicating the coordinates for the center point of the circle, ellipse, or arc. The **ScaleMode** property of object determines the units of measure used.
radius	Required. **Single** value indicating the radius of the circle, ellipse, or arc. The **ScaleMode** property of object determines the unit of measure used.
color	Optional. **Long** integer value indicating the RGB color of the circle>s outline. If omitted, the value of the **ForeColor** property is used. You can use the **RGB** function or **QBColor** function to specify the color.
start, end	Optional. Single–precision values. When an arc or a partial circle or ellipse is drawn, start and end specify (in radians) the beginning and end positions of the arc. The range for both is –2 pi radians to 2 pi radians. The default value for start is ۰ radians; the default for end is 2 * pi radians.
aspect	Optional. Single–precision value indicating the aspect ratio of the circle. The default value is 1,0, which yields a perfect (non–elliptical) circle on any screen.

واژه‌نامه

دایره	Circle
جزء	Component
جاری	Current
رسم‌کردن، کشیدن	Draw
خط	Line
فیلم	Movie
چند رسانه‌ای	Multimedia
جهت	Orientation
نقطه	Point
کیفیت	Quality
شعاع دایره	Radius
مقیاس	Scale
سبک، روش	Style

خلاصه مطالب

- ترسیمات گرافیکی را می‌توان روی فرم یا کنترل PictureBox یا چاپگر ایجاد کرد.
- به‌وسیله خصوصیت ScaleMode می‌توان مقیاس را در محورهای مختصات تنظیم کرد.
- به‌وسیله خصوصیت ScaleLeft و ScaleTop می‌توان مختصات نقطه مبنا را تنظیم کرد.
- خصوصیات ScaleHeight و ScaleWidth مقیاس را می‌توان در سیستم مختصات فرم یا کنترل PictureBox تنظیم کرد.
- به‌وسیله خصوصیت Scale نیز می‌توان مختصات نقطه مبنا و مقیاس سیستم مختصات را در فرم یا کنترل مورد نظر تنظیم کرد.
- به‌وسیله متد PSet می‌توان هر نقطه دلخواهی را در موقعیت مورد نظر ترسیم کرد.
- به‌وسیله متد Line می‌توان انواع خطوط مستطیل و مستطیل توپر را در مکان مورد نظر رسم کرد.
- متد Circle می‌تواند انواع دایره، بیضی یا کمانی از دایره و یا بیضی را رسم کند.
- به‌وسیله متد Point می‌توان رنگ یک نقطه را به‌دست آورد. این متد رنگ نقطه مورد نظر را به‌صورت یک عدد از نوع Long باز می‌گرداند.
- متدهای CurrentX و CurrentY موقعیت جاری مکان‌نما را در صفحه ترسیمات معین می‌کنند. خصوصیت CurrentX موقعیت جاری مکان‌نما را در محور X و CurrentY موقعیت جاری مکان نما را در محور Y معین می‌کنند.

- متد Cls می‌تواند صفحه ترسیمات را پاک کند.
- به‌وسیله متد Print هر نوع عبارت، می‌توان مقدار متغیرها و خصوصیات را نمایش داد.
- با استفاده از تابع Spc در متد Print می‌توان به تعداد مورد نظر فضای خالی ایجاد کرد.
- با استفاده از تابع Tab در متد Print می‌توان اطلاعات نمایشی را در ستون مورد نظر نمایش داد.
- متدهای TextWidth و TextHeight می‌توانند به ترتیب عرض و ارتفاع عبارت رشته‌ای را که دریافت می‌کنند معین کنند.
- به‌وسیله خصوصیت AutoRedraw می‌توان از حذف ترسیمات موجود در یک پنجره جلوگیری به عمل آورد.
- خصوصیت DrawMode می‌تواند رنگ ترسیمات گرافیکی را تعیین کند.
- به‌وسیله خصوصیت DrawStyle می‌توان نوع خطوط را در متدهای گرافیکی نظیر Line و Circle تعیین کرد.
- با استفاده از خصوصیت DrawWidth می‌توان ضخامت خطوط و نقاط را در متد Line، Circle و PSet مشخص کرد.
- خصوصیت FillStyle، نوع و حالت خطوطی که سطح یک شکل گرافیکی نظیر مستطیل، دایره یا بیضی را پر می‌کند تعیین می‌کند.
- به‌وسیله خصوصیت FillColor می‌توان رنگ مورد نظر را برای پوشاندن سطح یک شکل گرافیکی نظیر مستطیل، دایره یا بیضی تعیین کرد.
- تابع QBColor می‌تواند با دریافت یک عدد صحیح، رنگ متناظر آن را به‌صورت یک عدد از نوع Long تعیین کند.
- به‌وسیله تابع RGB می‌توانید ترکیبات رنگی مورد نظرتان را بر اساس مقدار سه رنگ اصلی آبی، قرمز و سبز ایجاد کنید.
- با استفاده از شئ Printer و خصوصیات و متدهای این شئ، می‌توان اطلاعات مورد نظر را به‌وسیله چاپگر روی کاغذ چاپ کرده و بر نحوه انجام عملیات چاپ نظارت کرد.
- به‌وسیله کنترل MCI و MAM می‌توان انواع فایل‌های صدا، موسیقی و ویدئویی را پخش کرد.
- به‌وسیله رویه‌های LoadPicture و PaintPicture می‌توان فایل‌های گرافیکی را روی کنترل کادر تصویر یا فرم نمایش داد.
- به‌وسیله رویه SavePicture می‌توان ترسیمات روی فرم یا کنترل کادر تصویر را روی دیسک ذخیره کرد.

آزمون نظری

۱ – به‌وسیله کدام خصوصیت می‌توان یکی از مقیاس‌های استاندارد را در ویژوال‌بیسیک انتخاب کرد؟

الف– ScaleMode ب– thgScaleHei ج– ScaleWidth د– Scale

۲ – کدام گزینه برای رسم یک مستطیل توپر توسط متد Line مناسب است؟

الف– B ب– BF ج– F د– FB

۳ – کدام تابع در متد Print می‌تواند اطلاعات نمایشی را در ستون مشخصی نمایش دهد؟

الف– Spc ب– Point ج–Tab د– Spcb

۴ – خروجی فرمان,,, 2 1000 ,(150,200) Circle به صورت..... است.

الف– بیضی افقی ب– بیضی عمودی

ج– نیمدایره د– ربع دایره

۵ – به‌وسیله کدام خصوصیت می‌توان ضخامت ترسیمات را تعیین کرد؟

الف– DrawMode ب– DrawStyle

ج– DrawWidth د– FillStyle

۶ – کدام متد می‌تواند عملیات چاپ را خاتمه دهد؟

الف– EndDoc ب– Port ج– NewPage د– KillDoc

۷ – کدام تابع می‌تواند ترکیبات رنگی را بر اساس رنگ‌های اصلی ایجاد کند؟

الف– QBColor ب– BackColor ج– RGB د– ColorMode

۸ – کدام خصوصیت، رنگ مورد نظر برای پر کردن یک مستطیل یا دایره را معین می‌کند؟

الف– FillStyle ب– FillColor

ج– ColorMode د– ForeColor

۹ – به‌وسیله کدام خصوصیت موقعیت جاری مکان نما در صفحه ترسیمات در جهت محور افقی معین می‌شود؟

الف– CurrentX ب– CurrentY

ج– Point د– گزینه‌های الف و ب صحیح هستند.

۱۰ – واحد اندازه‌گیری پیش‌فرض در سیستم مختصات ویژوال‌بیسیک عبارت است از:

الف– cm ب– inch ج– pixel د– twip

۱۱ – رویه SavePicture ترسیمات را به صورت فایل‌های گرافیکی از نوع ..... ذخیره می‌کند.

الف– JPG ب– BMP ج– ICO د– CUR

۱۲ – کدام یک از کنترل‌ها امکان پخش فایل‌های ویدئویی با پسوند MPG را دارند؟

الف– MCI ب– Image ج– MAM د– Picture

۱۳ - Which of the following answer can be used in circle method to draw an ellipse?

a- radius b- start c- aspect d- end

۱۴ - سیستم مختصات در ویژوال‌بیسیک را توضیح داده و نحوه تنظیم آن را بیان کنید.

۱۵ – نحوه ساخت و استفاده از شئ تصویر را توضیح دهید.

۱۶ – نحوه استفاده از شئ چاپگر را بیان کرده و خصوصیت‌ها و متدهای آن را توضیح دهید.

۱۷ – کاربرد متدهای PSet، Line، Circle و Point را با ذکر مثال توضیح دهید.

۱۸ – کنترل‌های چند رسانه‌ای را در ویژوال‌بیسیک نام برده و نحوه استفاده از آن‌ها را برای پخش فایل‌های موسیقی و نمایش فیلم توضیح دهید.

۱۹ – خصوصیت‌های CurrentX، CurrentY، DrawMode، DrawStyle و DrawWidth را توضیح دهید.

۲۰ – نحوه ذخیره‌سازی ترسیمات گرافیکی را با استفاده از شئ تصویر توضیح دهید.

آزمون عملی

۱ – پروژه‌ای طراحی کنید که شکل زیر را نمایش دهد.

۲ – پروژه‌ای طراحی کنید که کاربر بتواند انواع بیضی، دایره و کمان را با توجه به مقادیر دلخواهش مشاهده کند.

۳ – پروژه‌ای طراحی کنید که با دریافت معدل دانش‌آموزان ۵ کلاس ۲۰ نفره میانگین معدل هر کلاس را محاسبه کند, سپس نتایج را به‌صورت نمودار میله‌ای نمایش دهد.

۴ – پروژه‌ای طراحی کنید که به طور پیوسته دایره‌ها با اندازه و رنگ‌های متفاوت به صورت تصادفی ایجاد و نمایش داده شوند.

۵ – پروژه‌ای طراحی کنید که مختصات سه رأس یک مثلث را دریافت نموده، سپس آن‌را ترسیم کند.

۶ – پروژه‌ای طراحی کنید که حرکت یک اتومبیل را شبیه‌سازی کند.

۷ – پروژه‌ای طراحی کنید که به‌وسیله آن بتوان هر تصویر دلخواه را با اندازه موردنظر مشاهده کرد و در صورت تمایل آن‌را به‌صورت فایل گرافیکی از نوع BMP ذخیره نمود.

واحد کارسیزدهم

هدف جزیی

توانایی انجام یک پروژه عملی

هدف‌های رفتاری

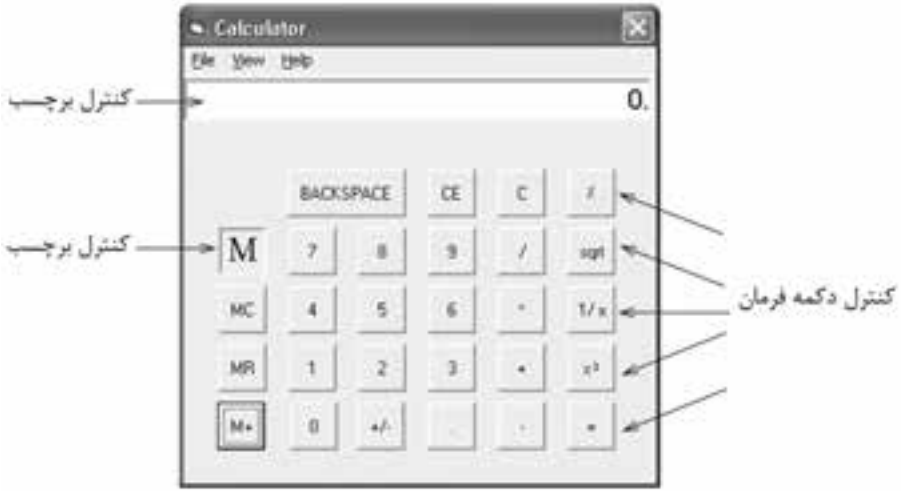
پس از مطالعه این واحد کار از فراگیر انتظار می‌رود که:

۱ – توانایی انجام یک پروژه برنامه‌نویسی را داشته باشد.

در این واحد کار یک پروژه برنامه‌نویسی واقعی را به صورت عملی انجام خواهید داد تا آنچه را که قبلاً فراگرفته‌اید به کار بگیرید. یک ماشین حساب مطابق شکل ۱–۱۳ ایجاد کنید که توانایی انجام محاسبات چهار عمل اصلی، محاسبه مجذور و جذر، قرینه اعداد، درصد و معکوس اعداد را مانند یک ماشین حساب الکترونیکی داشته باشد. برای این کار عملیات زیر را به ترتیب انجام دهید:

شکل ۱–۱۳

۱ – برنامه ویژوال‌بیسیک را اجرا کرده، یک پروژه از نوع Standard EXE ایجاد کنید که شامل یک فرم، کنترل دکمه‌های فرمان، دو برچسب مطابق شکل ۲–۱۳ و یک کنترل زمان‌سنج باشد، سپس خصوصیات آن‌ها را مطابق جداول ۱–۱۳ و ۲–۱۳ تنظیم کنید.



شکل ۲–۱۳

دکمه‌های فرمان دارای عرض و ارتفاع ۵۰۰ بوده و عنوان، نام و خصوصیات Top و Left آن‌ها مطابق شکل ۲-۱۳ است و خصوصیت Cancel تمام آن‌ها را مساوی False و خصوصیت Default آن‌ها (بجز کنترل =) را نیز روی False تنظیم کنید.

جدول ۱-۱۳ خصوصیات فرم

خصوصیت	مقدار
Name	frmmain
BorderStyle	1-Fixed Single
Caption	Calculator

جدول ۲-۱۳ خصوصیات کنترل‌ها

کنترل / خصوصیت	Label	Label	Timer
Name	lblvalue	lblmemory	tmrdelay
Alignment	1	2	-
BorderStyle	1	1	-
Enabled	True	True	False
Interval	-	-	150
AutoSize	False	False	-
Caption	0	بدون عنوان	-

به بخش تعاریف فرم بروید و متغیرهای مورد نیاز برنامه‌ها را به صورت زیر تعریف کنید:

```
Dim dblnol As Double, dblno2 As Double
Dim dblresult As Double, dblmemory As Double
Dim strop As String, strno As String
```

می‌بینید که متغیرهای مختلفی در بخش تعاریف، تعریف می‌شوند. متغیرهای dblno1 و dblno2 از نوع Double و به ترتیب برای ذخیره‌سازی عددی که قبل از انتخاب عملگر و عددی که بعد از انتخاب عملگر وارد می‌شوند، به کار می‌روند و متغیر dblresult از نوع Double و برای نگهداری نتیجه محاسبات روی متغیرهای dblno1 و dblno2 استفاده می‌شوند.

به‌علاوه متغیر dblmemory به عنوان حافظه ماشین حساب و متغیر رشته‌ای strop برای ذخیره کردن عمل ریاضی که کاربر انتخاب کرده، استفاده می‌شوند و به عنوان آخرین متغیر، یک متغیر رشته‌ای دیگر به نام strno عدد وارد شده به ماشین حساب را به صورت یک رشته در خود قرار می‌دهد تا در جریان اجرای برنامه با تبدیل آن به مقدار عددی در متغیرهای dblno۱ و dblno۲ ذخیره شود.

برای جلوگیری از تکرار دستورات در رویه‌های دیگر لازم است دو رویه را قبل از سایر رویه‌ها ایجاد نمایید. بنابراین اولین رویه را با نام dellblvalue و به صورت زیر تعریف کنید:

```
Private Sub dellblvalue( )
    If (strop = "*" Or strop = "/" Or strop = "+" Or strop = "-") And strno = "" Then
        lblvalue.Caption = ""
    End If
End Sub
```

این رویه بررسی می‌کند که آیا قبل از ورود یک رقم، یکی از دکمه‌های چهار عمل اصلی فشرده شده است یا خیر. چون در صورت انتخاب یک عملگر، ورود هر یک از ارقام به عنوان عدد دوم محسوب می‌شود، در نتیجه باید عدد وارد شده که قبل از انتخاب عملگر مورد نظر در کنترل برچسب نمایش داده می‌شود، پاک شود. این عمل با دستور lblvalue.Caption= "" انجام خواهد شد. پس از تعریف رویه dellblvalue که در رویدادهای Click ارقام ۱ الی ۹ و نقطه اعشار اجرا می‌شود، رویه دوم را با نام inputnumber و به صورت زیر تعریف کنید. این رویه وظیفه دریافت ارقام و تشکیل عدد مورد نظر کاربر را به عهده دارد.

```
Private Sub inputnumber (strdigit As String)
  If strdigit <> "." Then
    If Left(strno, 1) = "0" And Mid(strno,2, 1) < > "." Then
      strno = Mid(strno, 2) + strdigit
    Else
      strno = strno + strdigit
    End If
Else
  If InStr(1, strno, ".") = 0 Then
    If Len(lblvalue.Caption) = 0 Then strno ="0." Else _
```

```
                strno=strno +"."
  End If
    End If
  lblvalue.Caption = strno
End Sub
```

ایـــن رویه یک آرگومان به نام strdigit از نوع رشـــته‌ای دارد کـــه در زمان فراخوانی با توجه به این که روی کدام دکمه کلیک شـــده اســـت، رقم متناظر با آن را به رویه می‌دهد. پس از کلیک روی یکی از ارقام یک تا ۹ یا نقطه اعشار، رویه فراخوانی می‌شود و کاراکتر متناظر با رقم مربوطه در آرگومان strdigit ذخیره می‌شود. سپس با استفاده از یک دستور If مقدار این آرگومان بررسی می‌شود و در صورتی که مقدار آن مخالف کاراکتر نقطه باشد به این معنی اســـت که دکمه مربوطه یکی از ارقام یک تا ۹ می‌باشـــد، در نتیجه If موجود در بخش Then اجرا می‌شود و رشته strno را که مجموعه ارقام وارد شده می‌باشد، مورد بررسی قرار می‌دهد.

با استفاده از توابع Left و Mid رشته strno بررسی می‌شود. اگر اولین و دومین کاراکتر رشـــته strno برابر با صفر و نقطه باشـــند ("0.") به این معنی است که عدد بین صفر و یک اســـت، بنابراین رقم strdigit به رشـــته "0." اضافه می‌شـــود. اما در صورت نادرست بودن شـــرط رقم strdigit به وسیله دستور strno=strno + strdigit به مجموعه ارقام قبلی در رشته strno اضافه می‌شود. در واقع این If برای کنترل ارقام صفری است که قبل از نقطه اعشار و سایر ارقام غیرصفر وارد می‌شوند. به این صورت از دریافت رقم صفر قبل از سایر ارقام غیرصفر جلوگیری به عمل می‌آید.

امـــا اگر دکمه کلیک شـــده، نقطه اعشـــار باشـــد If موجـــود در بخـــش Else اولین If اجرا می‌شـــود. ایـــن If از ورود نقطه اعشـــار اضافه جلوگیری می‌کند. با اســـتفاده از تابع InStr وجـــود نقطـــه اعشـــار در ارقامی که تاکنون وارد شـــده‌اند یعنی متغیر strno بررســـی می‌شـــود. اگـــر رشـــته strno فاقد نقطه اعشـــار باشـــد If موجـــود در آن اجرا می‌شـــود و با اســـتفاده از تابع Len طول رشـــته بررســـی می‌شـــود و اگر این مقدار برابر صفر باشد به این معنی اســـت عدد با مقدار .0 شـــروع می‌شـــود، در غیر این صورت نقطه اعشـــار با دســـتور strno= strno+"0" بـــه ارقـــام قبلی اضافه خواهد شـــد. در پایان این رویـــه، ارقامی که تاکنون وارد شـــده‌اند با اســـتفاده از دســـتور lblvalue.Caption=strno در کنترل برچسب نمایش داده می‌شـــوند. بـــه این ترتیب می‌تـــوان انواع اعـــداد کوچک‌تر یا بزرگ‌تر از یـــک را وارد کرد.

در این مرحله امکان ورود اعداد را فراهم کرده، رویداد Click دکمه‌های رقمی ۱ تا ۹ و نقطه اعشار را به صورت زیر تنظیم کنید:

```
Private Sub cmd1_Click()
  Call dellblvalue
  Call inputnumber ("1")
End Sub
Private Sub cmd2_Click()
  Call dellblvalue
  Call inputnumber ("2")
End Sub

Private Sub cmd3_Click()
  Call dellblvalue
  Call inputnumber ("3")
End Sub

Private Sub cmd4_Click()
  Call dellblvalue
  Call inputnumber ("4")
End Sub

Private Sub cmd5_Click()
 Call dellblvalue
 Call inputnumber ("5")
End Sub

Private Sub cmd6_Click()
 Call dellblvalue
 Call inputnumber ("6")
End Sub

Private Sub cmd7_Click()
  Call dellblvalue
  Call inputnumber ("7")
End Sub
```

```
Private Sub cmd8_Click()
  Call dellblvalue
  Call inputnumber ("8")
End Sub

Private Sub cmd9_Click()
  Call dellblvalue
  Call inputnumber ("9")
End Sub

Private Sub cmddot_Click()
  Call dellblvalue
  Call inputnumber (".")
End Sub
```

۵ – در این مرحله رویداد دکمه فرمان رقم صفر را به صورت زیر تنظیم کنید:

```
Private Sub cmd0_Click()
If strop="*" Or strop = "+" Or strop="-" Or Stop="/" Then
    If Left(strno, 1)<>"0" Or Mid(strno, 2, 1)= "." Then
      strno = strno + "0"
    Else
     strno = ""
     lblvalue.Caption = ""
    End If
Else
    If Left(strno,1)<> "0" Or Mid(strno, 2,1) = "." Then strno=strno+"0"
End If
 lblvalue.Caption = strno
End Sub
```

این رویداد از یک دستور If تشکیل شده است که در بخش Then و Else آن نیز مجدداً یک دستور If به کار رفته است. در اولین دستور If فشرده شدن یکی از دکمه‌های مربوط به چهار عمل اصلی بررسی می‌شود. در صورتی که هیچ یک از آن‌ها استفاده نشده باشند نتیجه شرط نادرست بوده و If موجود در بخش Else اجرا می‌شود. در این If نیز با استفاده از تابع Left و Mid بررسی می‌شود که آیا اولین کاراکتر در متغیر strno صفر یا دومین کاراکتر در آن نقطه اعشار است یا خیر؟

۶ – اکنون رویداد Click مربوط به چهار عمل اصلی را به صورت زیر تنظیم کنید:

```
Private Sub cmdmultiply_Click()
  If strop = "" Then
    strop = "*"
    dblnol = Val(strno)
    strno = ""
Else
  If strno <> "" Then
    dblno2 = Val(strno)
    Call Compute
    dblno1 =Val(strno)
    strno = ""
  End If
  strop = "*"
 End If
End Sub

Private Sub cmddivision_Click()
  If strop = "" Then
    strop = "/"
    dblnol = Val(strno)
    strno = ""
Else
  If strno <> "" Then
    dblno2 = Val(strno)
    Call Compute
    dblno1 =Val(strno)
    strno = ""
  End If
  strop = "/"
 End If
End Sub

Private Sub cmdplus_Click()
  If strop = "" Then
```

```
    strop = "+"
    dblnol = Val(strno)
    strno = ""
Else
  If strno <> "" Then
    dblno2 = Val(strno)
    Call compute
    dblno1 =Val(strno)
    strno = ""
  End If
  strop = "+"
 End If
End Sub

Private Sub cmdminus_Click()
If strop = "" Then
    strop = "-"
    dblnol = Val(strno)
    strno = ""
Else
      If strno <> "" Then
          dblno2 = Val(strno)
          Call compute
          dblno1 =Val(strno)
          strno = ""
       End If
   strop = "-"
End If
End Sub
```

دستورات این چهار رویداد مشابه یکدیگر هستند. در این رویدادها ابتدا با استفاده از دستور If مقدار متغیر strop بررسی می‌شود تا مشخص شود تاکنون عملگری انتخاب شده است یا خیر (چون ممکن است عبارتی که به ماشین حساب داده می‌شود مشابه= 4 +3*2 یا مانند عبارت= 3 * 2 باشـــد). در صورتی که شـــرط درست باشد (یعنی عملگری تاکنون انتخاب نشـــده است) ابتدا کاراکتر متناسب با دکمه‌ای که کلیک شده است در متغیر strop

ذخیره می‌شود، مثلاً اگر روی دکمه * کلیک کنید کاراکتر "*" در strop ذخیره می‌شود، سپس عددی که قبل از کلیک روی چهار عمل اصلی وارد شده است و در متغیر strno نگهداری می‌شود با استفاده از تابع Val به نوع عددی تبدیل و در متغیر dblno1 به عنوان عدد اول ذخیره می‌شود و مجدداً مقدار "" در strno ذخیره می‌شود تا مقدار متغیر رشته‌ای strno برای دریافت عدد دوم از بین برود.

اما اگر قبلاً یکی از چهار عمل اصلی انتخاب شده باشند دیگر مقدار strop برابر رشته "" نیست و دستورات بخش Else اجرا می‌شوند. در این‌جا یک If دیگر استفاده می‌شود و اگر مقداری در متغیر strno موجود باشد به این معنی است که عدد دوم نیز وارد شده است. بنابراین در صورتی که عباراتی مانند 3+4*2 یا مشابه آن‌ها وارد شوند به این معنی است که قبل از دریافت عدد۴ باید محاسبه 3*2 انجام شود، پس عدد دوم در متغیر dblno2 ذخیره خواهد شد و با فراخوانی رویه compute محاسبات انجام می‌شود. رویه compute را نیز به صورت زیر در ماژول فرم ایجاد کنید.

```
Private Sub compute()
  Select Case strop
    Case "*": dblresult = dblnol * dblno2
    Case "/": dblresult = dblnol / dblno2
    Case "+": dblresult = dblnol + dblno2
    Case "-": dblresult = dblnol - dblno2
    Case "power": dblresult = dblnol ^ dblno2
  End Select
  lblvalue.Caption = dblresult
  strno = dblresult
End Sub
```

با فراخوانی این رویه و با استفاده از یک دستور Select Case، مقدار متغیر strop بررسی می‌شود تا با توجه به عمل ریاضی انتخاب شده اعمال ضرب، تقسیم، جمع یا تفریق روی اعداد dblno1 و dblno2 انجام شده و نتیجه در متغیر dblresult ذخیره شود. سپس مقدار این متغیر در خصوصیت Caption برچسب ذخیره می‌شود تا نتیجه محاسبات در پنجره ماشین حساب دیده شود و به همین صورت این مقدار برای انجام محاسبات بعدی در متغیر strno ذخیره می‌شود.

به این ترتیب پس از خاتمه اجرای رویه compute نتیجه محاسبات که در متغیر strno

ذخیره شده در متغیر dblno1 نیز قرار داده می‌شود تا در محاسبات بعدی به عنوان اولین عدد مورد استفاده قرار گیرد و پس از اجرای strno="" و حذف نتیجه محاسبات از متغیر strno دستورات If موجود در Else تمام می‌شود و به عنوان آخرین دستور در این بخش، کاراکتر متناسب با عمل ریاضی انتخاب شده در متغیر strop برای استفاده در مراحل بعدی ذخیره می‌شود. به این ترتیب یک دوره کامل از ورود اعداد و انجام محاسبات، کدنویسی و آماده شده است.

۷ – اکنون باید ماشین حساب را به‌گونه‌ای تنظیم کنید که در صورت استفاده از عباراتی مانند= 12.5+6 نیز محاسبات به درستی انجام شود، بنابراین رویداد Click دکمه فرمان تساوی را به صورت زیر تنظیم کنید:

```
Private Sub cmdequal_Click()
  If strop="/" Or strop="+" Or strop="-" Or strop="*" Then
    dblno2 = Val(strno)
    Call compute
  End If
  strop = ""
End Sub
```

در این رویداد نیز ابتدا مقدار متغیر strop بررسی می‌شود تا در صورت کلیک روی یکی از دکمه‌های چهار عمل اصلی محاسبات انجام شود، بنابراین در صورت درست بودن شرط موجود در دستور If ابتدا عدد دوم در متغیر dblno2 ذخیره می‌شود، سپس رویه compute فراخوانی می‌شود تا محاسبات مربوطه انجام شود و محتویات متغیر strop برای مراحل بعدی خالی شود. ذکر این نکته ضروری است که عدد اول نیز در این حالت در هنگام کلیک روی یکی از چهار عمل اصلی در متغیر dblno1 ذخیره می‌شود.

۸ – در این مرحله رویداد Click دکمه BACKSPACE را به صورت زیر تنظیم کنید:

```
Private Sub cmdbackspace_Click()
  If Len (lblvalue.Caption) = 1 Then
    lblvalue.Caption = "0"
    strno=""
Else
    lblvalue.Caption = Left(lblvalue.Caption, Len (lblvalue.Caption)-1)
    strno = lblvalue.Caption
  End If
End Sub
```

می‌بینید که این رویه نیز با یک دستور If آغاز می‌شود و با بررسی طول رشته‌ای که در کنترل برچسب نمایش داده می‌شود عمل حذف کاراکترها را از مقدار موجود، انجام می‌دهد. بنابراین اگر طول این رشته برابر یک باشد شرط درست بوده و مقدار خصوصیت Caption کنترل برچسب روی صفر تنظیم شده سپس دستور strno = "" برای اعمال تغییرات در محاسبات نیز اجرا می‌شود. در غیر این صورت با استفاده از توابع Len و Left یک کاراکتر از سمت راست رشته حذف می‌شود و مقدار خصوصیت Caption در متغیر strno نیز ذخیره می‌شود تا تغییرات انجام شده در محاسبات نیز اعمال شود.

۹ – اکنون رویداد مربوط به دکمه درصد را به صورت زیر تنظیم کنید:

```
Private Sub cmdpercent_Click()
   lblvalue.Caption = Val(lblvalue.Caption) * 0.01
   strno = lblvalue.Caption
   strop = ""
End Sub
```

در این رویداد نیز ابتدا با تبدیل خصوصیت Caption کنترل برچسب به عدد و ضرب آن در عدد ۰/۰۱ مقدار را براساس درصد محاسبه کرده و سپس مقدار به دست آمده در متغیر strno برای محاسبات بعدی ذخیره می‌شود و با اجرای دستور strno = "" از ایجاد تداخل و اشتباه که ممکن است در اثر انتخاب چند عملگر رخ دهد، جلوگیری به عمل می‌آید.

۱۰ – مانند مرحله ۸ رویداد Click مربوط به دکمه‌های به توان‌رسانی(X^2)، جذر (sqrt)، قرینه‌سازی (+/-) و معکوس‌سازی(1/x) را به این صورت تنظیم کنید. این چهار رویه عملکردی مشابه رویه رویداد Click دکمه درصد دارند.

```
Private Sub cmdpower_Click()
   lblvalue.Caption = Val(lblvalue.Caption) ^ 2
   strno = lblvalue.Caption
   strop= ""
End Sub

Private Sub cmdsign_Click()
 If strno <> "" Then
   lblvalue.Caption = -Val(lblvalue.Caption)
   strno = lblvalue.Caption
 End If
End Sub
```

```
Private Sub cmdsqrt_Click()
    lblvalue.Caption = Sqr(Val(lblvalue.Caption))
    strno = lblvalue.Caption
    strop= ""

End Sub
Private Sub cmdinverse_Click()
    If Val(lblvalue.Caption) <> 0 Then lblvalue.Caption=1/ Val(lblvalue.Caption)
    strno = lblvalue.Caption
    strop= ""
End Sub
```

۱۱ – اکنــون رویه‌های رویداد مربوط به دکمه‌هــای+M, MR و MC را به صورت زیر تنظیم کنید. دکمه+M برای ذخیره اعداد و نتیجه محاسبات در حافظه ماشین حساب, دکمه MR برای استفاده ازمحتویات حافظه ماشین‌حساب درعملیات موردنظر و دکمه MC برای پاک کردن حافظه ماشین حساب به‌کار می‌روند.

```
Private Sub cmdm_Click()
  If strop = "/" Or strop = "+" Or strop = "-" Or strop="*" Then
     dblno2 = Val(strno)
     Call compute
  End If
  dblmemory = dblmemory + Val(lblvalue.Caption)
  lblmemory.Caption = "M"
  strop= ""
End Sub

Private Sub cmdmc_Click()
   dblmemory = 0
   lblmemory.Caption = ""
End Sub

Private Sub cmdmr_Click()
   lblvalue.Caption = str(dblmemory)
   If dblmemory=0 Then
     strno= ""
   Else
```

```
    strno=str(dblmemory)
  End If
End Sub
```

رویه() cmdm_Click برای ذخیره مقادیر در حافظه ماشین حساب به کار می‌رود و با یک دستور If شروع می‌شود تا در صورت استفاده از اعمال ریاضی قبل از کلیک روی دکمه M+، ابتدا محاسبات با فراخوانی رویه compute انجام شود، سپس نتیجه عملیات به محتویات قبلی حافظه ماشین حساب که در متغیر dblmemory قرار دارد، اضافه می‌شود و با نمایش کاراکتر M در کنترل برچسب مربوطه، کاربر از ذخیره شدن اطلاعات در حافظه ماشین حساب مطلع می‌شود. البته در صورت عدم انتخاب یک عمل ریاضی دستورات بخش If اجرا نشده و طبیعی است که محاسباتی نیز انجام نمی‌شود، اما عدد موجود در نمایشگر ماشین حساب به حافظه اضافه خواهد شد. آخرین دستور در این رویه یعنی strop= "" نیز همان کاربرد قبلی در سایر رویه‌های مشابه را دارد.
رویه() cmdmc_Click برای حذف مقدار ذخیره شده در حافظه ماشین حساب به کار می‌رود و با اجرای آن ابتدا مقدار dblmemory صفر می‌شود سپس حرف M از کنترل برچسب مربوط به حافظه برداشته می‌شود.

رویه() cmdmr_Click نیز برای استفاده از مقدار ذخیره شده در حافظه در زمان انجام محاسبات به کار می‌رود و با تبدیل مقدار متغیر dblmemory به مقدار رشته‌ای، آن‌را در کنترل برچسب ماشین حساب نمایش داده و بعد دستور If را برای اعمال تغییرات در محاسبات اجرا می‌کند.

۱۲ – در این مرحله رویداد دکمه‌های C و CE را به صورت زیر تنظیم کنید. همان‌طور که می‌دانید دکمه C تمام اعداد ورودی, عملگرهای انتخاب شده و محتویات حافظه ماشین حساب را پاک می‌کند و دکمه CE عمل ریاضی انتخاب شده پس از ورود اولین عدد را از بین می‌برد، اما عدد اول وارد شده را حذف نمی‌کند.

```
Private Sub cmdc_Click()
    strno = ""
    lblvalue.Caption = "0"
    strop = ""
    dblno1 = 0
    dblno2 = 0
    dblresult = 0
    dblmemory = 0
```

```
    lblmemory.Caption = ""
End Sub
Private Sub cmdce_Click()
  If strop="/" Or strop="+" Or strop ="-" Or strop="*" Then
     lblvalue.Caption=dblno1
     strno=dblno1
     strop=""
  End If
End Sub
```

رویداد() cmdc_Click مربوط به دکمه C می‌شود و همان‌طور که می‌بینید کلیه متغیرهای عددی، رشته‌ای، حافظه و نمایشگر ماشین حساب را پاک می‌کند.

رویـــداد() cmdce_Click نیز نحوه انتخاب عملگرهای ریاضی را بعد از ورود عدد اول بررسی می‌کند و در صورت درست بودن شرط، متغیر dblno۱ را در خصوصیت Caption کنترل برچســـب ماشین حساب قرار داده و متغیر strop را خالی می‌کند تا عملگر انتخاب شده از بین رفته و عملیات به مرحله بعد از ورود عدد اول بازگردد.

۱۳ – اکنـــون باید رویدادهای صفحـــه کلید را نیز به‌گونه‌ای تنظیم کنید که در صورت استفاده از صفحه کلید نیز ماشین حساب قابل استفاده باشد. برای این کار ابتدا خصوصیت KeyPreview را برای فرم پروژه روی مقدار True و خصوصیت Style کلیه دکمه‌های فرمان را روی مقـــدار 1-Graphical تنظیم کنید، ســـپس رویـــداد KeyPress فرم را به صورت زیر تنظیم نمایید:

```
Private Sub Form_KeyPress(KeyAscii As Integer)
   Select Case KeyAscii
      Case 8:
         Call cmdbackspace_Click
         cmdbackspace.BackColor = vbRed
         tmrdelay.Enabled = True
      Case 48:
         Call cmd0_Click
         cmd0.BackColor = vbRed
         tmrdelay.Enabled = True
      Case 49:
         Call cmd1_Click
         cmd1.BackColor = vbRed
```

```
    tmrdelay.Enabled = True
Case 50:
    Call cmd2_Click
    cmd2.BackColor = vbRed
    tmrdelay.Enabled = True
Case 51:
    Call cmd3_Click
    Cmd3.BackColor = vbRed
    tmrdelay.Enabled = True
Case 52:
    Call cmd4_Click
    cmd4.BackColor = vbRed
    tmrdelay.Enabled = True
Case 53:
    Call cmd5_Click
    cmd5.BackColor = vbRed
    tmrdelay.Enabled = True
Case 54:
    Call cmd6_Click
    cmd6.BackColor = vbRed
    tmrdelay.Enabled = True
Case 55:
    Call cmd7_Click
    cmd7.BackColor = vbRed
    tmrdelay.Enabled = True
Case 56:
    Call cmd8_Click
    cmd8.BackColor = vbRed
    tmrdelay.Enabled = True
Case 57:
    Call cmd9_Click
    cmd9.BackColor = vbRed
    tmrdelay.Enabled = True
Case 43:
    Call cmdplus_Click
```

```
    cmdplus.BackColor = vbRed
    tmrdelay.Enabled = True
Case 45:
    Call cmdminus_Click
    cmdminus.BackColor = vbRed
    tmrdelay.Enabled = True
Case 46:
    Call cmddot_Click
    Cmddot.BackColor = vbRed
    tmrdelay.Enabled = True
Case 42:
    Call cmdmultiply_Click
    cmdmultiply.BackColor = vbRed
    tmrdelay.Enabled = True
Case 47:
    Call cmddivision_Click
    cmddivision.BackColor = vbRed
    tmrdelay.Enabled = True
Case 37:
    Call cmdpercent_Click
    cmdpercent.BackColor = vbRed
    tmrdelay.Enabled = True
Case 67, 99:
    Call cmdc_Click
    cmdc.BackColor = vbRed
    tmrdelay.Enabled = True
Case 69, 101:
    Call cmdce_Click
    cmdce.BackColor = vbRed
    tmrdelay.Enabled = True
Case 80, 112:
    Call cmdpower_Click
    cmdpower.BackColor = vbRed
    tmrdelay.Enabled = True
Case 83, 115:
    Call cmdsqrt_Click
```

```
        cmdsqrt.BackColor = v
        bRed
        tmrdelay.Enabled = True
    Case 73, 105:
        Call cmdinverse_Click
        cmdinverse.BackColor = vbRed
        tmrdelay.Enabled = True
    Case 68, 100:
        Call cmdmc_Click
        cmdmc.BackColor = vbRed
        tmrdelay.Enabled = True
    Case 77, 109:
        Call cmdm_Click
        cmdm.BackColor = vbRed
        tmrdelay.Enabled = True
    Case 82, 114:
        Call cmdmr_Click
        cmdmr.BackColor = vbRed
        tmrdelay.Enabled = True
    Case 71, 103:
        Call cmdsign_Click
        cmdsign.BackColor = vbRed
        tmrdelay.Enabled = True
    Case 61:
        Call cmdequal_Click
        cmdequal.BackColor = vbRed
        tmrdelay.Enabled = True
  End Select
End Sub
```

مشـــاهده می‌کنید که با استفاده از یک دســـتور Select Case مقدار آرگومان KeyAscii رویداد KeyPress بررســـی می‌شود و با توجه به کلید فشرده شده رویداد Click متناظر با کلیدی که قبلاً طراحی شده است، فراخوانی می‌شود. این کار سبب می‌شود تا از تکرار دســـتورات و شلوغ شدن ماژول فرم جلوگیری شود و در هر Case رویداد Click مربوط به دکمه‌ای که فشرده شده است فراخوانی شود. پس از خاتمه اجرای رویه رویداد کلید

فشرده شده، با استفاده از خصوصیت BackColor کنترل مربوطه، رنگ کنترل به قرمز تغییر داده می‌شود. سپس خصوصیت Enabled کنترل زمان‌سنج روی True تنظیم می‌شود تا زمان‌سنج شروع به کار کند و پس از گذشت زمان کوتاهی رنگ کنترل را به حالت اول بازگرداند و خصوصیت Enabled خود را False می‌کند تا برای دفعه بعد آماده شود. به این صورت با فشردن کلیدهای صفحه کلید روی پنجره ماشین حساب، فشرده شدن کلیدها با تغییر رنگ قابل مشاهده است. سپس رویداد Timer کنترل زمان‌سنج را به صورت زیر تنظیم کنید:

```
Private Sub tmrdelay_Timer()
    cmddot.BackColor = &H8000000F
    cmd0.BackColor = &H8000000F
    cmd1.BackColor = &H8000000F
    cmd2.BackColor = &H8000000F
    cmd3.BackColor = &H8000000F
    cmd4.BackColor = &H8000000F
    cmd5.BackColor = &H8000000F
    cmd6.BackColor = &H8000000F
    cmd7.BackColor = &H8000000F
    cmd8.BackColor = &H8000000F
    cmd9.BackColor = &H8000000F
    cmdplus.BackColor = &H8000000F
    cmdminus.BackColor = &H8000000F
    cmdmultiply.BackColor = &H8000000F
    cmddivision.BackColor = &H8000000F
    cmdbackspace.BackColor = &H8000000F
    cmdc.BackColor = &H8000000F
    cmdce.BackColor = &H8000000F
    cmdpercent.BackColor = &H8000000F
    cmdpower.BackColor = &H8000000F
    cmdsqrt.BackColor = &H8000000F
    cmdinverse.BackColor = &H8000000F
    cmdmc.BackColor = &H8000000F
    cmdm.BackColor = &H8000000F
    cmdmr.BackColor = &H8000000F
```

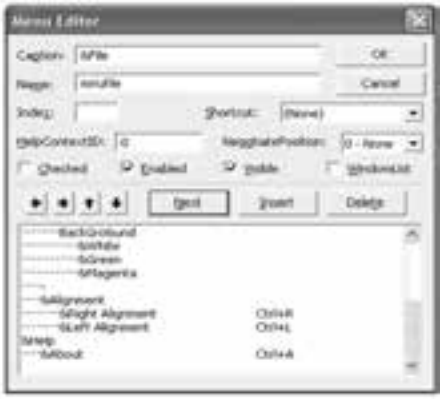
```
    cmdsign.BackColor = &H8000000F
    cmdequal.BackColor = &H8000000F
    tmrdelay.Enabled = False
End Sub
```

۱۴ - رویـــداد Load فرم را نیز به صورت زیـــر تنظیم کنید تا مقادیر متغیرها مقداردهی اولیه شود:

```
Private Sub Form_Load()
    cmdc _Click()
End Sub
```

۱۵ - نوار منو و گزینه‌های موجود در آن‌را مطابق شکل ۳–۱۳ و ۴–۱۳ ایجاد کنید. روی کادر علامـــت Enabled گزینـــه Black و White کلیـــک کنیـــد و آن‌ها را از حالت انتخاب خـــارج کنید. به علاوه روی کادر علامت Checked گزینه Right Alignment کلیک کنید و آن‌را انتخاب نمایید.

شکل ۳–۱۳



شکل ۴–۱۳

۱۶ – اکنون به ماژول فرم بروید و رویداد Click گزینه‌های Clear و Exit را به صورت نشـــان داده شـــده در صفحه بعد تنظیم کنید. گزینه Clear همان عملکرد دکمه C را دارد و انتخاب گزینه Exit ســـبب نمایش یک کادر پیغام خواهد شـــد که در صورت کلیک روی دکمه OK برنامه خاتمه یافته و در صورت کلیک روی دکمه Cancel کاربر به پنجره ماشین حساب باز می‌گردد.

```
Private Sub mnuclear_Click()
    Call cmdc_Click
End Sub
```

```
Private Sub mnuexit_Click()
    Dim intanswer As Integer
    intanswer = MsgBox("Do You Want To Exit ?",vbOKCancel _
                    +vbInformation, "Exit")
    If intanswer = vbOK Then Unload Me
End Sub
```

۱۷-در این مرحله رویداد مربوط به گزینه‌های زیرمنوهای ForeGround و BackGround را برای تغییر رنگ قلم و زمینه نمایشگر ماشین حساب، به صورت زیر تنظیم کنید:

```
Private Sub mnublack_Click()
    lblvalue.ForeColor = vbBlack
    mnublack.Enabled = False
    mnublue.Enabled = True
    mnured.Enabled = True
End Sub

Private Sub mnublue_Click()
    lblvalue.ForeColor = vbBlue
    mnublack.Enabled = True
    mnublue.Enabled = False
    mnured.Enabled = True
End Sub

Private Sub mnured_Click()
    lblvalue.ForeColor = vbRed
    mnublack.Enabled = True
    mnublue.Enabled = True
    mnured.Enabled = False
End Sub

Private Sub mnuwhite_Click()
    lblvalue.BackColor = vbWhite
    mnuwhite.Enabled = False
    mnugreen.Enabled = True
    mnumagenta.Enabled = True
End Sub
```

```
Private Sub mnugreen_Click()
    lblvalue.BackColor = vbGreen
    mnuwhite.Enabled = True
    mnugreen.Enabled = False
    mnumagenta.Enabled = True
End Sub

Private Sub mnumagenta_Click()
    lblvalue.BackColor = vbMagenta
    mnuwhite.Enabled = True
    mnugreen.Enabled = True
    mnumagenta.Enabled = False
End Sub
```

۱۸ -اکنــون رویدادClick مربوط به زیرمنــوی Alignment را تنظیم کنید. برای این کار رویه‌های دو گزینه این زیر منو را به صورت زیر تنظیم کنید:

```
Private Sub mnuleft_Click()
    lblvalue.Alignment = vbLeftJustify
    mnuleft.Checked = True
    mnuright.Checked = False
End Sub

Private Sub mnuright_Click()
    lblvalue.Alignment = vbRightJustify
    mnuleft.Checked = False
    mnuright.Checked = True
End Sub
```

در این صورت با انتخاب هر یک از این گزینه‌ها، چک مارک در کنار گزینه‌ای که کلیک شده است، قرار داده می‌شود و چک مارک گزینه دیگر حذف می‌شود. به علاوه با توجه به گزینه کلیک شــده، خصوصیت Alignment نمایشگر ماشین حساب نیز تنظیم می‌شود.

۱۹ – در این مرحله لازم است تا برای گزینه About در منوی Help دستورات لازم را تایپ کنید. هدف از این گزینه ارایه شــماره نگارش نرم‌افزار به کاربر است، بنابراین یک فرم با مشخصات ارایه شده در جدول ۳-۱۳ و مطابق با شکل ۵-۱۳ به پروژه اضافه کنید.

شکل ۵-۱۳

جدول ۳-۱۳

کنترل / خصوصیت	Image	Label	Command Button	Form
Name	imgcalculate	lblversion	cmdok	frmabout
Caption	_	Calculator Version 1-1383	&OK	About
Stretch	True	_	_	_
BorderStyle	_	۱	_	۴
Alignment	_	۲	_	_

۲۰ – پس از اضافه کردن فرم جدید آن‌را با نام About ذخیره کنید و سپس رویداد Click دکمه OK و Unload فرم About را به صورت زیر تنظیم کنید:

```
Private Sub cmdok_Click()
   Unload Me
End Sub

Private Sub Form_Unload(Cancel As Integer)
    frmmain.Enabled = True
End Sub
```

۲۱ – در این مرحله رویه رویداد Click گزینه About را از منوی Help، به صورت زیر تنظیم کنید. اولین دستور سبب نمایش فرم About می‌شود و دستور دوم، فرم اصلی را غیرفعال می‌کند و تا زمانی که کاربر روی دکمه OK فرم About کلیک نکند یا آن را نبندد نمی‌تواند از ماشین‌حساب استفاده کند.

```
Private Sub mnuabout_Click()
    frmabout.Show
```

```
    frmmain.Enabled = False
End Sub
```

۲۲ – پروژه و فرم اصلی را با نام calculator ذخیره کنید، سپس آن‌را اجرا کرده و برای حالت‌های مختلف آزمایش کنید.

۲۳ – به اجرای پروژه خاتمه داده و پنجره ویژوال‌بیسیک را ببندید.

آزمون پایانی (نظری)

۱ – مقدار K پس از خاتمه حلقه زیر چقدر است؟

```
K=1
Do  While  (K<3)
  K=K+ 1
Loop
```

الف– ۴ ب– ۳ ج– ۲ د– ۱

۲ – در صورتی‌که Option Base 1 باشد تعداد اعضای آرایه A (5) چقدر خواهد بود؟

الف– ۵ ب– ۶ ج– ۷ د– ۸

۳ – خروجی دستور Right («AliReza», 3) چیست؟

الف– «Ali» ب– "eza" ج– «iReza» د– «AliRe»

۴ – کدام نوع داده می‌تواند هر نوع داده‌ای را نگهداری کند؟

الف– Variant ب– Single ج– Boolean د– Long

۵ – کدام خاصیت کنترل کادر متن، تراز متن را در کنترل تعیین می‌کند؟

الف– Alignment ب– MaxLength

ج– BorderStyle د– Appearance

۶ – خروجی دستور WeekdayName (Weekday (Now)) چیست؟

الف– شماره روز جاری ب– شماره هفته جاری

ج– نام روز جاری د– نام ماه جاری

۷– اگر آرایه grade(10To17,2To9) تعریف شود، آنگاه حاصل LBound(grade,2) چیست؟

الف– ۱۰ ب– ۱۷ ج– ۲ د– ۹

۸ – خروجی دستور Replace («This is a book.», «i», «I», , , vbBinaryCompare) چیست؟

الف– «ThIs Is a book.» ب– «This Is a book.»

ج– «s is a book.» د– «s Is a book.»

۹ – با فرض این‌که strname = "SuperComputer" باشد، حاصل عبارت

Left (Right (strname , 5),3) چیست؟

الف– «pu» ب– «Sup»

ج– «per» د– «put»

۱۰ – کدام خاصیت در کنترل کادر لیست، شماره عضو انتخاب شده در آن را تعیین می کند؟

الف– ListIndex ب– ListCount

ج– Index د– Item

۱۱ – کدام کاراکتر در متد Print سبب می‌شود اطلاعات نمایشی بدون فاصله در کنار هم قرار گیرند؟

الف– ; ب– ,

ج– ! د– #

۱۲ – حاصل عبارت(InStr(5,»hardwAre and software» , «ware» , 0 چیست؟

الف– ۵ ب– صفر

ج– ۱۸ د– ۱

۱۳ – در صورتی‌که یک متغیر در بخش تعاریف ماژول فرم با دستور Dim معرفی شود ..... .

الف– در زمان اجرای برنامه پیام خطا نمایش داده می‌شود.

ب– متغیر در تمام رویه‌های ماژول کد قابل استفاده است.

ج– متغیر در تمام رویه‌های فرم قابل استفاده است.

د– متغیر در تمام ماژول‌های پروژه قابل استفاده است.

۱۴ – حاصل عبارت(MonthName (Month(Date) , True چیست؟

الف– تاریخ جاری سیستم

ب– نام ماه جاری به صورت کامل

ج– نام ماه جاری به صورت خلاصه

د– شماره ماه جاری

۱۵ – در صورت تعریف یک رویه با کلمه کلیدی Public ..... .

الف– رویه مورد نظر در تمام ماژول‌های فرم و ماژول‌های کد قابل استفاده است.

ب– رویه مورد نظر در تمام ماژول‌های فرم قابل استفاده است.

ج– امکان فراخوانی آن به‌وسیله رویه‌های رویداد وجود ندارد.

د- امکان فراخوانی آن به‌وسیله رویه‌های آماده ویژوال‌بیسیک وجود ندارد.

۱۶ - حاصل عبارت Asc(Chr(65)) چیست؟

الف- «A»

ب- ۶۵

ج- یک رشته خالی

د- پیام خطا نمایش داده می‌شود.

۱۷ - یک رویه فرعی با کدام‌یک از عبارت‌های زیر خاتمه می‌یابد؟

الف- Function ب- Call

ج- End Sub د- Sub

۱۸ - کدام خاصیت در کنترل MCI امکان نمایش فیلم را در کنترل PicturBox فراهم می‌کند؟

الف- DeviceType ب- hWndDisplay

ج- FileName د- Command

۱۹ - کدام کاراکتر برای تعریف یک متغیر از نوع رشته‌ای به کار می‌رود؟

الف- & ب- #

ج- ! د- $

۲۰ - حاصل عبارت "A">"B" And 5<3 Or (True) Not چیست؟

الف- True ب- False

ج- صفر د- ۱

۲۱ - اگر الگوریتم زیرمجموع و میانگین تعداد مشخصی از اعداد را محاسبه کند، کدام گزینه در مورد مرحله ۶ درست است؟

۱- شروع

۲- I ← ۱ و Sum ← ۰ و c ← ۰

۳- N را دریافت کن

۴- M را دریافت کن

۵- Sum ← Sum + M

۶- .....

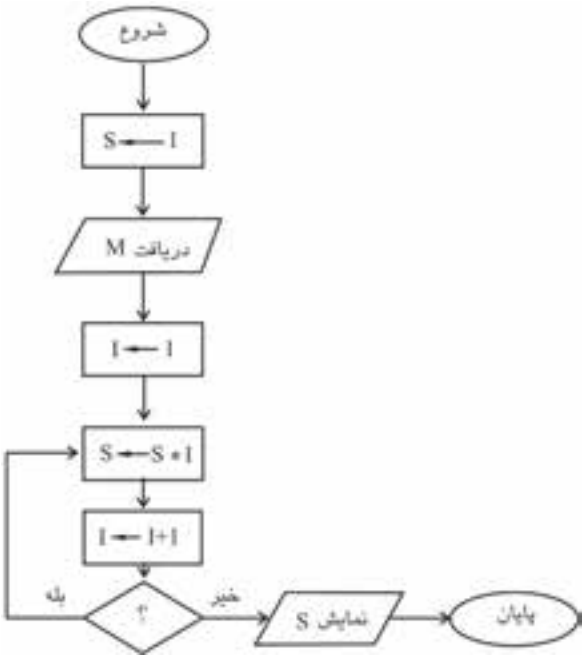
۷- اگر I < =N آن‌گاه برو به مرحله ۴

۸- $\frac{Sum}{N}$ را نمایش بده

۹- پایان

الف- I ← I + ۱ ب- I ← I - ۱ ج- N ← N + ۱ د- N ← Sum - ۱

۲۲ – در صورتی‌که بخواهیم حاصل‌ضرب اعداد ۱ تا M را به دست آوریم کدام گزینه را به‌جای علامت سؤال قرار می‌دهیم؟



الف- I <= M

ب- I > = M

ج- M > I

د- M < I

۲۳ – کدام رویداد توانایی تشخیص کلیدهای تابعی را ندارد؟

الف- KeyUp ب- KeyDown

ج- KeyPress د- KeyUp و KeyDown

۲۴ – الگوریتم زیر چه عملی انجام می‌دهد؟

۱- شروع

۲- x را دریافت کن

۳- اگر x < ۰ است آن‌گاه x ← -x

۴- x را نمایش بده

۵- پایان

الف- محاسبه قدر مطلق x ب- قرینه اعداد مثبت

ج- جزء صحیح x د- الگوریتم هیچ عملی انجام نمی‌دهد.

۲۵ – کدامیک از موارد زیر از ویژگی‌های الگوریتم محسوب نمی‌شود؟

الف– رعایت ترتیب　　ب– خاتمه پذیر بودن

ج– ارایه جزییات کافی　　د– ارایه فرمول‌های پیچیده

۲۶ – کدام نوع داده برای ذخیره‌سازی سن یک فرد مناسب‌تر است؟

الف– Integer　　ب– Long　　ج– Word　　د– Byte

۲۷ – کـدام خاصیت، اولویت انجـام رویدادهای صفحه کلید را بین فرم و کنترل‌های موجود در آن تعیین می‌کند؟

الف– خاصیت Key در فرم　　ب– خاصیت KeyPreview در فرم

ج– خاصیت Key در کنترل‌ها　　د– خاصیت KeyPreview در کنترل‌ها

۲۸ – حاصل عبارت3 + 2* 18\3 چیست؟

الف– ۱۳　　ب– ۱۴

ج– ۱۵　　د– ۱۶

۲۹ – عنوان یک فرم به وسیله کدام خاصیت تعیین می‌شود؟

الف– Enabled　　ب– Text

ج– Name　　د– Caption

۳۰ – به‌وسیله کدام متد می‌توان رنگ یک نقطه از تصویر را به‌دست آورد؟

الف– PSet　　ب– Point

ج– Tab　　د– Spc

آزمون پایانی (عملی)

پروژه‌ای طراحی کنید که یک پازل را شبیه‌سازی کند.

پاسخنامه

پیش‌آزمون

۱– ب)　　۲– ب)　　۳– ب)

آزمون نظری واحد کار ۱

۱– ج)　　۲– الف)　　۳ – ج)　　۴– ب)
۵– الف)　　۶– الف)　　۷– الف)　　۸– د)
۹– الف)　　۱۰– ب)　　۱۱ – c)

آزمون نظری واحد کار ۲

۱– ج)　　۲–ب)　　۳– د)　　۴–ب)
۵– ب)　　۶– الف)　　۷– الف)　　۸– ج)
۹– الف)　　۱۰ – ب)　　۱۱– b)

آزمون نظری واحد کار۳

۱– ب)　　۲– ب)　　۳ – ب)　　۴– الف)
۵– ب)　　۶– ج)　　۷– د)　　۸– ج)

۹– ج)　　۱۰– ب)　　۱۱– د)　　۱۲– الف)
۱۳– ب)　　۱۴– الف)　　۱۵– ب)　　۱۶– d)
۱۷– b)

آزمون نظری واحد کار ۴

۱– ج)　　۲– الف)　　۳– ب)　　۴ – ب)
۵– ج)　　۶– ج)　　۷– الف)　　۸– ب)
۹– c)

آزمون نظری واحد کار ۵

۱–ج)　　۲– ب)　　۳– ب)　　۴– الف)
۵– ب)　　۶– الف)　　۷– ج)　　۸– الف)
۹– د)　　۱۰–ج)　　۱۱– b)

آزمون نظری واحد کار ۶

۱– ب)　　۲– الف)　　۳– ج)　　۴– ج)
۵– الف)　　۶– د)　　۷– ب)　　۸– ج)

۹- ب)　۱۰- ج)　۱۱- ب)　۱۲- ج)
۱۳- الف)　۱۴- d)

آزمون نظری واحد کار ۷

۱- ب)　۲- ب)　۳- د)　۴- ب)
۵- الف)　۶- ب)　۷- ب)　۸- ب)
۹- د)　۱۰- ج)　۱۱- الف)　۱۲- د)
۱۳- c)

آزمون نظری واحد کار ۸

۱- ج)　۲- ب)　۳- الف)　۴- ب)
۵- ج)　۶- ب)　۷- ج)　۸- ج)
۹- a)

آزمون نظری واحد کار ۹

۱- ب)　۲- ج)　۳- ب)　۴- الف)
۵- ب)　۶- ب)　۷- ج)　۸- الف)
۹- الف)　۱۰- الف)　۱۱- b)

آزمون نظری واحد کار ۱۰

۱- ب)　۲- الف)　۳- ب)　۴- ب)
۵- د)　۶- ج)　۷- د)　۸- ب)
۹- d)

آزمون نظری واحد کار ۱۱

۱- ب)　۲-د)　۳- ج)　۴- ب)
۵- الف)　۶- ج)　۷- الف)　۸- ب)
۹- د)　۱۰- الف)　۱۱- b)

آزمون نظری واحد کار ۱۲

۱- الف)　۲-ب)　۳- ج)　۴- ب)
۵- ج)　۶- د)　۷- ج)　۸- ب)
۹- الف)　۱۰- د)　۱۱- ب)　۱۲- ج)
۱۳- c)

آزمون پایانی (نظری)

۱– ب)	۲– الف)	۳– ب)	۴– الف)
۵– الف)	۶– ج)	۷– ج)	۸– الف)
۹– د)	۱۰– الف)	۱۱– الف)	۱۲– ج)
۱۳– ج)	۱۴– ج)	۱۵– الف)	۱۶– ب)
۱۷– ج)	۱۸– ب)	۱۹– د)	۲۰– ب)
۲۱ – الف)	۲۲– الف)	۲۳–ج)	۲۴– الف)
۲۵– د)	۲۶– د)	۲۷– ب)	۲۸– ج)
۲۹– د)	۳۰– ب)		

ضمیمه ۱
نصب نرم‌افزار Visual Basic 6.0

برای نصب زبان برنامه‌نویسی ویژوال‌بیسیک ابتدا باید دیسک‌های CD نرم‌افزار **Microsoft Visual Studio 6** نگارش نهایی (Enterprise Edition) را تهیه کنید. در این‌جا چگونگی نصب این نرم‌افزار از روی دیسک CD، توضیح داده می‌شود. مدل‌های دیگر نیز تقریباً دارای مراحل نصب مشابهی هستند.

CD نصب نرم‌افزار را داخل درایو قرار دهید. اگر CD شما از نوع Aut - Run باشد بعد از قرار گرفتن CD در درایو مربوطه، برنامه نصب ویندوز به طور خودکار اجرا می‌شود و اولین کادر محاوره مربوط به نصب نرم‌افزار ظاهر می‌شود، اما در صورتی که برنامه نصب به طور خودکار اجرا نشد برنامه My Computer یا Window Explorer را اجرا کرده و در ریشه درایو CD خود، فایل Setup. exe را پیدا کنید و سپس روی آن دابل کلیک کنید. برنامه نصب اجرا شده و اولین کادر محاوره نصب نمایش داده می‌شود (شکل ۱).

شکل ۱

اکنون به ترتیب این مراحل را دنبال کنید:

۱ – ابتدا روی دکمه Next در اولین کادر محاوره کلیک کنید.

۲ – پس از مشاهده شکل ۲ روی دکمه انتخاب I accept the agreement و سپس روی دکمه Next کلیک نمایید.

نکته در صورت تمایل می‌توانید در هر کادرمحاوره با کلیک روی دکمه Back <به کادر محاوره قبل بازگردید.

شکل۲

۳ – در شکل ۳ در دو کادر متنی که در بخش:Please enter your product>s ID number کادر محاوره قرار گرفته‌اند، شـــماره (ID) نرم‌افزار را تایپ کنید. سپس در کادر متن‌های Your name: و : Your company>s name اســـامی دلخواه خود را بنویسید و در پایان مجدداً روی دکمه Next کلیک کنید.

شکل۳

نکته در صورت نادرست بودن شماره ID امکان نصب برنامه وجود ندارد. اگر شماره ID ارایه شـــده در این‌جا قابل اســـتفاده نباشد، می‌توانید از فایل‌های موجود در CD، شماره ID را پیدا کنید یا آن را از فروشنده درخواست کنید.

۴ – در کادرمحاوره شکل ۴ روی دکمه Next کلیک کنید.

۵ – در کادرمحاوره شکل ۵ روی دکمه OK کلیک کنید تا کامپیوتر مجدداً راه‌اندازی شود

شکل۴

شکل۵

۶ – پس از راه‌اندازی مجدد کامپیوتر کمی صبر کنید، کادرمحاوره‌ای که ظاهر می‌شود (شـــکل ۶) کـــه روش‌های مختلفی را بـــرای نصب نرم‌افزار پیشـــنهاد می‌کند، روی دکمه انتخاب Custom و سپس روی دکمه Next کلیک کنید.

۷ – در کادرمحاوره شـــکل ۷ می‌توانید مسیر نصب نرم‌افزار را انتخاب کنید. برای این کار مسیر دلخواه خود را در کادر متن تایپ کنید یا با استفاده از دکمه Browse این کار را انجام دهید.

شکل۶

شکل۷

در ایـــن کادر محـــاوره حداقل فضای مورد نیـــاز برای نصب نرم‌افـــزار و فضای آزاد درایوی که نرم‌افزار روی آن نصب می‌شـــود، قابل مشـــاهده است. توجه داشته باشید که فضای آزاد روی درایو مورد نظر کمتر از حداقل فضای پیشنهادی نباشد.

در پایان این مرحله روی دکمه Next کلیک کنید تا کادرمحاوره بعد (شکل ۸) ظاهر شود.

شکل۸

۸ – در این مرحله، مدتی صبر کنید تا کادرمحاوره بعدی نمایش داده شود.

شکل ۹

۹ – در این کادر محاوره (شـــکل ۹) روی دکمه Continue کلیک کنید تا کادر محاوره مطابق شـــکل ۱۰ مشاهده شـــود. در این کادر محاوره روی دکمه OK کلیک کنید تا روند نصب نرم‌افزار ادامه یابد.

شکل ۱۰

نکته اگر مایل هستید عملیات نصب را خاتمه دهید، می‌توانید در کادر محاوره شکل ۹ روی دکمه Exit Setup کلیک کنید و در کادر محاوره‌ای که ظاهر می‌شود (شکل ۱۱) روی دکمه Exit Setup کلیک نمایید و اگر می‌خواهید مجدداً به کادر محاوره قبل (شکل ۹) بازگردید روی دکمه Resume Setup کلیک کنید.

شکل ۱۱

پس از کلیک روی دکمه OK، پنجره نصب نرم‌افزار به صورت شکل ۱۲ در می‌آید. مدتی صبر کنید تا وارد مرحله بعد شوید.

شکل ۱۲

۱۰ – اگر نسخه‌های دیگری از بسته نرم‌افزاری Visual Studio یا همین نگارش را قبلاً نصب کرده باشید، کادر محاوره‌ای مشابه شکل ۱۳ نمایش داده می‌شود. برای اطمینان بیشتر از ایجاد تغییرات در بخش‌های دیگر سیستم خود، در صورت مشاهده این کادر محاوره روی دکمه No کلیک کنید.

شکل ۱۳

۱۱- پس از انجام مرحله ۱۰ پنجره نصب برنامه مجدداً به صورت شکل ۱۲ مشاهده می‌شود، مدتی صبر کنید تا کادر محاوره بعدی مطابق شکل ۱۴ نمایش داده شود.

شکل ۱۴

۱۲-در این مرحله (شکل ۱۴) شما باید زبان‌های برنامه‌نویسی، مانند زبان‌های برنامه‌نویسی VB6، VC++ 6 و VFoxPro6 اجزای گرافیکی مانند انواع فایل‌های گرافیکی که در پروژه‌های برنامه‌نویسی می‌توان از آن‌ها استفاده نمود و سایر ابزارهای مورد نیاز خود را برای نصب انتخاب کنید. چون در این کتاب، زبان برنامه‌نویسی ویژوال‌بیسیک ۶ مورد نظر است؛ بنابراین با کلیک روی کادر علامت مربوط به موارد زیر آن‌ها را انتخاب کنید یا سایر موارد را در لیست این کادر محاوره از حالت انتخاب خارج کنید.

کادرهای علامتی که باید انتخاب شوند عبارتند از:

Microsoft Visual Basic 6.0
ActiveX
Data Access
Graphics
Tools

پس از انجام تنظیمات گفته شده، کادر محاوره مربوطه به صورت شکل ۱۵ در می‌آید

شکل ۱۵

در خاتمه این مرحله روی دکمه Continue کلیک کنید.

نکته • هر یک از گروه‌های موجود در لیست، به چند جزء کوچک‌تر تقسیم می‌شوند که اگر بخواهید در یک گروه فقط بعضی از اجزای آن را نصب کنید، می‌توانید از دکمه Change Option در کادر محاوره شکل ۱۴ استفاده کنید. البته پیشنهاد می‌شود از دستکاری این بخش‌ها بدون آگاهی کافی خودداری نمایید.

• اگر بخواهید تمام گروه‌ها را در لیست ۱۴ انتخاب کنید، روی دکمه Select All کلیک کنید.

• در صورتی که بخواهید مسیر نصب هر یک از گروه‌ها را روی دیسک سخت تغییر دهید می‌توانید از دکمه Change Folder... در کادر محاوره شکل ۱۴ استفاده کنید.

۱۳ – پس از انجام مرحله ۱۲، ابتدا برنامه نصب فضای لازم را برای نصب نرم‌افزار محاسبه می‌کند (شکل ۱۶) و در صورت وجود فضای آزاد به اندازه مناسب، مرحله انجام عملیات کپی فایل‌های نرم‌افزار مطابق شکل ۱۷ آغاز می‌شود.

شکل ۱۶

۱۴ – پس از خاتمه مرحله ۱۳ کادر محاوره‌ای مطابق شکل ۱۸ نمایش داده می‌شود که بیانگر به روز کردن اطلاعات مورد نیاز سیستم است.

۱۵ – در مرحله بعد کادر محاوره‌ای مطابق شکل ۱۹ نمایش داده می‌شود، مبنی بر این‌که عمل نصب با موفقیت انجام شده است. پس از کلیک روی دکمه OK کادر محاوره دیگری مطابق شکل ۲۰ نمایش داده می‌شود، در این کادر محاوره روی دکمه Restart Windows کلیک کنید تا سیستم مجدداً راه‌اندازی شود.

شکل ۱۷


Visual Studio 6.0 Enterprise Setup

شکل ۱۸


Visual Studio 6.0 Enterprise Setup
OK

شکل ۱۹

شکل ۲۰

۱۶ – پس از راه‌اندازی مجدد سیستم کادرمحاوره دیگری (شکل ۲۱) ظاهر می‌شود. اگر دیسک‌های CD مربوط به راهنمای MSDN مایکروسافت را در اختیار داشته باشید، می‌توانید به وسیله این بخش راهنمای مزبور را نصب کنید تا امکان استفاده از منوی Help ویژوال‌بیسیک فراهم شود. برای این کار کافی است دیسک‌های CD مربوطه را داخل درایو قرار داده و روی دکمه Next کلیک کنید.

شکل ۲۱

اما اگر دیسک‌های CD مربوطه را دراختیار نداشته باشید باید روی کادر علامت MSDN Install موجود در این کادر محاوره کلیک کنید تا از حالت انتخاب خارج شـود؛ سـپس روی دکمه Next کلیک کنید.

۱۷ – پـس از انجام مرحله ۱۶ (در صورت عدم نصب MSDN) یک کادر پیغام مطابق شـکل ۲۲ مبنی بر عدم امکان اسـتفاده از منوی Help در برنامه ویژوال‌بیسیک نشان داده خواهد شد. روی دکمه Yes کلیک کنید.

شکل ۲۲

۱۸ – در این مرحله کادر محاوره‌ای مطابق شکل ۲۳ برای نصب بعضی از سرویس‌ها نمایش داده می‌شود که به آن‌ها نیازی ندارید. مجدداً روی دکمه Next کلیک کنید.

شکل ۲۳

۱۹ – در این مرحله نیز در کادر محاوره شـکل ۲۴ روی دکمه Next کلیک کنید، زیرا نیازی به نصب سرویس‌های موجود در لیست این کادر محاوره نخواهید داشت.

۲۰ – اگر مایل هستید در مرحله بعد اطلاعات خود را در سایت اینترنتی مربوطه ثبت کنید (در صورت عدم رعایت حقوق Copy Right امکان استفاده از این بخش میسر نیست).

در پایان روی دکمه Finish کلیک کنید تا نصب نرم‌افزار خاتمه یابد سپس سیستم را restart نمایید.

مطابق شکل ۲۵ در کادر محاوره مربوطه، روی کادر علامت Register Now کلیک کنید تا از حالت انتخاب خارج شود، سپس روی دکمه Finish کلیک کنید.

۲۱ – در این مرحله عملیات نصب نرم‌افزار خاتمه یافته است و شما می‌توانید زبان برنامه‌نویسی ویژوال‌بیسیک را از منوی Start ویندوز اجرا کنید.

شکل ۲۴

شکل ۲۵

ضمیمه ۲

کد کلیدها برای رویدادهای صفحه کلید

کدهای مربوط به کلیدهای حرفی در رویدادهای صفحه کلید

توضیح	کد کلید	ثابت رشته‌ای
کلید A	65	vbKeyA
کلید B	66	vbKeyB
کلید C	67	vbKeyC
کلید D	68	vbKeyD
کلید E	69	vbKeyE
کلید F	70	vbKeyF
کلید G	71	vbKeyG
کلید H	72	vbKeyH
کلید I	73	vbKeyI
کلید J	74	vbKeyJ
کلید K	75	vbKeyK
کلید L	76	vbKeyL
کلید M	77	vbKeyM
کلید N	78	vbKeyN
کلید O	79	vbKeyO
کلید P	80	vbKeyP
کلید Q	81	vbKeyQ
کلید R	82	vbKeyR
کلید S	83	vbKeyS
کلید T	84	vbKeyT
کلید U	85	vbKeyU
کلید V	86	vbKeyV
کلید W	87	vbKeyW
کلید X	88	vbKeyX
کلید Y	89	vbKeyY
کلید Z	90	vbKeyZ
کلید SPACEBAR	32	vbKeySpace
کلید BACKSPACE	8	vbKeyBack
کلید TAB	9	vbKeyTab
کلید ENTER	13	vbKeyReturn
کلید ESC	27	vbKeyEscape

کدهای مربوط به کلیدهای رقمی در رویدادهای صفحه کلید

ثابت رشته‌ای	کد کلید	توضیح
vbKey0	48	کلید 0
vbKey1	49	کلید 1
vbKey2	50	کلید 2
vbKey3	51	کلید 3
vbKey4	52	کلید 4
vbKey5	53	کلید 5
vbKey6	54	کلید 6
vbKey7	55	کلید 7
vbKey8	56	کلید 8
vbKey9	57	کلید 9

کدهای مربوط به کلیدهای رقمی بخش عددی صفحه کلید در رویدادهای KeyDown و KeyUp

ثابت رشته‌ای	کد کلید	توضیح
vbKeyNumpad0	96	کلید 0
vbKeyNumpad1	97	کلید 1
vbKeyNumpad2	98	کلید 2
vbKeyNumpad3	99	کلید 3
vbKeyNumpad4	100	کلید 4
vbKeyNumpad5	101	کلید 5
vbKeyNumpad6	102	کلید 6
vbKeyNumpad7	103	کلید 7
vbKeyNumpad8	104	کلید 8
vbKeyNumpad9	105	کلید 9
vbKeyMultiply	106	کلید *
vbKeyAdd	107	کلید +
vbKeySubtract	109	کلید -
vbKeyDecimal	110	کلید 0
vbKeyDivide	111	کلید /

کدهای مربوط به کلیدهای تابعی در رویدادهای KeyDown و KeyUp

توضیح	کد کلید	ثابت رشته‌ای
کلیـد F1	112	vbKeyF1
کلیـد F2	113	vbKeyF2
کلیـد F3	114	vbKeyF3
کلیـد F4	115	vbKeyF4
کلیـد F5	116	vbKeyF5
کلیـد F6	117	vbKeyF6
کلیـد F7	118	vbKeyF7
کلیـد F8	119	vbKeyF8
کلیـد F9	120	vbKeyF9
کلیـد F10	121	vbKeyF10
کلیـد F11	122	vbKeyF11
کلیـد F12	123	vbKeyF12
کلیـد F13	124	vbKeyF13
کلیـد F14	125	vbKeyF14
کلیـد F15	126	vbKeyF15
کلیـد F16	127	vbKeyF16

کدهای مربوط به سایر کلیدها در رویدادهای KeyDown و KeyUp

ثابت رشته‌ای	کد کلید	توضیح
vbKeyShift	16	SHIFT key
vbKeyControl	17	CTRL key
vbKeyMenu	18	ALT key
vbKeyPause	19	PAUSE key
vbKeyCapital	20	CAPS LOCK key
vbKeyPageUp	33	PAGE UP key
vbKeyPageDown	34	PAGE DOWN key
vbKeyEnd	35	END key
vbKeyHome	36	HOME key
vbKeyLeft	37	LEFT ARROW key
vbKeyUp	38	UP ARROW key
vbKeyRight	39	RIGHT ARROW key
vbKeyDown	40	DOWN ARROW key
vbKeyInsert	45	INS key
vbKeyDelete	46	DEL key
vbKeyNumlock	144	NUM LOCK key

کدهای کاراکتر حرفی کوچک برای رویداد KeyPress

توضیح	کد کلید	ثابت رشته‌ای
کلید a	97	VbKeya
کلید b	98	VbKeyb
کلید c	99	VbKeyc
کلید d	100	vbKeyd
کلید e	101	vbKeye
کلید f	102	vbKeyf
کلید g	103	vbKeyg
کلید h	104	vbKeyh
کلید i	105	vbKeyi
کلید j	106	vbKeyj
کلید k	107	vbKeyk
کلید l	108	vbKeyl
کلید m	109	vbKeym
کلید n	110	vbKeyn
کلید o	111	vbKeyo
کلید p	112	vbKeyp
کلید q	113	vbKeyq
کلید r	114	vbKeyr
کلید s	115	vbKeys
کلید t	116	vbKeyt
کلید u	117	vbKeyu
کلید v	118	vbKeyv
کلید w	119	vbKeyw
کلید x	120	vbKeyx
کلید y	121	vbKeyy
کلید z	122	vbKeyz

کدهای سایر کاراکترها برای رویداد KeyPress

توضیح	کد کلید
کلید >	60
کلید <	62
کلید ,	44
کلید ?	63
کلید /	47
کلید \	92
کلید ¦	124
کلید :	58
کلید ;	59
کلید »	34
کلید >	39
کلید }	123
کلید {	125
کلید]	91
کلید [	93
کلید =	61
Underline	95
)	40
(	41
&	38
^	94
%	37
$	36
#	35
@	64
!	33
~	126
`	96

مقادیر مربوط به آرگومان Shift در رویدادهای KeyDown و KeyUp

توضیح	ثابت عددی	ثابت رشته‌ای
در صورتی‌که کلید Shift فشرده شود.	1	vbShiftMask
در صورتی‌که کلید Ctrl فشرده شود.	2	vbCtrlMask
در صورتی‌که کلید Alt فشرده شود.	4	vbAltMask
در صورتی‌که کلیدهای Shift+Ctrl فشرده شوند.	3	vbShiftMask+vbCtrlMask
در صورتی‌که کلیدهای Shift+Alt فشرده شوند.	5	vbShiftMask+vbAltMask
در صورتی‌که کلیدهای Ctrl+Alt فشرده شوند.	6	vbCtrlMask+vbAltMask
در صورتی‌که کلیدهای Shift+Ctrl+Alt فشرده شوند.	7	vbShiftMask+vbCtrlMask+ vbAltMask

ضمیمه ۳

کد و پیام‌های خطا در ویژوال‌بیسیک

Code	Message
3	Return without GoSub
5	Invalid procedure call
6	Overflow
7	Out of memory
9	Subscript out of range
10	This array is fixed or temporarily locked
11	Division by zero
13	Type mismatch
14	Out of string space
16	Expression too complex
17	Can't perform requested operation
18	User interrupt occurred
20	Resume without error
28	Out of stack space
35	Sub, Function, or Property not defined
47	Too many DLL application clients
48	Error in loading DLL
49	Bad DLL calling convention
51	Internal error
52	Bad file name or number
53	File not found
54	Bad file mode
55	File already open
57	Device I/O error
58	File already exists
59	Bad record length
61	Disk full
62	Input past end of file
63	Bad record number
67	Too many files

Code	Message
68	Device unavailable
70	Permission denied
71	Disk not ready
74	Can't rename with different drive
75	Path/File access error
76	Path not found
91	Object variable or With block variable not set
92	For loop not initialized
93	Invalid pattern string
94	Invalid use of Null
97	Can't call Friend procedure on an object that is not an instance of the defining class
98	A property or method call cannot include a reference to a private object, either as an argument or as a return value
298	System DLL could not be loaded
320	Can't use character device names in specified file names
321	Invalid file format
322	Can't create necessary temporary file
325	Invalid format in resource file
327	Data value named not found
328	Illegal parameter; can't write arrays
335	Could not access system registry
336	Component not correctly registered
337	Component not found
338	Component did not run correctly
360	Object already loaded
361	Can't load or unload this object
363	Control specified not found
364	Object was unloaded
365	Unable to unload within this context
368	The specified file is out of date. This program requires a later version
371	The specified object can't be used as an owner form for Show
380	Invalid property value
381	Invalid property-array index
382	Property Set can't be executed at run time
383	Property Set can't be used with a read-only property
385	Need property-array index
387	Property Set not permitted
393	Property Get can't be executed at run time
394	Property Get can't be executed on write-only property
400	Form already displayed; can't show modally
402	Code must close topmost modal form first
419	Permission to use object denied
422	Property not found
423	Property or method not found
424	Object required
425	Invalid object use
429	Component can't create object or return reference to this object

Code	Message
430	Class doesn't support Automation
432	File name or class name not found during Automation operation
438	Object doesn't support this property or method
440	Automation error
442	Connection to type library or object library for remote process has been lost
443	Automation object doesn't have a default value
445	Object doesn't support this action
446	Object doesn't support named arguments
447	Object doesn't support current locale setting
448	Named argument not found
449	Argument not optional or invalid property assignment
450	Wrong number of arguments or invalid property assignment
451	Object not a collection
452	Invalid ordinal
453	Specified not found
454	Code resource not found
455	Code resource lock error
457	This key is already associated with an element of this collection
458	Variable uses a type not supported in Visual Basic
459	This component doesn't support the set of events
460	Invalid Clipboard format
461	Method or data member not found
462	The remote server machine does not exist or is unavailable
463	Class not registered on local machine
480	Can't create AutoRedraw image
481	Invalid picture
482	Printer error
483	Printer driver does not support specified property
484	Problem getting printer information from the system. Make sure the printer is set up correctly
485	Invalid picture type
486	Can't print form image to this type of printer
520	Can't empty Clipboard
521	Can't open Clipboard
735	Can't save file to TEMP directory
744	Search text not found
746	Replacements too long
31001	Out of memory
31004	No object
31018	Class is not set
31027	Unable to activate object
31032	Unable to create embedded object
31036	Error saving to file
31037	Error loading from file

Invalid procedure call or argument (Error 5)

Some part of the call can't be completed. This error has the following causes and solutions:

An argument probably exceeds the range of permitted values. For example, the Sin function can only accept values within a certain rang e.
Positive arguments less than 2,147,483,648 are accepted, while 2,147,483,648 generates this error.
Check the ranges permitted for arguments.

This error can also occur if an attempt is made to call a procedure that isn't valid on the current platform.
For example, some procedures may only be valid for Microsoft Windows, or for the Macintosh, and so on.
Check platform-specific information about the procedure.

For additional information, select the item in question and press F1.

Overflow (Error6)

An overflow results when you try to make an assignment that exceeds the limitations of the target of the assignment.
This error has the following causes and solutions:

The result of an assignment, calculation, or data type conversion is too large to be represented within the range of values allowed for
that type of variable.
Assign the value to a variable of a type that can hold a larger range of values.

An assignment to a property exceeds the maximum value the property can accept.
Make sure your assignment fits the range for the property to which it is made.

You attempt to use a number in a calculation, and that number is coerced into an integer, but the result is larger than an integer. For example:

```
Dim x As Long
x = 2000 * 365   ' Error: Overflow
```

To work around this situation, type the number, like this:

```
Dim x As Long
x = CLng(2000) * 365
```

For additional information, select the item in question and press F1

Out of memory (Error 7)

More memory was required than is available, or a 64K segment boundary was encountered. This error has the following causes and solutions:

You have too many applications, documents, or source files open.
Close any unnecessary applications, documents, or source files that are open.

You have a module or procedure that's too large.
Break large modules or procedures into smaller ones. This doesn't save memory, but it can prevent hitting 64K segment boundaries.

You are running Microsoft Windows in standard mode.
Restart Microsoft Windows in enhanced mode.

You are running Microsoft Windows in enhanced mode, but have run out of virtual memory.
Increase virtual memory by freeing some disk space, or at least ensure that some space is available.
You have terminate-and-stay-resident programs running.
Eliminate terminate-and-stay-resident programs.
You have many device drivers loaded.
Eliminate unnecessary device drivers.

You have run out of space for Public variables.

Reduce the number of Public variables.

Subscript out of range (Error 9)

Elements of arrays and members of collections can only be accessed within their defined ranges. This error has the following causes and solutions:

You referenced a nonexistent array element.

The subscript may be larger or smaller than the range of possible subscripts,

or the array may not have dimensions assigned at this point in the application.

Check the declaration of the array to verify its upper and lower bounds.

Use the UBound and LBound functions to condition array accesses if you're working with arrays that are redimensioned.

If the index is specified as a variable, check the spelling of the variable name.

You declared an array but didn't specify the number of elements. For example, the following code causes this error:

```
Dim MyArray() As Integer
MyArray(8) = 234  ' Causes Error 9.
```

Visual Basic doesn't implicitly dimension unspecified array ranges as 0 - 10. Instead,

you must use Dim or ReDim to specify explicitly the number of elements in an array.

You referenced a nonexistent collection member.

Try using the For Each...Next construct instead of specifying index elements.

You used a shorthand form of subscript that implicitly specified an invalid element.

For example, when you use the ! operator with a collection, the ! implicitly specifies a key. For example, object!keyname.

value is equivalent to object.item(keyname).value. In this case, an error is generated if keyname represents an invalid key in the collection.

To fix the error, use a valid key name or index for the collection.

Division by zero (Error 11)

Division by zero isn't possible. This error has the following cause and solution:

The value of an expression being used as a divisor is zero.
Check the spelling of variables in the expression. A misspelled variable name can implicitly create a numeric variable that is initialized to zero.
Check previous operations on variables in the expression, especially those passed into the procedure as arguments from other procedures.

Type mismatch (Error 13)

Visual Basic is able to convert and coerce many values to accomplish data type assignments that weren't possible in earlier versions. However,
this error can still occur and has the following causes and solutions:

The variable or property isn't of the correct type. For example,
a variable that requires an integer value can't accept a string value unless the whole string can be recognized as an integer.
Try to make assignments only between compatible data types. For example, an Integer can always be assigned to a Long,
a Single can always be assigned to a Double, and any type (except a user-defined type) can be assigned to a Variant.
An object was passed to a procedure that is expecting a single property or value.
Pass the appropriate single property or call a method appropriate to the object.
A module or project name was used where an expression was expected, for example:
Debug.Print MyModule

Specify an expression that can be displayed.

You attempted to mix traditional Basic error handling with Variant values having the Error

subtype (10, vbError), for example:
Error CVErr(n)

To regenerate an error, you must map it to an intrinsic Visual Basic or a user-defined error, and then generate that error.

A CVErr value can' be converted to Date. For example:
MyVar = CDate(CVErr(9))

Use a Select Case statement or some similar construct to map the return of CVErr to such a value.

At run time, this error typically indicates that a Variant used in an expression has an incorrect subtype,
or a Variant containing an array appears in a Print # statement.
To print arrays, create a loop that displays each element individually.

Out of string space (Error 14)

Visual Basic permits you to use very large strings. However, the requirements of other programs and
the way you manipulate your strings may cause this error. This error has the following causes and solutions:

Expressions requiring that temporary strings be created for evaluation may cause this error. For example,
the following code causes an Out of string space error on some operating systems:

```
MyString = "Hello"
For Count = 1 To 100
   MyString = MyString & MyString
Next Count
```

Assign the string to a variable of another name.

Your system may have run out of memory, which prevented a string from being allocated.
Remove any unnecessary applications from memory to create more space.

Sub, Function, or Property not defined (Error 35)

A Sub, Function, or Property procedure must be defined to be called. This error has the following causes and solutions:

You misspelled the name of your procedure.
Check the spelling and correct it.
You tried to call a procedure from another project without explicitly adding a reference to that project in the References dialog box.
To add a reference

Display the References dialog box.

Find the name of the project containing the procedure you want to call. If the project name doesn>t appear in the References dialog box,
click the Browse button to search for it.

Click the check box to the left of the project name.

Click OK.
The specified procedure isn't visible to the calling procedure.
Procedures declared Private in one module can't be called from procedures outside the module.
If Option Private Module is in effect,
procedures in the module aren't available to other projects. Search to locate the procedure.

You declared a Windows dynamic-link library (DLL) routine, but the routine isn't in the specified library or code resource.

Check the ordinal (if you used one) or the name of the routine. Make sure your version of the

DLL is the correct one.
The routine may only exist in later versions of the DLL.
If the directory containing the wrong version precedes the directory containing the correct one in your path, the wrong DLL is accessed.
You gave the right DLL name, but it isn>t the version that contains the specified function.

File not found (Error 53)

The file was not found where specified. This error has the following causes and solutions:

A statement, for example, Kill, Name, or Open, refers to a file that doesn>t exist.
Check the spelling of the file name and the path specification.
An attempt has been made to call a procedure in a dynamic-link library (DLL) or Macintosh code resource,
but the library specified in the Lib clause of the Declare statement can't be found.
Check the spelling of the file name and the path specification.

In the development environment, this error occurs if you attempt to open a project or load a text file that doesn't exist.
Check the spelling of the project name or file name and the path specification.

File already exists (Error 58)

This error has the following causes and solutions:

This error occurs at run time when the new file name, for example, one specified in a Name statement, is identical to a file name that already exists.
Specify a new file name in the Name statement or delete the old file before specifying it in a Name statement.

You used the Save As command to save a currently loaded project, but the project name already exists.

Use a different project name if you don't want to replace the other project.

Disk full (Error 61)

This error has the following causes and solutions:

There isn't enough room on the disk for the completion of a Print #, Write #, or Close operation.
Move some files to another disk or delete some files.
There isn't enough room on the disk to create required files.
Move some files to another disk or delete some files.

Disk not ready (Error 71)

:This error has the following causes and solutions

There is no disk in the specified drive.
.Put a disk in the drive and retry the operation

The drive door of the specified drive is open.
Close the drive door and retry the operation.

For loop not initialized (Error 92)

For loop counters must be initialized. This error has the following cause and solution:

You jumped into the middle of a For...Next loop.
Remove the jump into the loop. Placing labels inside a For...Next loop isn't recommended.

Invalid pattern string (Error 93)

The pattern string specified in the Like operation of a search is invalid. This error has the

following cause and solution:

A common example of an invalid character list expression is [a-b , where the right bracket is missing.
Review the valid characters for list expressions.

Invalid use of Null (Error 94)

Null is a Variant subtype used to indicate that a data item contains no valid data. This error has the following cause and solution:

You are trying to obtain the value of a Variant variable or an expression that is Null. For example:

```
MyVar = Null
For Count = 1 To MyVar
. . .
Next Count
```

Make sure the variable contains a valid value.

The specified object can't be used as an owner form for Show() (Error 371)

You must use an appropriate object with the Show method.

Invalid property value (Error 380)

Most properties only accept values of a certain type, within a certain range. This error has the following cause and solution:
An inappropriate value has been assigned to a property.
See the property's Help topic to determine what types and range of values are appropriate for the property.

Property Set can't be executed at run time (Error 382)

It may not be possible to obtain a reference to a property at run time.This error has the following cause and solution:

You tried to get a reference to a property that's either read-only or write-only at run time.
Since you can use a reference for both reading and writing, the property must provide run-time support for both operations for a reference to be obtained at run time.

Property Set can't be used with a read-only property (Error 383)

It may not be possible to obtain a reference to a property at run time. This error has the following cause and solution:

You tried to get a reference to a property that's read-only at run time.
Since you can use a reference for both reading and writing, the property must provide run-time support for both operations for a reference to be obtained at run time. You can only use a Property Get with this property.

Property not found (Error 422)

Not all objects support the same set of properties. This error has the following cause and solution:

This object doesn't support the specified property.
Check the spelling of the property name. Also, you may be trying to access something like a «text» property when the object actually supports a «caption» or some similarly named property. Check the object's documentation.

Property or method not found (Error 423)

The spelling of an object name must exactly match the definition in its object library. This error

You may have misspelled the name of the object. To see what properties and methods are defined for an object, display the Object Browser. Select the appropriate object library to view a list of available properties and methods.

Named argument not found (Error 448)

A named argument may not be used in a procedure invocation unless it appears in the procedure

definition. This error has the following cause and solution:

You specified a named argument, but the procedure was not defined to accept an argument by that name.
Check the spelling of the argument name.

Code resource not found (Error 454)

This error can only occur on the Macintosh. This error has the following cause and solution:

A call was made to a procedure in a code resource, but the code resource could not be found. Check to be sure the resource is available and properly referenced.

For additional information, select the item in question and press F1.

Out of memory (Error 31001)

Your system could not allocate or access enough memory or disk space for the specified operation.

فهرست منابع

۱- راهنمای MSDN

2- Microsoft Visual Basic 6 Programmer’s Guide (Microsoft press)