به نام خدا


# زبان برنامه‌نویسی
# Visual Basic 6.0

## (جلد دوم)

**ویژه دانشجویان مقاطع مختلف تحصیلی، دانش‌آموزان فنی‌وحرفه‌ای،**

**کاردانش و هنرستان‌ها**

**مؤلف**

**مهندس منصور ولی‌نژاد**


دارنده گواهینامه ISO 9001/2000
در زمینه نشر کتاب و طراحی جلد

## مقدمه ناشر

حمد و سپاس ایزد منان را که با الطاف بیکران خود این توفیق را به ما ارزانی داشت تا بتوانیم در راه ارتقای دانش عمومی و فرهنگ این مرز و بوم در زمینه چاپ و نشر کتب علمی دانشگاهی، علوم پایه و به ویژه علوم کامپیوتر و انفورماتیک گام‌هایی هر چند کوچک برداشته و در انجام رسالتی که بر عهده داریم مؤثر واقع شویم. گستردگی علوم و توسعه روزافزون آن، شرایطی را به وجود آورده که هر روز شاهد تحولات اساسی چشمگیری در سطح جهان هستیم. این گسترش و توسعه نیاز به منابع مختلف از جمله کتاب را به عنوان قدیمی‌ترین و راحت‌ترین راه دستیابی به اطلاعات و اطلاع‌رسانی، بیش از پیش روشن می‌کند. در این راستا، واحد انتشارات مؤسسه فرهنگی هنری دیباگران تهران با همکاری جمعی از اساتید، مؤلفان، مترجمان، متخصصان، پژوهشگران، محققان و نیز پرسنل ورزیده و ماهر در زمینه امور نشر در صدد هستند تا با تلاش‌های مستمر خود برای رفع کمبودها و نیازهای موجود، منابعی پربار، معتبر و با کیفیت مناسب در اختیار علاقه‌مندان قرار دهند.

کتابی که در دست دارید با همت **"مهندس منصور ولی‌نژاد"** و تلاش جمعی از همکاران انتشارات میسر گشته که شایسته است از یکایک این گرامیان تشکر و قدردانی کنیم.

**ویراستار :** خانم‌ها شیوا غمگسار و راحله عرفی

**ویرایش کامپیوتری و صفحه‌آرایی :** خانم معصومه باقری

**طراحی روی جلد :** خانم شیما صدرا

**امور چاپ و نشر :** آقای حیدر شفیعی

**ناظر چاپ :** آقای کریم براغ

در خاتمه از خوانندگان عزیز و دانش پژوهان گرامی خواهشمندیم ما را با ارایه پیشنهادها و انتقادهای خود در بهبود کمی و کیفی کارهای انجام شده راهنمایی نمایند تا بتوانیم در آینده کتاب‌هایی با کیفیت بهتر تقدیم حضورشان کنیم.

**مدیر انتشارات**

**مؤسسه فرهنگی هنری دیباگران تهران**

## مقدمه مؤلف

**تقدیم به اساتید، پدر ، مادر و همسر گرامی‌ام که**
**تقسیم داشته‌هایم را با دیگران از آن‌ها آموخته‌ام.**

بیش از نیم قرن از ظهور اولین کامپیوتر می‌گذرد، از آن زمان تاکنون یکی از مباحث اصلی مورد نظر متخصصان کامپیوتر، چگونگی استفاده از سخت افزار به وسیله برنامه‌های کامپیوتری(نرم‌افزار) بوده است چرا که بدون به کارگیری نرم‌افزار عملاً امکان استفاده از کامپیوتر میسر نیست؛ در نتیجه، اولین زبان برنامه نویسی با نام زبان ماشین در اولین کامپیوترها مورد استفاده قرار گرفت. با گذشت زمان و طراحی و تولید سیستم‌های جدید و پیشرفته‌تر، نیاز بیشتری برای به کارگیری زبان‌های برنامه نویسی که بتواند این امر را سریع‌تر، راحت‌تر و با امکانات کافی انجام دهد، احساس شد و سبب ارایه زبان‌های برنامه نویسی سطح بالا شد. با ظهور سیستم عامل ویندوز نسل جدیدی از زبان‌های برنامه نویسی به نام VISUAL پا به عرصه گذاشت، این زبان‌های برنامه نویسی علاوه بر امکانات زبان‌های قبلی، شرایط تولید نرم‌افزار مطابق با استانداردهای شرکت مایکروسافت را فراهم می‌کنند. در این راستا شرکت مایکروسافت با ارتقا و دگرگونی یکی از زبان‌های برنامه نویسی به نام بیسیک توانست یک زبان برنامه نویسی قدرتمند و با قابلیت بالا برای برنامه نویسی در محیط ویندوز ارایه کند.

کتاب حاضر با توجه به نیاز جامعه دانش‌آموزی و دانشجویی و با هدف آموزش آسان و کامل مفاهیم برنامه‌نویسی به مخاطب تألیف شده است. در تمامی فصول تمرینات به صورت کاملاً مرحله به مرحله و با جزییات کافی گنجانده شده است و فراگیر به راحتی می‌تواند مطالب را بیاموزد، به علاوه در هر فصل با ارایه خلاصه مطالب فصل و آزمون‌های چهارگزینه‌ای مطالبی را که آموخته است، ارزیابی کند. کتاب حاضر جلد دوم از مجموعه‌ای دو جلدی است. این مجموعه به گونه‌ای گردآوری شده است که علاوه بر فراگیران مبتدی، نیازهای دانشجویان کامپیوتر و برنامه‌نویسان را نیز برآورده کند.

در این‌جا از تمامی دوستان و همکاران در مؤسسه فرهنگی هنری دیباگران تهران، مدیر عامل محترم مؤسسه فناوران اطلاعات تهران جناب آقای دکتر سعید سعادت، مدیر انتشارات فرهنگی هنری دیباگران تهران سرکار خانم خدیجه سعادت و راهنمایی‌های جناب آقای مهندس الله‌وردی که اینجانب را در این امر یاری کرده‌اند، کمال تشکر و قدردانی را دارم.

در پایان از تمامی اساتید، متخصصان، دانشجویان، دانش‌آموزان و همکاران گرامی تقاضا دارم که با پیشنهادات و انتقادات خود اینجانب را در جهت رفع معایب و بهبود مطالب کتاب یاری کنند. به‌علاوه در صورت تمایل به دریافت غلط‌نامه جلد اول و راهنمایی‌های لازم می‌توانید، با آدرس‌های پست الکترونیکی زیر مکاتبه نمایید:

VBBook6@yahoo.com VBBook6@Hotmail.com

**منصور ولی‌نژاد**

# فهرست مطالب


مقدمه ناشر ..... ۱۱
مقدمه مؤلف ..... ۱۲

## فصل دهم : توانایی استفاده از انواع آرایه در ویژوال بیسیک

پیش آزمون ..... ۱۵
مقدمه ..... ۱۷
۱-۱۰ تعریف انواع آرایه در ویژوال بیسیک ..... ۱۸
۱-۱-۱۰ آرایه با ابعاد ثابت ..... ۱۸
۲-۱-۱۰ آرایه با ابعاد متغیر (Dynamic Array) ..... ۲۴
۲-۱۰ نحوه ارسال آرایه‌ها به رویه‌ها ..... ۲۷
۳-۱۰ فراخوانی یک رویه با تعداد آرگومان‌های نامعین ..... ۲۹
۴-۱۰روش‌های مرتب سازی آرایه‌ها ..... ۳۰
۱-۴-۱۰ روش مرتب‌سازی حبابی (Bubble Sort) ..... ۳۱
۲-۴-۱۰ روش مرتب‌سازی Shell ..... ۳۲
۵-۱۰ روش‌های جستجوی داده‌ها در آرایه‌ها ..... ۳۵
۱-۵-۱۰روش جستجوی خطی (Linear Search) ..... ۳۵
۲-۵-۱۰روش جستجوی دودویی (Binary Search) ..... ۳۶
۶-۱۰ آرایه‌های چند بعدی ..... ۳۹
۷-۱۰ توابع UBound و LBound ..... ۴۷
۸-۱۰ تابع Filter ..... ۴۹
۹-۱۰ تابعSplit ..... ۵۲
۱۰-۱۰ تابع Join ..... ۵۶
خلاصه مطالب ..... ۵۸
آزمون پایانی ..... ۵۹
دستور کار آزمایشگاه ..... ۶۰
پاسخ پیش آزمون ..... ۶۱
پاسخ آزمون پایانی ..... ۶۱

## فصل یازدهم : توانایی استفاده از جلوه‌های‌گرافیکی و چاپ در ویژوال بیسیک


پیش آزمون ..... ۶۴
مقدمه ..... ۶۵
۱-۱۱ مفهوم سیستم مختصات در ویژوال بیسیک ..... ۶۵
۲-۱۱ تغییر سیستم مختصات ..... ۶۶
۳-۱۱ خواص و متدهای گرافیکی ..... ۷۴
۱-۳-۱۱ متد PSet ..... ۷۵
۲-۳-۱۱ متد Line ..... ۷۷
۳-۳-۱۱ متد Circle ..... ۷۹
۴-۳-۱۱ متد Point ..... ۸۰
۵-۳-۱۱ خواصCurrentX و CurrentY ..... ۸۱
۶-۳-۱۱ متد Cls ..... ۸۳
۷-۳-۱۱ متد Print ..... ۸۳
۸-۳-۱۱ متدهای TextHeight, TextWidth ..... ۸۵
۹-۳-۱۱ خاصیت AutoRedraw ..... ۸۷
۱۰-۳-۱۱ خاصیت DrawMode ..... ۸۹
۱۱-۳-۱۱ خاصیت DrawStyle ..... ۹۱
۱۲-۳-۱۱ خاصیت DrawWidth ..... ۹۲
۱۳-۳-۱۱ خاصیت FillStyle ..... ۹۴
۴-۱۱ تابع QBcolor ..... ۹۶
۵-۱۱ تابع RGB ..... ۹۷
۶-۱۱ شئ چاپگر PRINTER OBJECT ..... ۹۹
۱-۶-۱۱ متدهای چاپ ..... ۹۹
۲-۶-۱۱ خواص شئ چاپگر ..... ۱۰۰
خلاصه مطالب ..... ۱۰۴
آزمون پایانی ..... ۱۰۶
دستور کار آزمایشگاه ..... ۱۰۸
پاسخ پیش آزمون ..... ۱۰۸
پاسخ آزمون پایانی ..... ۱۰۸

## فصل دوازدهم : نحوه استفاده ازکنترل‌های ذاتی وActiveX ویژوال بیسیک


پیش آزمون ..... ۱۱۰
مقدمه ..... ۱۱۱
۱-۱۲ کنترل کادر لیست (ListBox) ..... ۱۱۱
۱-۱-۱۲ خواص کنترل کادر لیست (ListBox) ..... ۱۱۲
۲-۱-۱۲ متدهای کنترل ListBox ..... ۱۲۰
۳-۱-۱۲ رویدادهای کنترل ListBox ..... ۱۲۲
۲-۱۲ کنترل ComboBox ..... ۱۲۲
۱-۲-۱۲ خواص کنترل ComboBox ..... ۱۲۲
۲-۲-۱۲ متدهای کنترلComboBox ..... ۱۲۵
۳-۱۲ کنترل CommonDialog ..... ۱۲۸
۱-۳-۱۲ نحوه ایجاد کادرهای محاوره باز کردن و ذخیره سازی فایل‌ها ..... ۱۳۰
۲-۳-۱۲ نحوه ایجاد کادر محاوره قلم(Font) ..... ۱۳۶
۳-۳-۱۲ نحوه ایجاد کادر محاوره رنگ(Color) ..... ۱۴۰
۴-۳-۱۲ نحوه ایجاد کادر محاوره چاپ(Print) ..... ۱۴۲
۵-۳-۱۲ نحوه ایجاد کادر محاوره راهنما(Help) ..... ۱۴۶
۴-۱۲ کنترل DriveListBox ..... ۱۴۶
۱-۴-۱۲ خواص کنترل DriveList Box ..... ۱۴۶
۲-۴-۱۲ متدهای کنترل DriveListBOX ..... ۱۴۷
۳-۴-۱۲رویدادهای کنترل DriveListBox ..... ۱۴۸
۵-۱۲ کنترل DirListBox ..... ۱۴۹
۱-۵-۱۲ خواص کنترل DirListBox ..... ۱۵۰
۲-۵-۱۲ متدهای کنترل DirListBox ..... ۱۵۱
۳-۵-۱۲ رویدادهای کنترل DirListBox ..... ۱۵۱
۴-۵-۱۲ دستورات مدیریت پوشه‌ها ..... ۱۵۵
۶-۱۲ کنترل FileListBox ..... ۱۵۶
۱-۶-۱۲ خواص کنترل FileListBox ..... ۱۵۷
۲-۶-۱۲ متدهای کنترل FileListBox ..... ۱۵۸
۳-۶-۱۲ رویدادهای کنترل FileListBox ..... ۱۵۸

۱۲-۷ کنترل MonthView ..... ۱۵۸
۱۲-۷-۱ خواص کنترل MonthView ..... ۱۵۹
۱۲-۷-۲ متدهای کنترل MonthView ..... ۱۶۵
۱۲-۷-۳ رویدادهای کنترل MonthView ..... ۱۶۵
۱۲-۸ کنترل DTPicker ..... ۱۶۶
۱۲-۸-۱ خواص کنترل DTPicker ..... ۱۶۷
۱۲-۸-۲ متدهای کنترل DTPicker ..... ۱۶۹
۱۲-۸-۳ رویه‌های کنترل DTPicker ..... ۱۶۹
۱۲-۹ کنترل FlatScrollBar ..... ۱۶۹
۱۲-۹-۱ خواص کنترل‌های FlatScrollBar ..... ۱۷۰
۱۲-۹-۲ متدهای کنترل FlatScrollBar ..... ۱۷۲
۱۲-۹-۳رویدادهای کنترل FlatScrollBar ..... ۱۷۳
۱۲-۱۰ کنترل ImageList ..... ۱۷۴
۱۲-۱۰-۱ خواص کنترل ImageList ..... ۱۷۵
۱۲-۱۰-۲ متدهای کنترل ImageList ..... ۱۷۶
۱۲-۱۱ کنترل ImageCombo ..... ۱۷۶
۱۲-۱۱-۱ خواص کنترل ImageCombo ..... ۱۷۷
۱۲-۱۱-۲ متدهای کنترل ImageCombo ..... ۱۷۸
۱۲-۱۱-۳ رویدادهای کنترل ImageCombo ..... ۱۸۰
۱۲-۱۲ کنترل MaskedEdit ..... ۱۸۰
۱۲-۱۲-۱ خواص کنترل MaskedEdit ..... ۱۸۱
۱۲-۱۲-۲ متدهای کنترل MaskedEdit ..... ۱۸۳
۱۲-۱۲-۳ رویدادهای کنترل MaskedEdit ..... ۱۸۳
۱۲-۱۳ کنترل RichTextBox ..... ۱۸۴
۱۲-۱۳-۱ خواص کنترل RichTextBox ..... ۱۸۴
۱۲-۱۳-۲ متدهای کنترل RichTextBox ..... ۱۸۶
۱۲-۱۳-۳ رویدادهای کنترل RichTextBox ..... ۱۸۷
۱۲-۱۴ کنترل‌های HScrollBar وVScrollBar ..... ۱۸۷
۱۲-۱۴-۱ خواص کنترل‌های نوار پیمایش ..... ۱۸۷
۱۲-۱۴-۲ متدهای کنترل‌های نوار پیمایش ..... ۱۸۸

۱۲-۱۴-۳ رویدادهای کنترل‌های نوار پیمایش ..... ۱۸۸
۱۲-۱۵ کنترل Slider ..... ۱۸۹
۱۲-۱۵-۱ خواص کنترل Slider ..... ۱۸۹
۱۲-۱۵-۲ متدهای کنترل Slider ..... ۱۹۱
۱۲-۱۵-۳ رویدادهای کنترل Slider ..... ۱۹۲
۱۲-۱۶ کنترل UpDown ..... ۱۹۳
۱۲-۱۶-۱ خواص کنترل UpDown ..... ۱۹۴
۱۲-۱۶-۲ متدهای کنترل UpDown ..... ۱۹۶
۱۲-۱۶-۳ رویدادهای کنترل UpDown ..... ۱۹۶
۱۲-۱۷ رابط‌های گرافیکی چند سندی MDI ..... ۱۹۹
۱۲-۱۸ نحوه ایجاد انواع منو در ویژوال بیسیک ..... ۲۰۳
۱۲-۱۹ فرم‌های آماده (TemplateForms) ..... ۲۱۳
خلاصه مطالب ..... ۲۱۵
آزمون پایانی ..... ۲۱۶
دستور کار آزمایشگاه ..... ۲۱۷
پاسخ پیش آزمون ..... ۲۱۷
پاسخ آزمون پایانی ..... ۲۱۷

## فصل سیزدهم : نحوه استفاده از رویدادهای ماوس و صفحه کلید

پیش آزمون ..... ۲۲۰
مقدمه ..... ۲۲۱
۱۳-۱ رویدادهای ماوس ..... ۲۲۱
۱۳-۱-۱ رویداد MouseDown ..... ۲۲۱
۱۳-۱-۲ رویداد MouseUp ..... ۲۲۳
۱۳-۱-۳ رویداد MouseMove ..... ۲۲۵
۱۳-۱-۴ رویداد DragDrop ..... ۲۲۵
۱۳-۲ رویدادهای صفحه کلید ..... ۲۲۶
۱۳-۲-۱ رویداد KeyDown ..... ۲۲۶
۱۳-۲-۲ رویداد KeyPress ..... ۲۳۰
۱۳-۲-۳ رویداد KeyUp ..... ۲۳۰

۱۳-۳ خاصیت KeyPreview ..... ۲۳۲
خلاصه مطالب ..... ۲۳۳
آزمون پایانی ..... ۲۳۴
دستور کار آزمایشگاه ..... ۲۳۵
پاسخ پیش آزمون ..... ۲۳۵
پاسخ آزمون پایانی ..... ۲۳۵

## فصل چهاردهم : نحوه خطایابی و خطازدایی برنامه‌ها در ویژوال بیسیک

پیش آزمون ..... ۲۳۹
مقدمه ..... ۲۴۰
۱۴-۱ مدیریت خطاهای زمان اجرا به وسیله دستور On Error GoTo ..... ۲۴۰
۱۴-۲ نحوه خطایابی و خطازدایی پروژه‌ها در ویژوال بیسیک ..... ۲۴۵
۱۴-۲-۱ حالت توقف (Break Mode) ..... ۲۴۵
۱۴-۲-۲ پنجره اجرای فوری دستورات (Immediate Window) ..... ۲۴۸
۱۴-۲-۳ شئ Debug (Debug object) ..... ۲۵۰
۱۴-۲-۴ منوی Debug (Debug Menu) ..... ۲۵۲
خلاصه مطالب ..... ۲۸۴
آزمون پایانی ..... ۲۸۷
دستور کار آزمایشگاه ..... ۲۸۸
پاسخ پیش آزمون ..... ۲۸۹
پاسخ آزمون پایانی ..... ۲۸۹

## فصل پانزدهم : نحوه استفاده از امکانات ویژوال بیسیک در پردازش فایل‌ها، پوشه‌ها و درایوها

پیش آزمون ..... ۲۹۳
مقدمه ..... ۲۹۴
۱۵-۱ مفاهیم مدل شئ FSO ..... ۲۹۴
۱۵-۱-۱ شئ Drive ..... ۲۹۴
۱۵-۱-۲ شئ Folder ..... ۲۹۴
۱۵-۱-۳ شئ File ..... ۲۹۵

۱۵-۱-۴ شئ File System Object ..... ۲۹۵
۱۵-۱-۵ شئ Text Stream ..... ۲۹۵
۱۵-۲ نحوه پردازش فایل‌ها، پوشه‌ها و درایوها با مدل شئ FSO ..... ۲۹۵
۱۵-۲-۱ نحوه ایجاد یک متغیر از مدل شئ FSO ..... ۲۹۵
۱۵-۲-۲ نحوه اختصاص دادن متد به شئ ایجاد شده ..... ۲۹۶
۱۵-۲-۳ نحوه دسترسی به خواص و ویژگی‌های شئ ایجاد شده ..... ۲۹۷
۱۵-۲-۴ متدهای مدل شئ FSO ..... ۲۹۹
خلاصه مطالب ..... ۳۲۱
آزمون پایانی ..... ۳۲۳
دستور کار آزمایشگاه ..... ۳۲۴
پاسخ پیش آزمون ..... ۳۲۴
پاسخ آزمون پایانی ..... ۳۲۴

## فصل شانزدهم : توانایی استفاده از فایل‌ها با دسترسی تصادفی و خواندن و نوشتن داده‌ها در آن‌ها

پیش آزمون ..... ۳۲۷
مقدمه ..... ۳۲۸
۱۶-۱ ویژگی‌های فایل‌ها با دسترسی تصادفی ..... ۳۲۹
۱۶-۲ نحوه تعریف نوع داده رکورد ..... ۳۳۰
۱۶-۳ نحوه باز کردن فایل‌ها با روش دسترسی تصادفی ..... ۳۳۲
۱۶-۳-۱ نحوه نوشتن اطلاعات در یک فایل با روش دستیابی تصادفی ..... ۳۳۳
۱۶-۳-۲ نحوه خواندن اطلاعات از یک فایل با روش دستیابی تصادفی ..... ۳۳۶
خلاصه مطالب ..... ۳۴۳
آزمون پایانی ..... ۳۴۴
دستور کار آزمایشگاه ..... ۳۴۵
پاسخ پیش آزمون ..... ۳۴۵
پاسخ آزمون پایانی ..... ۳۴۵

## فصل هفدهم : برنامه نویسی شئ‌گرا در ویژوال بیسیک

پیش آزمون ..... ۳۴۸

مقدمه ..... ۳۴۹
۱-۱۷ کلاس (Class) و مفاهیم مربوط به آن ..... ۳۴۹
۲-۱۷ کپسوله کردن یا مخفی کردن اطلاعات (Encapsulation) ..... ۳۵۰
۳-۱۷ پلی مورفیسم یا چند شکلی (Polymorphism) ..... ۳۵۰
۴-۱۷ وراثت (Inheritance) ..... ۳۵۱
۵-۱۷ نحوه ایجاد یک Class ..... ۳۵۱
۱-۵-۱۷ ایجاد یک کلاس جدید با استفاده از ماژول کلاس ..... ۳۵۱
۲-۵-۱۷ ابزار Class Builder ..... ۳۶۳
۶-۱۷ ابزار مرورگر شئ Object Browser ..... ۳۷۶
خلاصه مطالب ..... ۳۸۲
آزمون پایانی ..... ۳۸۴
دستور کار آزمایشگاه ..... ۳۸۵
پاسخ پیش آزمون ..... ۳۸۶
پاسخ آزمون پایانی ..... ۳۸۶

## فصل هجدهم : توانایی ایجاد و مدیریت پایگاه داده

پیش آزمون ..... ۳۸۸
مقدمه ..... ۳۸۹
۱-۱۸ نحوه ایجاد یک فایل پایگاه داده (Database File) ..... ۳۹۰
۱-۱-۱۸ نحوه ایجاد یک جدول(Table) ..... ۳۹۳
۲-۱-۱۸ نحوه ورود، ویرایش و حذف داده‌ها در جدول ..... ۳۹۶
۲-۱۸ کنترل‌های Data Base ..... ۴۰۰
۱-۲-۱۸ کنترل Data ..... ۴۰۱
۳-۱۸ نحوه ایجاد پرس‌وجوها(QUERY) ..... ۴۰۴
خلاصه مطالب ..... ۴۰۶
آزمون پایانی ..... ۴۰۷
دستور کار آزمایشگاه ..... ۴۰۸
پاسخ پیش آزمون ..... ۴۰۸
پاسخ آزمون پایانی ..... ۴۰۸
**منبع.. ..... ۴۰۸**

فصل دهم

## هدف کلی

# توانایی استفاده از انواع آرایه‌ها در ویژوال بیسیک

| زمان (ساعت) | |
|---|---|
| نظری | عملی |
| ۳ | ۶ |

### هدفهای رفتاری ▼

**پس از مطالعه این فصل 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 مرتب کردن اعضای یک آرایه با روش‌های زیر را داشته باشد.
الف- مرتب‌سازی با روش حبابی Bubble Sort ب- مرتب‌سازی با روش Shell
۶- توانایی جستجوی اطلاعات را در آرایه‌ها با روش خطی و دودویی داشته باشد.
۷- توانایی ایجاد و استفاده از آرایه‌های دوبعدی را داشته باشد.

## هدف کلی

# توانایی استفاده از انواع آرایه‌ها در ویژوال بیسیک

| زمان (ساعت) | |
|---|---|
| نظری | عملی |
| ۳ | ۶ |

### هدفهای رفتاری ▼

**پس از مطالعه این فصل از فراگیر انتظار می‌رود که:**

۸- نحوه استفاده از توابع Ubound و LBound را بداند.

۹- نحوه استفاده از توابع Filter، Split و Join را بداند.

# پیش آزمون

**۱- در صورتی که تعداد و نوع دکمه‌های فرمان در تابع MsgBox تعیین نشود چه دکمه‌هایی به‌عنوان پیش فرض استفاده می‌شوند؟**

۱- OK ۲- Cancel ۳- OK و Cancel ۴- Yes و No

**۲- کدام تابع می‌تواند عددی بودن یک رشته را بررسی کند؟**

۱- Val ۲- Str

۳- Isnumeric ۴- گزینه‌های ۱ و ۳ صحیح هستند.

**۳- کدام تابع برای جستجوی یک رشته در رشته دیگر استفاده می‌شود ؟**

۱- Instr ۲- Mid

۳- Instrev ۴- گزینه‌های ۱ و ۳ صحیح هستند.

**۴- کدام تابع برای حذف فواصل خالی از هر دو طرف یک عبارت رشته‌ای مناسب است؟**

۱- LTrim ۲- RTrim ۳- Trim ۴- هیچ‌کدام

**۵- به‌وسیله کدام تابع می‌توان یک کاراکتر را به تعداد دفعات معینی تکرار کرد؟**

۱- Str ۲- String

۳- StrReverse ۴- UCase

**۶- کدام تابع برای محاسبه فاصله زمانی بین دو تاریخ مناسب است؟**

۱- Date ۲- DatePart

۳- DateValue ۴- DateDiff

**۷- کدام یک از گزینه‌ها برای نمایش یک عدد به‌صورت درصد به‌وسیله تابع Format مناسب است؟**

۱- Currency ۲- Percent ۳- Scientific ۴- Fixed

**۸- در صورتی که عنوان یک کادر ورود داده (InputBox) تعیین نشود، .....**

۱- کادر ورود داده بدون عنوان نمایش داده می‌شود.

۲- نام پروژه برای عنوان در نظر گرفته می‌شود.

۳- عبارت InputBox برای عنوان استفاده می‌شود.

۴- پیام خطا نمایش داده می‌شود.

**۹- کدام تابع می‌تواند نام یک ماه را به‌وسیله شماره ترتیب ماه‌ها در سال محاسبه کند؟**

۱- Month ۲- MonthName

۳- MonthPart ۴- DateValue

**۱۰- به‌وسیله کدام تابع می‌توانید رشته‌ها را با یکدیگر مقایسه کنید؟**

۱- Str ۲- String

۳- StrComp ۴- Mid

## مقدمه

تاکنون با روش‌های مختلف ذخیره‌سازی اطلاعات در حافظه اصلی کامپیوتر آشنا شده‌اید که استفاده از انواع متغیرها و خواص 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؟

همان‌طور که می‌دانید 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 یا بردار) 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 ازداده‌هارادراعضای یک آرایه ذخیره کنید.

استفاده از آرایه‌ها در برنامه‌های بزرگ اجتناب ناپذیر 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 در داخل این محدوده قرار می‌گیرند. ویژوال بیسیک 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.

**نکته**

مفهوم آرایه در زبان‌های برنامه‌نویسی مثل ماتریس در ریاضیات است. برای درک بهتر آرایه‌ها بهتر است مفاهیم مربوط به ماتریس‌های سطری، ستونی و ماتریس‌های دوبعدی را در ریاضیات مورد توجه قرار دهید.

**توجه :** به دلیل طولانی بودن بعضی از دستورات، قرار دادن آن‌ها در یک سطر امکان‌پذیر نبوده است و حتی المقدور با استفاده از کاراکتر خط زیر ( _ ) دستور در چند خط نوشته شده است. دقت کنید بعضی از دستوراتی که بدون کاراکتر خط زیر در چند خط نوشته شده‌اند، در زمان نوشتن دستورات در ویژوال بیسیک در یک خط قرار بگیرند تا سبب ایجاد خطا نشوند.

## ۱-۱۰ تعریف انواع آرایه در ویژوال بیسیک

در ویژوال بیسیک دو نوع آرایه وجود دارد: آرایه با ابعاد ثابت و آرایه با ابعاد متغیر (آرایه پویا Dynamic). ابتدا به نحوه تعریف آرایه‌ها با ابعاد ثابت می‌پردازیم:

### ۱-۱-۱۰ آرایه با ابعاد ثابت

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

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 ویژوال بیسیک به‌طور پیش فرض اولین اندیس آرایه‌ها از شماره صفر آغاز می‌شود بنابراین در مثال قبل با توجه به مقدار دامنه بالایی، اندیس‌های آرایه از صفر تا ۱۴ خواهند بود 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
```

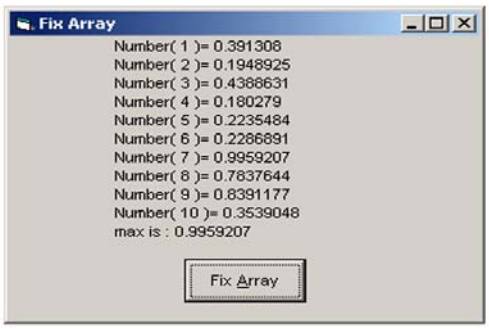
آرایه sums دارای اولین عضو با شماره اندیس ۱۰۰ و آخرین عضو با شماره اندیس ۱۲۰ خواهد بود به عبارت دیگر ۲۱ عضو خواهد داشت.

برای آن‌که با نحوه کار آرایه‌ها بهتر آشنا شوید به ذکر مثالی در این رابطه می‌پردازیم:

فرض کنید می‌خواهیم رویه‌ای بنویسیم که ده عدد را به‌صورت تصادفی ایجاد کرده و در آرایه‌ای ذخیره کند، سپس بزرگ‌ترین عدد را از بین ده عدد به دست آمده پیدا کرده و به همراه اعداد تصادفی تولید شده نشان دهد.

```
Sub  myrandom( )
        Dim  i  As  Integer,  intno(9)  As  Single
        Dim  max  As  Single
        Randomize
        max  =  0
        For  i  =  0  To  9
                intno(i)  =  Rnd
                If  max  <  intno(i)  Then  max  =  intno(i)
        Next  i
        For  i  =  0  To  9
                Print , "number("; i + 1; ")="; intno(i)
        Next  i
        Print  ,  "MAX  IS  :";  max
End  Sub
```

خروجی این رویه پس از فراخوانی مشابه شکل ۱-۱۰ خواهد بود.




**شکل ۱-۱۰**

همان‌طور که در رویه myrandom مشاهده می‌کنید 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 intno(0) را به‌وسیله تابع Rnd مقداردهی می‌کنیم و به‌وسیله یک If مقدار ماکزیمم را در هر تکرار محاسبه می‌کنیم همان‌طور که می‌بینید به‌وسیله حلقه For و اندیس i توانستیم به تک تک اعضای آرایه دسترسی پیدا کنیم و برای نمایش اعدادی که در آرایه intno ذخیره شده‌اند نیز از یک حلقه استفاده کرده‌ایم. البته شما می‌توانید به جای For از حلقه‌های دیگر نیز استفاده کنید اما با استفاده از حلقه For آسان‌تر خواهد بود.

**مثال :** رویه‌ای بنویسید که دو ماتریس ۵ عضوی یک بعدی از اعداد تصادفی را ایجاد کرده و حاصل ضرب آن‌ها را محاسبه کرده 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
```

همان‌طور که در رویه فوق ملاحظه می‌کنید از سه آرایه ۵ عضوی برای شبیه‌سازی ماتریس‌ها استفاده شده است و به‌وسیله اولین حلقه For اعضای دو ماتریس no1 و no2 مقداردهی‌شده‌اند سپس بااستفاده از حلقه For دوم حاصل‌ضرب اعضای متناظر دو ماتریس در عضو متناظر ماتریس حاصل‌ضرب یعنی no3 قرار می‌گیرد.

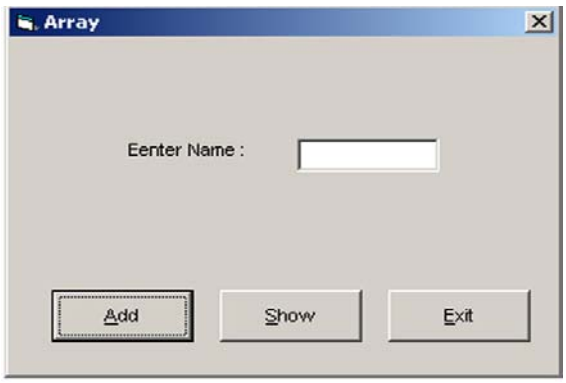
تاکنون با انواع آرایه‌های عددی آشنا شدید اما اگر لازم باشد که داده‌های رشته‌ای را به‌صورت آرایه ذخیره کنید چگونه باید عمل کرد؟

در واقع فرق زیادی بین آرایه‌های رشته‌ای و عددی وجود ندارد فقط در مورد آرایه‌های رشته‌ای طول هر عضو می‌تواند ثابت یا متغیر باشد.

به‌عنوان نمونه به ذکر مثالی در این رابطه اشاره می‌کنیم:

**مثال :** پروژه‌ای طراحی کنید که با استفاده از یک فرم و یک کنترل TextBox و سه دکمه فرمان بتواند اسامی ۱۰ نفر را دریافت کرده و در یک آرایه رشته‌ای ذخیره کند، این کار به‌وسیله کنترل TextBox و یکی از دکمه‌های فرمان (Add) انجام می‌شود.

به‌وسیله دکمه فرمان دوم (Show) نیز بتوان در هر لحظه اسامی وارد شده را مشاهده کرد و به‌وسیله دکمه فرمان سوم در هر لحظه امکان خروج از برنامه وجود داشته باشد. شکل ظاهری فرم و کنترل را می‌توانید در شکل ۲-۱۰ مشاهده کنید.




**شکل ۲-۱۰**

در این برنامه بخش اصلی برنامه به رویداد Click دکمه فرمان Add مربوط می‌شود با توجه به توضیحات ارایه شده و این مطلب‌که اگر کاربر نامی را در TextBox بنویسد و دکمه Add را بفشارد نام مورد نظر در یکی از اعضای آرایه ذخیره می‌شود.

البته قبل از هرچیز باید متغیرهای مورد نیاز را تعریف کنید. برای این کار ابتدا یک ماژول کد به پروژه اضافه کنید سپس در بخش تعاریف آن آرایه‌ای را با نام strname و با تعداد اعضای خواسته شده (۱۰عضو) به‌صورت عمومی تعریف کنید.

```
Public  strname (9) As String *20
```

بنابراین هر یک از اعضای این آرایه (strname) قادر به دریافت ۲۰ کاراکتر هستند.

اکنون رویداد Click کنترل دکمه فرمان Add را به شکل زیر کد نویسی کنید:

```
Private  Sub  cmdadd_Click( )
        Static  i  As  Integer
        If  i < 10  Then
                strname(i)  =  Text1.Text
                i  =  i  +  1
        Else
                MsgBox  "Array  is  full"
        End  If
End   Sub
```

همان‌طور که در رویداد فوق مشاهده می‌کنید از یک متغیر i به‌عنوان شمارنده اندیس آرایه و

به‌صورت Static استفاده شده است تا به‌وسیله مقدار این متغیر بتوان عضو بعدی در آرایه را با نامی‌که کاربر در TextBox می‌نویسد پر کرد. اگر این متغیر به‌صورت محلی تعریف شود و همواره اسامی‌در اولین عضو آرایه ذخیره شود 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. به هر صورت اگر (i < 10) باشد آن‌گاه محتویات خاصیت Text کنترل TextBox در یکی از اعضای آرایه ذخیره می‌شود و سپس مقدار i یک واحد افزایش می‌یابد تا در سری 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.

تا این جا بخش ورود داده‌ها کامل شده است، اکنون می‌خواهیم بخش داده‌های ورودی را طراحی کنیم. برای انجام این کار ابتدا یک فرم دیگر با نام frmdisplay به پروژه اضافه کنید و در روی آن یک دکمه فرمان با نام cmddone قرار دهید سپس دستورات زیر را در رویداد Click دکمه فرمان Show بنویسید.

```
Private  Sub  cmdshow_Click( )
        frmarray.Hide
        frmDisplay.Show
End  Sub
```

اکنون رویداد دکمه فرمان cmddone را در فرم frmdisplay به‌صورت زیر تنظیم کنید:

```
Private    Sub    cmddone_Click( )
               FrmDisplay.hide
               Frmarray.show
End    Sub
```

پس تنظیم رویداد دکمه فرمان cmddone در فرم frmdisplay دستورات زیر را در رویداد Activate همین فرم بنویسید.

```
Private  Sub  Form_Activate(  )
        Dim  j
        For  j  =  0  To  9
               Print  "NAME ( "; j ; " )= "; strname( j )
        Next
End  Sub
```

همان‌طور که در سه رویه رویداد فوق مشاهده کردید دو رویداد اول فقط نمایش یا عدم نمایش

فرم‌های مربوطه را به عهده دارند و رویه Activate فرم frmdisplay نیز باعث خواهد شد تا پس از فعال شدن فرم محتویات آرایه به وسیله یک حلقه نمایش داده شود. برنامه را اجرا کنید و تعداد ۱۰ نام را وارد کرده و پس از ورود هر نام دکمه Add را کلیک کنید و پس از پایان ورود داده‌ها به‌وسیله دکمه Show محتویات آرایه را مشاهده کنید و در پایان با کلیک دکمه Done به فرم اول باز گردید.

**تمرین :** برنامه قبل را به گونه‌ای 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 مشاهده کردید 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.

**نکته**

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 ۱۰ عضو را به‌وسیله اعداد تصادفی مقداردهی کنیم. (این مسأله را قبلاً حل کرده‌اید اما می‌خواهیم ببینیم که در صورت استفاده از فرمان Option Base 1 در بخش تعاریف ماژول فرم یا ماژول کد چه تغییراتی در دستورات رخ خواهد داد).

```
Sub  myrandom( )
        Dim  i  As  Integer,  intno(10)  As  Single
        Randomize
```

```
        For  i  =  1  To  10
               intno(  i  )  =  Rnd
        Next  i
End  Sub
```

همان‌طور که مشاهده می‌کنید در تعریف آرایه به جای عدد ۹ از عدد ۱۰ استفاده کرده‌ایم و مقدار پایانی در حلقه For را نیز از ۹ به ۱۰ تغییر داده‌ایم و مقدار شروع شمارنده را نیز از صفر به ۱ تبدیل کرده‌ایم.

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.

## ۲-۱-۱۰ آرایه با ابعاد متغیر (Dynamic Array)

گاهی اوقات ممکن است که از اندازه یک آرایه اطلاع نداشته باشید و بخواهید ابعاد آرایه را در زمان اجرا تنظیم کنید، در این صورت استفاده از آرایه با ابعاد ثابت، کارگشا نخواهد بود.

ویژوال بیسیک نوع دیگری از آرایه‌ها را با نام Dynamic Array در اختیار شما قرار می‌دهد. ابعاد آرایه‌ای از این نوع را می‌توانید با توجه به نیازتان مکرراً تغییر دهید.

آرایه‌های پویا (Dynamic) به راحتی در ویژوال بیسیک قابل استفاده هستند؛ این نوع از آرایه در مدیریت بهتر و بهینه از حافظه کامپیوتر، نقش به‌سزایی را ایفا می‌کنند به‌عنوان مثال شاید بخواهید از یک آرایه با ابعاد بزرگ در مدت زمان کوتاهی استفاده کنید و زمانی که از آرایه استفاده نمی‌کنید آن را از حافظه پاک کرده و حافظه سیستم را آزاد کنید.

برای تعریف یک آرایه دینامیک بهتر است مانند آرایه‌های ثابت عمل کرده اما از ذکر ابعاد آرایه خودداری کنید.

به‌عنوان مثال به فرمان زیر توجه کنید:

```
Dim dynnumber (  )   AS   integer
```

اما در صورت استفاده از آرایه در این مرحله پیام خطا نمایش داده خواهد شد.

برای قابل استفاده شدن آرایه‌های پویا (Dynamic) پس از تعریف آن باید ابعاد آن را با استفاده از فرمان ReDim تعیین کنید. به‌عنوان مثال پس از تعریف آرایه Dynnumber از نوع integer ابعاد آن را به شکل زیر تعیین کنید:

```
ReDim dynumber  (10)
```

اگر Option Base 0 باشد آرایه دارای یازده عضو و اگر Option Base 1 باشد آرایه دارای ۱۰ عضو (از ۱ تا۱۰) خواهد بود.

همان‌طور که گفته شد مزیت آرایه‌های پویا (Dynamic) در این است که می‌توان ابعاد آن را در زمان اجرا تغییر داد. مثلاً ابعاد آرایه dynnumber را می‌توان با فرمان ReDim dynumber (20) از یازده عضو به بیست و یک عضو تغییر داد.

**نکته**

در صورت استفاده از فرمان ReDim مقادیر موجود در تمام اعضای آرایه از بین خواهد رفت بنابراین در استفاده مجدد از دستور ReDim با دقت کافی اقدام کنید.

**نکته**

در صورتی که بخواهید مقادیر موجود در آرایه در زمان تغییر ابعاد آن حفظ شوند از کلمه کلیدی Preserve همراه با دستور ReDim استفاده 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)
For  i  =  1  To  5
        myarray  (i)
Next  i
ReDim  myarray(10)
For  i  =  1  To  10
        Print  myarray(i)
Next  i
```

همان‌طور که مشاهده می‌کنید با استفاده از تعریف آرایه پویا (Dynamic) آرایه myarray را با ۵ عضو تعریف کرده‌ایم سپس به‌وسیله یک حلقه For مقادیر ۱ تا ۵ را در آرایه قرار دادیم. پس از حلقه For اول مجدداً دستور ReDim را به کار گرفته‌ایم تا تعداد اعضای آرایه را دو برابر کنیم پس از تغییر ابعاد آرایه، حلقه For دوم مقادیر موجود در آرایه را نمایش می‌دهد اما همان‌طور که قبلاً گفتیم 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 فرمان ( 10 ) ReDim myarray از فرمان ReDim Preserve myarray (10) استفاده کنیم، 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 در صورت تمایل دستورات فوق را به‌وسیله رویداد Click یک دکمه فرمان آزمایش و نتیجه را در دو حالت بحث شده، بررسی کنید.

**نکته**

در صورت کاهش ابعاد یک آرایه به‌وسیله فرمان ReDim مقادیر مربوط به اعضای حذف شده از بین می‌روند.

**نکته**

به‌وسیله دستور ReDim نیز می‌توانید یک آرایه پویا Dynamic تعریف کنید شکل کلی این فرمان برای تعریف یک آرایه پویا به‌صورت زیر است :

ReDim نام آرایه (دامنه بالایی) As نوع داده

مثلاً فرمان زیر یک آرایه با ۲۰ عضو و از نوع رشته‌ای تعریف می‌کند.

```
ReDim fam (19) As String
```

**نکته**

در صورت استفاده از دستور ReDim برای تغییر ابعاد یک آرایه ثابت در هنگام اجرای برنامه، پیام خطای Array already dimensioned نمایش داده می‌شود.

تاکنون با نحوه تعریف و استفاده از آرایه‌های پویا (Dynamic) آشنا شدید، اکنون لازم است تا با نحوه حذف یک آرایه پویا از حافظه آشنا شوید.

برای انجام این کار می‌توانید از فرمان Erase استفاده کنید. شکل کلی فرمان Erase به‌صورت زیر است:

Erase نام آرایه

وقتی یک آرایه با این فرمان حذف می‌شود تمام مقادیر آن از بین خواهد رفت و شما می‌توانید مجدداً با استفاده از فرمان ReDim از آرایه استفاده کنید اگر پس از حذف یک آرایه پویا، بدون استفاده از فرمان ReDim سعی در دست‌یابی به اعضای آن داشته باشید پیام خطای Subscript out of range نمایش داده می‌شود.

در این‌جا لازم است به این نکته اشاره‌کنیم که به‌وسیله فرمان Erase می‌توانید مقادیر موجود در یک آرایه ثابت را حذف کنید، اما فضاهای مربوط به اعضای آرایه در حافظه از بین نخواهند رفت.

برای درک بهتر مسأله، مثال آخر را مجدداً مورد بررسی قرار می‌دهیم، اما این بار از دستور Erase و یک آرایه با ابعاد ثابت نیز در برنامه استفاده می‌کنیم. به دستورات زیر توجه کنید (با فرض این که Option Base 1 است):

```
Dim myarray1( ) As Integer, myarray2(5) As Integer
Dim i As Integer
```

```
ReDim  myarray1(5)
For  i  =  1  To  5
       myarray1  (i)
       myarray2  (i)
Next  i
Erase   myarray1,  myarray2
ReDim  myarray1(5)
For  i  =  1  To  10
       Print  myarray1(i)
       Print  myarray2(i)
Next  i
```

همان‌طور که مشاهده می‌کنید در این مجموعه دستورات از دو آرایه، پویا (Dynamic) و آرایه‌ای با ابعاد ثابت استفاده شده است. در اولین حلقه For، متغیرهای myarray 1 و myarray2 به ترتیب مقداردهی شده‌اند سپس به‌وسیله دستور Erase، آرایه myarray1 از حافظه کاملاً پاک شده اما آرایه myarray 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 mayrray 1 (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.

| نکته |
| --- |
| آرایه‌های پویا (Dynamic) را نیز می‌توانید به‌صورت Public، Private یا Static تعریف کنید. |

## ۲-۱۰ نحوه ارسال آرایه‌ها به رویه‌ها

تاکنون در این فصل با نحوه تعریف آرایه و چگونگی استفاده از انواع آن آشنا شده‌اید اما گاهی اوقات لازم است تا آرایه‌ها را به یک رویه فرعی و یا تابع ارسال کنید. در این‌جا با ذکر یک مثال با چگونگی انجام این کار آشنا می‌شوید.

**مثال :** می‌خواهیم تابعی بنویسیم که با دریافت یک آرایه عددی از نوع Integer با ۱۰ عضو، کوچک‌ترین عضو را پیدا کرده و باز گرداند.

ابتدا به تعریف تابع می‌پردازیم، چون آرگومان ورودی یک آرایه است بنابراین در هنگام تعریف آرگومان در تابع از دو پرانتز ( ) استفاده می‌کنیم.

```
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
```

همان‌طور که در تعریف تابع می‌بینید برای تعریف یک آرگومان از نوع آرایه فقط کافی است دو پرانتز در ابتدای نام آرایه ذکر کنید.

برای فراخوانی این تابع نیز می‌توانید فرمانی مشابه فرمان زیر بنویسید:

```
Print minelement (myarray ( ) )
```

در مورد رویه‌های فرعی نیز به همین صورت می‌توانید اقدام کنید، اما ذکر یک نکته در این جا لازم است که آرایه‌ها به‌طور پیش فرض به‌صورت ارسال با مرجع Call By Reference ارسال می‌شوند، البته بهتر از روش ارسال با مقدار Call By Value است زیرا در حالت ارسال با مرجع دیگر، فضای جداگانه دیگری به آرایه در رویه اختصاص داده نمی‌شود و وقتی شما از آرایه‌های بزرگ استفاده می‌کنید این مسأله از اهمیت به سزایی در مدیریت حافظه سیستم برخوردار خواهد بود و اگر بخواهید یک آرایه را به‌طور ارسال با مقدار، به یک رویه انتقال دهید با پیام خطا روبه‌رو می‌شوید. اما می‌توانید هر عضو آرایه را به‌صورت جداگانه به‌صورت ارسال با مقدار به رویه ارسال کنید. فرض کنید می‌خواهیم اعضای یک آرایه را به‌وسیله یک رویه فرعی به‌صورت جداگانه دریافت کرده و مجموع آن‌ها را محاسبه کنیم بنابراین ابتدا به تعریف رویه می‌پردازیم (با فرض این که Option Base 1 است).

چون قرار است که از روش ارسال با مقدار استفاده کنیم بنابراین در تعریف آرگومان x در تعریف رویه از کلمه کلیدی ByVal استفاده می‌کنیم پس:

```
Sub  mysum(ByVal  x  As  Single)

        Sum  =  Sum  +  x

End  Sub
```

**نکته**

توجه داشته باشید که متغیر sum از نوع عمومی‌در سطح ماژول است.

اکنون به فراخوانی این رویه می‌پردازیم چون اعضای آرایه به‌طور جداگانه و به‌صورت ارسال با مقدار به رویه ارسال می‌شوند بنابراین فراخوانی رویه می‌تواند در یک حلقه For انجام شود:

```
For  i  =  1  To  5

        Call  mysum(a(i))
```

```
Next  i

Print  Sum
```

که البته از این شکل نیز می‌توانید استفاده کنید.

```
Call mysum ( (a (i) ))
```

در صورتی که در زمان فراخوانی رویه از دو پرانتز در اطراف آرگومان استفاده کنید می‌توانید از ذکر کلمه کلیدی ByVal قبل از نام آرگومان در تعریف رویه صرف‌نظر کنید. به‌وسیله حلقه For، ۵ بار رویه فراخوانی می‌شود تا مقادیر آرگومان‌ها در متغیر عمومیsum جمع شود و با پایان حلقه مقدار sum نمایش داده خواهد شد.

**نکته**

در صورت تمایل می‌توانید رویه را در ماژول فرم و حلقه For را در رویه رویداد Click یک دکمه فرمان قرار دهید و برنامه را تست‌کنید؛ البته قبل از فراخوانی، آرایه را تعریف و مقداردهی کنید.

## ۳-۱۰ فراخوانی یک رویه با تعداد آرگومان‌های نامعین

یکی دیگر از ویژگی‌های آرایه‌ها، فراخوانی رویه‌ها با تعداد آرگومان‌های نامشخص است.

معمولاً تعداد آرگومان‌های یک رویه در فراخوانی آن، با توجه به تعریف رویه تعیین می‌شود اما گاهی اوقات ممکن است از تعداد آرگومان‌های ثابتی در فراخوانی یک رویه استفاده نشود.

در چنین مواردی می‌توانید با استفاده از کلمه کلیدی ParamArray قبل از نام آرگومان این عمل را انجام دهید البته آرگومان باید آرایه‌ای از نوع Variant باشد برای روشن شدن بهتر موضوع به رویه‌های زیر توجه کنید:

```
Function sum(ParamArray intnums( ) As Variant) As Variant
        Dim  i  As  Integer,  intsum  As  Variant
        For  i  =  LBound(intnums)  To  UBound(intnums)
                Intsum  =  intnums(i)  +  intsum
        Next  i
        sum  =  intsum
```

```
End  Function

Private  Sub  cmdsum_Click()

        Print  sum(5,  2,  3,  6)

        Print  sum(5,  2,  1)

End  Sub
```

همان‌طور که در تابع sum مشاهده می‌کنید آرگومان intnums به‌وسیله کلمه کلیدی ParamArray در تعریف رویه، به‌صورت یک آرایه از نوع Variant معرفی شده است در نتیجه در فراخوانی رویه sum از داخل رویداد Click دکمه فرمان cmdsum، از هر تعداد آرگومانی می‌توان استفاده کرد. مقادیر ارسالی به‌صورت اعضای یک آرایه به آرایه intnums انتقال داده می‌شوند. در رویه sum با استفاده از توابع LBound و UBound مقدار دامنه پایینی و بالایی آرایه intnums به دست می‌آید تا حلقه for به‌وسیله اندیس i به تک تک مقادیر ارسالی به تابع دسترسی پیدا کرده و آن‌ها را با یکدیگر جمع کند و به‌عنوان مقدار بازگشتی به محل فراخوانی باز گرداند؛ بنابراین در فراخوانی اول مقدار ۱۶ و در فراخوانی دوم مقدار ۸ به دست آمده و نمایش داده می‌شود.

**نکته**

رویه‌های فوق را می‌توانید در قالب یک پروژه طراحی و اجرا کنید.

**نکته**

توابع LBound و UBound در همین فصل توضیح داده می‌شوند.

**نکته**

در صورت استفاده از انواع دیگر داده برای آرگومانی که با کلمه کلیدی ParamArray معرفی می‌شود پیام خطایی نمایش داده خواهد شد.

## ۱۰-٤ روش‌های مرتب‌سازی آرایه‌ها

یکی از ویژگی‌های دیگر آرایه‌ها، پدیده مرتب‌سازی یا Sorting است. در برنامه‌نویسی واقعی معمولاً لازم است تا اطلاعات دریافت شده و یا نتیجه محاسبات را به‌صورت مرتب شده در اختیار کاربران قرار دهید.

تاکنون روش‌های متعددی جهت 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 Quick 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 به توضیح دو روش مرتب‌سازی حبابی و Shell خواهیم پرداخت البته تمام روش‌هایی که از آن‌ها نام برده‌ایم می‌توانند داده‌ها را به‌صورت صعودی (یعنی از کوچک به بزرگ) یا نزولی (یعنی از بزرگ به کوچک) مرتب کنند.

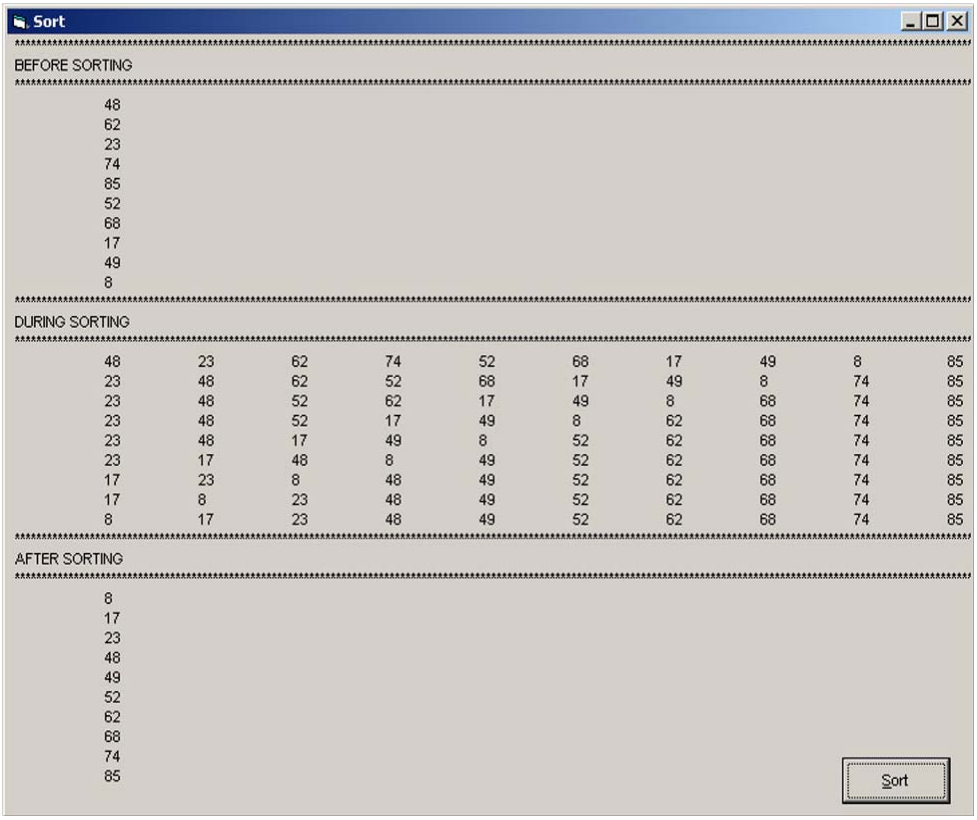
## ۱-۴-۱۰ روش مرتب‌سازی حبابی (Bubble Sort)

این روش ساده‌ترین و کندترین 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 جهت مرتب شدن داده‌ها مصرف می‌شود بنابراین توصیه می‌شود تا در صورت کوچک بودن ابعاد یک آرایه از این روش استفاده کنید. از مزایای این روش به سادگی آن می‌توان اشاره کرد. در زیر دستوراتی را می‌بینید که می‌تواند یک آرایه ۱۰ عضوی را به روش حبابی و به‌صورت صعودی مرتب کند.

```
For  i  =  0  To  9
        For  j  =  0  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
```

همان‌طور که مشاهده می‌کنید عمل مرتب‌سازی با دو حلقه که n-۱ بار تکرار می‌شوند (n تعداد اعضای آرایه است) انجام می‌شود. در واقع به‌وسیله یک If در داخل حلقه دوم مقدار هر عضو با عضو بعدی مقایسه می‌شود و در صورت بزرگ‌تر بودن عضوی که اندیس آن j است مقدار آن با مقدار عضوی

که اندیس آن j+1 است تعویض می‌شود. با ادامه روند عملیات، این کار آنقدر تکرار می‌شود تا آرایه مرتب شود؛ می‌توانید روند انجام عملیات را در شکل ۳-۱۰ مشاهده کنید: در ابتدا، مقادیر موجود در آرایه را قبل از مرتب شدن و در انتها، همان آرایه را به‌صورت مرتب شده 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 ۳-۱۰**

## ۳-۴-۲-۱۰ روش مرتب‌سازی Shell

این روش مرتب‌سازی نسبت به روش حبابی از عملکرد بهتری برخوردار است. در این روش از یک بازه برای مقایسه عناصر موجود در آرایه استفاده می‌شود تا هر مقدار، در جایگاه تقریبی خود قرار بگیرد. مثلاً اگر ابعاد آرایه ۲۰ باشد، ابتدا بازه ۱۰ یعنی نصف داده‌ها در نظر گرفته می‌شود و عنصر اول با یازدهم و عنصر دوم با دوازدهم و ... عنصر دهم با بیستم مقایسه شده و در صورت نیاز مقادیر آن‌ها

جابه‌جا خواهد شد. در مرحله بعد مقدار بازه دوباره نصف می‌شود و این‌بار عناصر اول با ششم و دوم با هفتم و الی آخر، این کار تا بازه با مقدار یک ادامه می‌یابد تا تمام عناصر به‌صورت صعودی یا نزولی مرتب شوند. در این‌جا به ذکر تابعی که می‌تواند یک آرایه را با روش Shell مرتب کند، می‌پردازیم:

```
Private Function shellsort(myarray() As Integer)
    Dim gap, i, temp, j
    gap = UBound(myarray) \ 2
    Do While (gap > 0)
        For i = (gap + 1) To UBound(myarray)
            j = i - gap
            Do While (j > 0)
                k = j + gap
                If (myarray(j) <= myarray(k)) Then
                    j = 0
                Else
                    temp = myarray(j)
                    myarray(j) = myarray(k)
                    myarray(k) = temp
                    j = j - gap
                End If
            Loop
        Next i
        gap = gap \ 2
    Loop
End Function
```

① ②

همان‌طور که در تابع فوق مشاهده می‌کنید از متغیر gap به‌عنوان فاصله و بازه مورد نیاز برای مقایسه اعضای آرایه و از متغیرهای j و k برای دسترسی به اعضای طرفین بازه gap استفاده می‌شود. در جدول ۱-۱۰ مقدار متغیرها و نحوه جابه‌جایی و مرتب شدن یک آرایه ۶ عضوی با روش Shell نمایش داده شده است.

| | | | | ① | ② | ③ | ④ | ⑤ | ⑥ |
|---|---|---|---|---|---|---|---|---|---|
| | | | | 78 | 68 | 21 | 10 | 39 | 2 |
| i | j | k | gap | | | | | | |
| 4 | 1 | 4 | 3 | ① 10 | > | 21 | ④ 78 | 39 | 2 |
| | -2 | | | | | | | | |
| 5 | 2 | 5 | | 10 | ② 39 | 21 | < | ⑤ 68 | 2 |
| | -1 | | | | | | | | |
| 6 | 3 | 6 | | 10 | 39 | ③ 2 | 78 | < 68 | ⑥ 21 |
| | 0 | | | | | | | | |
| | | | 1 | | | | | | |
| 2 | 1 | 2 | | ① 10 | < ② 39 | 2 | 78 | 68 | 21 |
| | 0 | | | | | | | | |
| 3 | 2 | 3 | | 10 | ② 2 | < ③ 39 | 78 | 68 | 21 |
| | 1 | | | | | | | | |
| | | 2 | | ① 2 | > ② 10 | 39 | 78 | 68 | 21 |
| | 0 | | | | | | | | |
| 4 | 3 | 4 | | 2 | 10 | ③ 39 | < ④ 78 | 68 | 21 |
| | 2 | | | | | | | | |
| | | 3 | | 2 | ② 10 | < ③ 39 | 78 | 68 | 21 |
| | 0 | | | | | | | | |
| 5 | 4 | 5 | | 2 | 10 | 39 | ④ 68 | > ⑤ 78 | 21 |
| | 3 | | | | | | | | |
| | | 4 | | 2 | 10 | ③ 39 | < ④ 68 | 78 | 21 |
| | 0 | | | | | | | | |
| 6 | 5 | 6 | | 2 | 10 | 39 | 68 | ⑤ 21 | > ⑥ 78 |
| | 4 | | | | | | | | |
| | | 5 | | 2 | 10 | 39 | ④ 21 | > ⑤ 68 | 78 |
| | 3 | | | | | | | | |
| | | 4 | | 2 | 10 | ③ 21 | > ④ 39 | 68 | 78 |
| | 2 | | | | | | | | |
| | | 3 | | 2 | ② 10 | < ③ 21 | 39 | 68 | 78 |
| | 0 | | | | | | | | |
| | | 0 | | | | | | | |

**توجه**

در جدول فوق هر خط توپر نشان دهنده یک‌بار اجرای حلقه Do While شماره ۱ و هر خط‌چین (- - -) نشان دهنده یک بار اجرای حلقه For و هر خط نقطه (ـ ـ) نشان دهنده یک‌بار اجرای حلقه Do While شماره ۲ می‌باشد.

# ۱۰-۵ روش‌های جستجوی داده‌ها در آرایه‌ها

تاکنون روش‌های مختلفی برای ورود، ذخیره‌سازی و مرتب‌سازی داده‌ها آموختید. اما گاهی اوقات لازم است تا داده‌ای را در 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 (Linear) و روش جستجوی دو دویی (Binary) را نام برد.

## ۱۰-۵-۱ 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
```

همان‌طور که در دستورات فوق مشاهده می‌کنید به‌وسیله حلقه for و یک فرمان if تک تک اعضای آرایه با مقدار mynumber مقایسه می‌شوند و در صورت پیدا شدن مقدار مشابه مقدار True برگشت داده می‌شود و پس از آن به‌وسیله Exit Function خروج از تابع اتفاق می‌افتد. اما اگر تمام اعضای آرایه بررسی شوند و مقدار برابر با mynumber پیدا نشود با بازگشت دادن مقدار False این موضوع روشن خواهد شد.

به‌عنوان مثالی دیگر فرض کنید می‌خواهیم یک نام را در یک آرایه رشته‌ای 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 قبلی است و فقط برای پیدا کردن نام مورد نظر (myname) در آرایه از تابع StrComp استفاده می‌شود.

## ۲-۵-۱۰ روش جستجوی دو دویی (Binary Search)

روش جستجوی خطی در آرایه‌های بزرگ بسیار کند و وقت‌گیر است، برای آن‌که زمان جستجو کاهش یابد از راه حل ساده‌ای می‌توان استفاده کرد. 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 و سپس عضو میانی آرایه با مقدار مورد جستجو، مقایسه می‌شوند، اگر مقدار مورد جستجو کوچک‌تر از عضو میانی باشد جستجو در قسمت بالایی عضو میانی صورت می‌گیرد و اگر مقدار مورد جستجو بزرگ‌تر از عضو میانی باشد جستجو در قسمت پایینی 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.

دستورات بعدی نحوه انجام این کار را نشان می‌دهند:

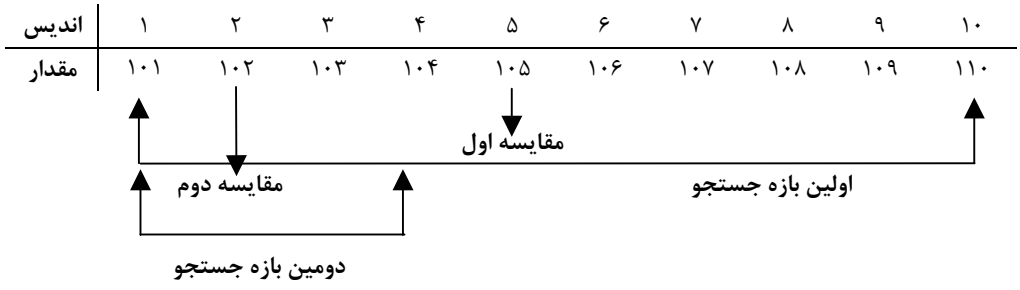
```
Function mysearch(myarray( ) As Integer, mynum As _
Integer) As Boolean

Dim i As Integer, intlow As Integer, 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
```

همان‌طور که مشاهده می‌کنید تابع پس از تعریف متغیرهای مورد نظر، مقدار اندیس بالایی و پایینی در آرایه را به‌وسیله توابع LBound , HBound در متغیرهای inthigh و intlow قرار می‌دهد.

سپس یک حلقه Do while عملیات جستجو را به شکل زیر انجام می‌دهد:

ابتدا مقدار اندیس میانی در آرایه محاسبه می‌شود برای این کار اندیس پایینی با بالایی جمع شده و مقدار صحیح تقسیم آن‌ها بر عدد ۲ توسط تابع Int به دست می‌آید.

فرض کنید که آرایه دارای ۱۰ عضو است و اندیس پایینی 1 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 mynumber 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 آرایه myarray (5) کوچک‌تر از مقدار mynumber 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 اندیس ۶ و۱۰ را مورد جستجو قرار می‌دهد؛ اما اگر شرط موجود در ElseIf غلط باشد بنابراین عنصر مورد جستجو در نیمه بالایی آرایه قرار دارد و با استفاده از بخشElse 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 به گونه‌ای 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 ۲۰ را در آرایه ۱۰ عضوی (مقدار اعضای آن از ۱۰۱ تا ۱۱۰ می‌باشند) جستجو کنیم. عملکرد تابع به‌صورت زیر نمایش داده می‌شود:




**شکل ۴-۱۰**

همان‌طور که در شکل ۴-۱۰ مشاهده می‌کنید با انجام دومین مقایسه مقدار intlow=1 و مقدار inthigh=2 خواهد بود و چون مقدار ۲۰ کوچک‌تر از ۱۰۲ است پس ۱-۲ = inthigh می‌شود.

در سومین مقایسه مقدار ۱ = intlow و مقدار ۱ = inthigh است و چون مقدار ۲۰ کوچک‌تر از ۱۰۲ است پس حاصل inthigh=1-1، صفر خواهد شد بنابراین در اجرای مجدد حلقه مقدار inthigh بزرگ‌تر یا مساوی intlow نخواهد بود (۱ > ۰) و اجرای حلقه خاتمه یافته و مقدار False بازگشت داده می‌شود.

همان‌طور که مشاهده کردید تعداد مقایسه‌هایی که در روش دودویی صورت می‌گیرد به مراتب کمتر از روش خطی است و در نتیجه سرعت پردازش در برنامه افزایش می‌یابد. توجه داشته باشید که با افزایش زیاد تعداد اعضا در آرایه این تفاوت بین دو روش جستجو به‌طور چشمگیری افزایش می‌یابد.

## ۶-۱۰ آرایه‌های چند بعدی

تاکنون با نحوه تعریف و چگونگی 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.

استفاده از آرایه‌های دو بعدی و یا بالاتر می‌تواند پاسخگوی نیازهای برنامه‌نویسی در این زمینه باشد، البته بحث در رابطه با آرایه‌های سه بعدی و بالاتر از حوصله این کتاب خارج است و در این‌جا به ذکر نحوه تعریف و استفاده از آرایه‌های دو بعدی می‌پردازیم.

آرایه‌های دو بعدی مانند ماتریس‌های 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 ماتریس $A_{۴\times۳}$ توجه کنید، این ماتریس دارای ۴ سطر و ۳ ستون است.

| شماره سطر \ شماره ستون | 0 | 1 | 2 |
|---|---|---|---|
| 0 | $a_{00}$ | $a_{01}$ | $a_{02}$ |
| 1 | $a_{10}$ | $a_{11}$ | $a_{12}$ |
| 2 | $a_{20}$ | $a_{21}$ | $a_{22}$ |
| 3 | $a_{30}$ | $a_{31}$ | $a_{32}$ |

همان‌طور که می‌بینید برای دسترسی به هر یک از عناصر آرایه a از دو اندیس استفاده می‌شود مثلاً عنصر $a_{۲۱}$ ، عضوی است که در سطر شماره ۲ و ستون شماره ۱ قرار دارد.

در آرایه نیز از همین شکل آدرس‌دهی برای دست‌یابی به اعضای یک آرایه دو بعدی استفاده می‌شود.

برای تعریف آرایه دو بعدی می‌توانید یکی از این روش‌ها را به کار ببرید:

[ Dim ¦ Public ¦ Private ¦ Static ] نام آرایه (دامنه بالایی ستون‌ها و دامنه بالایی سطرها)

[ Dim ¦ Public ¦ Private ¦ Static ] نام آرایه (دامنه بالایی ستون‌ها و دامنه بالایی سطرها) As نوع داده

[ Dim ¦ Public ¦ Private ¦ Static ] نام آرایه (دامنه بالایی ستون‌ها To دامنه پایینی ستون‌ها و دامنه‌بالایی سطرها To دامنه پایینی سطرها) As نوع داده

با توجه به مکان تعریف و کاربرد آرایه می‌توانید یکی از کلمات کلیدی موجود در [ ] را انتخاب کنید.

به‌عنوان مثال فرض کنید می‌خواهیم نمرات سه درس دانش‌آموزان یک کلاس ۵ نفره را به‌صورت یک ماتریس دو بعدی بنویسیم. اطلاعات را به‌صورت این جدول می‌توان نوشت :

| | شیمی | فیزیک | ریاضی | شماره ستون / شماره سطر |
|---|---|---|---|---|
| دانش‌آموز اول | ۱۶ | ۱۴ | ۱۷ | ۰ |
| دانش‌آموز دوم | ۸ | ۱۰ | ۱۲ | ۱ |
| دانش‌آموز سوم | ۱۵ | ۱۸ | ۲۰ | ۲ |
| دانش‌آموز چهارم | ۱۹ | ۱۳ | ۱۴ | ۳ |
| دانش‌آموز پنجم | ۱۱ | ۲۰ | ۱۷ | ۴ |
| | ۲ | ۱ | ۰ | |

جدول فوق را می‌توان به‌صورت ماتریس نوشت:

$$M_{۵\times۳}\begin{bmatrix} ۱۷ & ۱۴ & ۱۶ \\ ۱۲ & ۱۰ & ۱۸ \\ ۲۰ & ۱۸ & ۱۵ \\ ۱۴ & ۱۳ & ۱۹ \\ ۱۷ & ۲۰ & ۱۱ \end{bmatrix}$$

همان‌طور که مشاهده می‌کنید به‌وسیله ماتریس M با ۵ سطر (۰ تا ۴) و ۳ ستون (۰ تا ۲) توانستیم اطلاعات جدول را به‌صورت کامل اما خلاصه‌تر، نمایش دهیم، اکنون می‌خواهیم این ماتریس را به‌صورت یک آرایه دو بعدی تعریف کنیم بنابراین یکی از این موارد را می‌توان جهت انجام این‌کار استفاده نمود:

```
Dim student (4,2)   As   Single
Dim  student (1 To 5 , 1  To 3) As  Single
```

و دستور دوم را می‌توان به‌صورت زیر تعریف کرد:

```
Dim student (0  To  4,0  To 2)
```

**نکته**

توجه داشته باشید که قوانین دستور Option Base برای آرایه‌های دو بعدی نیز صادق است.

اما در این‌جا لازم است که چگونگی ذخیره‌سازی داده‌ها یا دست‌یابی به مقادیر موجود در یک آرایه دو بعدی را بیاموزید.

معمولاً برای دست‌یابی به اعضای یک آرایه دو بعدی می‌توانید از دو حلقه For تو در تو استفاده کنید البته می‌توانید از سایر حلقه‌ها نیز استفاده کنید.

مثلاً فرض کنید که می‌خواهیم مقادیر موجود در آرایه student را نمایش دهیم. باید این دستورات را در نظر بگیرید :

```
For  i  =  0  To  4
        For  j  =  0  To  2
                Print  student(i,  j)
```

```
        Next  j
Next  i
```

همان‌طور که مشاهده می‌کنید از دو حلقهFor استفاده شده است حلقهFor اول به‌وسیله شمارنده i شماره سطر و حلقه دوم شماره ستون را در آرایه کنترل می‌کنند، بنابراین با هر بار اجرای حلقه اول، حلقه دوم، سه بار تکرار می‌شود تا عناصر هر سطر را جهت نمایش آماده کند. مثلاً زمانی که i = ۰ است، حلقه دوم مقادیرJ را به ترتیب روی ۰، ۱و۲ تنظیم می‌کند درنتیجه اعضای student (0,0)، student (0,1) و student (0,2) مورد دست‌یابی قرار می‌گیرند. با خاتمه حلقه دوم، حلقه اول مقدار i را یک واحد افزایش می‌دهد و حلقه دوم مجدداً از j =۰ آغاز شده و بدین ترتیب عناصر سطر دوم (i=1) نیز در سه بار اجرای حلقه دوم در دسترس قرار می‌گیرند.

**مثال :** می‌خواهیم رویدادی بنویسیم که ماتریس (آرایه) دو بعدی A ۴×۴ را با اعداد تصادفی مقداردهی کرده سپس اعضای روی قطر اصلی آن را به صفر تبدیل کند (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
                If  i  =  j  Then  A(i,  j)  =  0
        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
```

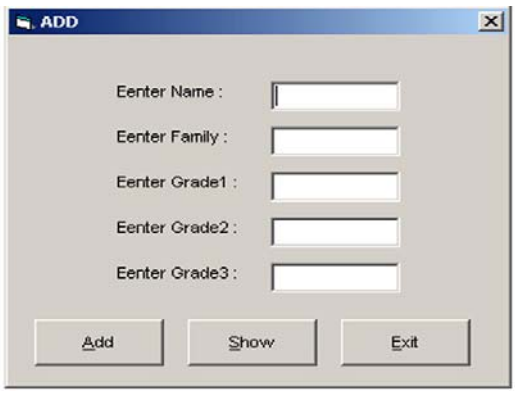
رویه فرعی فوق در صورت اجرا، خروجی مشابه شکل زیر خواهد داشت.

```
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 را مقداردهی کردیم و برای صفر کردن عناصر روی قطر اصلی مجدداً با استفاده از دو حلقه For تو در تو و یک فرمان If عناصری را که اندیس سطر و ستون برابر دارند، پیدا کرده و مقدارشان را به صفر تغییر داده‌ایم چون عناصر قطر اصلی دارای شماره سطر و ستون مساوی هستند از شرط i = j استفاده کرده‌ایم. در پایان نیز مجدداً با استفاده از دو حلقه For اعضای آرایه را به‌صورت فوق نمایش می‌دهیم

**مثال :** می‌خواهیم پروژه‌ای طراحی کنیم که بتواند نام، نام خانوادگی و نمره سه درس ده دانش‌آموز را در آرایه ذخیره کند و در ضمن بتوان هر لحظه اطلاعات دانش‌آموزان را مشاهده کرد، فرم‌ها و کنترل‌های برنامه را مطابق شکل‌های ۵-۱۰ و ۶-۱۰ طراحی کنید. پروژه از دو فرم تشکیل می‌شود فرم frmadd برای ورود به داده‌ها و فرم frmdisplay برای نمایش اطلاعات ورودی استفاده می‌شوند.



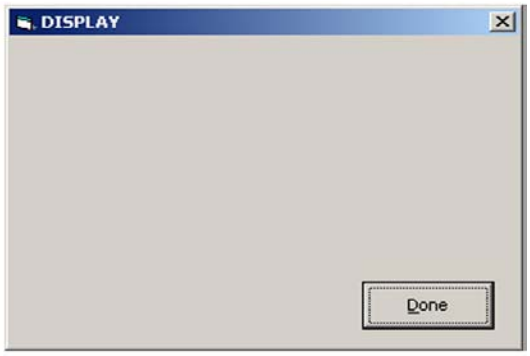

**شکل ۵-۱۰**

پس از طراحی فرم‌های پروژه، ابتدا آرایه‌های موردنیاز را مطابق دستورات زیر در یک ماژول‌کد بنویسید.

```
Public stuname (9,1)  As string 25
Public stugrade (9,2)   As single
```

همان‌طور که می‌بینید یک آرایه دوبعدی به نام stuname برای نگهداری نام و نام خانوادگی دانش‌آموزان با ۱۰ سطر و ۲ ستون تعریف کرده‌ایم تا نام در ستون اول و نام خانوادگی در ستون دوم نگهداری شود و چون اطلاعات ۱۰ نفر ذخیره می‌شود بنابراین تعداد سطرها را ۱۰ انتخاب کرده‌ایم (صفر الی ۹) تا نام و نام خانوادگی هر دانش‌آموز در یک سطر قرار بگیرد. طول نام و نام خانوادگی هر

فرد حداکثر ۲۵ کاراکتر خواهد بود اما برای ذخیره‌سازی نمرات دانش‌آموزان از یک آرایه دوبعدی دیگر از نوع اعداد اعشاری، با ۱۰ سطر (صفر الی ۹) و ۳ ستون (صفر الی ۲) استفاده شده است که نمرات هر دانش‌آموز در یک سطر جداگانه در آرایه ذخیره می‌شود و دو آرایه به‌صورت موازی و در کنار هم می‌توانند اطلاعات ده نفر را نگهداری کنند، به این معنی که اگر نام و نام خانوادگی دانش‌آموزی در سطر اول آرایه (سطر شماره صفر) stuname ذخیره شود، نمرات همان دانش‌آموز در سطر اول آرایه (سطر شماره صفر) stuname ذخیره می‌شود. به این صورت دو آرایه به‌صورت مجزا اما به موازات هم می‌توانند اطلاعات را نگهداری کنند.




**شکل ۶-۱۰**

برای ذخیره شدن اطلاعات در آرایه‌ها از رویداد Click دکمه فرمان Add استفاده می‌شود.

```
Private  Sub  cmdadd_Click( )
        Static  i  As  Integer,  j  As  Integer
        If    (  i  <  10)    Then
              stuname( i,  0)  =  txtname.Text
              stuname( i,  1)  =  txtfame.Text
              stugrade( i,  0)  =  Val(txtgrad1.Text)
              stugrade( i,  1)  =  Val(txtgrad2.Text)

              stugrade( i,  2)  =  Val(txtgrad2.Text)
             i  =  i  +  1
        Else
```

```
                MsgBox  "Array  Is  Full."
        End  If
End  Sub
```

با توجه به مطالبی که قبلاً گفته شد دستورات موجود در این رویه کاملاً واضح هستند. فرمان If از اضافه شدن اطلاعات بیش از اندازه خواسته شده جلوگیری می‌کند و در صورتی که اطلاعات در تمام اعضای آرایه ذخیره نشده باشند نام هر فرد در ستون اول و نام خانوادگی وی در ستون دوم هر سطر از آرایه stuname و نمرات وی در سه ستون از آرایه stugrade ذخیره می‌شود البته با استفاده از تابع Val مقادیر موجود در کنترل‌های TextBox مربوط به نمرات، تبدیل به مقدار عددی می‌شوند.

رویه‌های دیگر به رویدادهای Click دکمه فرمان cmdshow در فرم frmadd و دکمه فرمان Done در فرم frmdsplay مربوط می‌شوند که در زیر دستورات مربوط به این بخش‌ها دیده می‌شوند.

```
Private  Sub  cmdshow_Click(  )
        frmadd.Hide
        frmdispaly.Show
End  Sub

Private  Sub  cmd_done_Click(  )
        frmdispaly.Hide
        frmadd.Show
End  Sub
```

راجع به عملکرد این رویه‌ها در مثال‌های قبلی توضیح کافی داده شده است. آخرین رویه، مربوط به نمایش اطلاعات وارد شده است که در رویداد Activate فرم frmdisplay انجام می‌شود.

```
Private  Sub  Form_Activate(  )
        Dim  j,  k
        For  j  =  0  To  9
                Print  "NAME  IS  =",  stuname( j,  0)
                Print  "family  is  =",  stuname( j,  1)
```

```
                Print  "GRADE1  IS  =",  stugrade( j,  0)
                Print  "GRADE2  IS  =",  stugrade( j,  1)
                Print  "GRADE3  IS  =",  stugrade( j,  2)
        Next  j
End  Sub
```

در رویه قبل نیز به‌وسیله یک حلقه For به اطلاعات ذخیره شده در دو آرایه دسترسی پیدا می‌کنید.

**مثال :** می‌خواهیم یک رویه فرعی بنویسیم که یک ماتریس پایین مثلثی ۴×۴ را ایجاد کرده و نمایش دهد. (ماتریس پایین مثلثی، ماتریسی است که عناصر روی قطر اصلی و زیر قطر اصلی یک و بقیه عناصر صفر باشند).

```
Sub  mymatrix(  )
        Dim  matrix1( 3,  3)  As  Integer
        Dim  j  As  Integer
        Dim  j  As  Integer
        For  i  =  0  To  3
                For  j  =  0  To  3
                        If  ( j  >  i)  Then
                                matrix1( i,  j)  =  0
                        Else
                                matrix1( i,  j)  =  1
                        End  If
                Next  j
        Next  i
End  Sub
```

همان‌طور که در رویه قبل مشاهده می‌کنید ماتریس دوبعدی را به‌وسیله دوحلقه For مقداردهی کرده‌ایم اما برای تولید یک ماتریس پایین مثلثی از یک If استفاده شده است تا در صورتی که (j > i) باشد مقدار عضو مربوطه را در ماتریس صفر کند به عبارت دیگر به این وسیله اعضای بالای قطر اصلی را صفر می‌کنیم و در صورت غلط بودن شرط فوق یعنی در صورتی که (j < = i) باشد، عضو مربوطه در ماتریس، مقدار یک را دریافت خواهد کرد و اعضای روی قطر اصلی و زیر قطر اصلی یک خواهند شد در ادامه دستورات به‌وسیله دو حلقه For دیگر، ماتریس حاصل را نمایش می‌دهند. شکل ۷-۱۰ماتریس ایجاد شده در رویه قبل را نمایش می‌دهد.

```
1 0 0 0
1 1 0 0
1 1 1 0
1 1 1 1
```

**شکل ۷-۱۰**

## ۷-۱۰ توابع UBound و LBound

همان‌طور که در بخش‌های قبل نیز مشاهده کردید از این توابع جهت دست‌یابی به دامنه پایینی و بالایی یک آرایه استفاده شد. در این‌جا به توضیح کامل این دو تابع می‌پردازیم :

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 به‌صورت زیر است :

LBound (array name [,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. در آرایه‌های چند بعدی با استفاده از آرگومان اختیاری dimension می‌توانید بعدی از آرایه که می‌خواهید اندیس آغازین آن را پیدا کنید به‌صورت یک عدد صحیح تعیین کنید. مقدار ۱ برای بعد اول، ۲ برای بعد دوم و الی آخر. در صورت عدم استفاده از این آرگومان، مقدار پیش فرض ۱ خواهد بود.

به‌عنوان مثال به این دستورات توجه کنید :

```
Dim  A( 1  To  100,  10  To  13 )

Dim  B(10)

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 فرمان Print به ترتیب ۱ و ۱۰ خواهد بود و در صورت اجرای فرمان 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 می‌باشد که : Option Base 0 است.

تابع UBound می‌تواند با دریافت نام یک آرایه دامنه بالایی آن را به‌صورت یک عدد از نوع Long باز گرداند 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. آرگومان arrayname در واقع نام آرایه‌ای است که می‌خواهید اندیس انتهایی آن را به دست آورید. در آرایه‌های چند بعدی با استفاده از آرگومان اختیاری dimension می‌توانید بعدی از آرایه که می‌خواهید اندیس انتهایی آن را پیدا کنید، به‌صورت یک عدد صحیح تعیین کنید.

مقدار ۱ برای بعد اول، ۲ برای بعد دوم و الی آخر. در صورت عدم استفاده از این آرگومان، مقدار پیش فرض ۱ خواهد بود.

به‌عنوان مثال به دستورات زیر توجه کنید :

```
Dim  A( 1  To  100,  10  To  13 )

Dim  B( 10 )

Print  UBound( A,  1)

Print  UBound( A,  2)

Print  UBound( B )
```

اولین و دومین فرمان Print به ترتیب مقادیر ۱۰۰ و ۱۳ را نمایش می‌دهند و فرمان آخر نیز مقدار ۱۰ را نمایش خواهد داد.

**نکته**

فرمان Option Base تأثیری روی تابع UBound نمی‌گذارد.

به‌عنوان مثال به این دستورات توجه کنید:

```
Dim  A(5)
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.

## ۱۰-۸ تابع Filter

به‌وسیله این تابع می‌توانید یک مقدار رشته‌ای را در یک آرایه رشته‌ای جستجو کنید. این تابع پس از انجام جستجو، نتیجه جستجو را در آرایه دیگری ذخیره می‌کند. شکل کلی این تابع به‌صورت زیر است :

Filter (sourcearray,match [,include[,compare]] )

همان‌طور که مشاهده می‌کنید این آرایه دارای دو آرگومان اجباری و دو آرگومان اختیاری است. آرگومان sourcearray نام آرایه‌ای است که جستجو در اعضای آن انجام می‌شود و آرگومان match ، یک ثابت یا یک متغیر رشته‌ای است که در آرایه sourcearray جستجو می‌شود.

آرگومان سوم include یک آرگومان اختیاری از نوع منطقی است. اگر مقدار این آرگومان True باشد آرایه بازگشتی شامل اعضایی از آرایه sourcearray است که مقدار match در آن‌ها وجود داشته باشد اما اگر مقدار این آرگومان False باشد آرایه بازگشتی شامل اعضایی از آرایه sourcearray است که مقدار match در آن‌ها وجود ندارد. در صورت عدم استفاده از این آرگومان مقدار پیش فرض True خواهد بود. آرگومان چهارم compare نیز یک آرگومان اختیاری است که مقادیر آن در جدول ۱-۱۰ ارایه شده است. در صورت عدم استفاده از این آرگومان مقدار پیش فرض صفر خواهد بود.

**جدول ۱-۱۰ 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 |

برای روشن شدن مطلب به این دستورات توجه کنید :

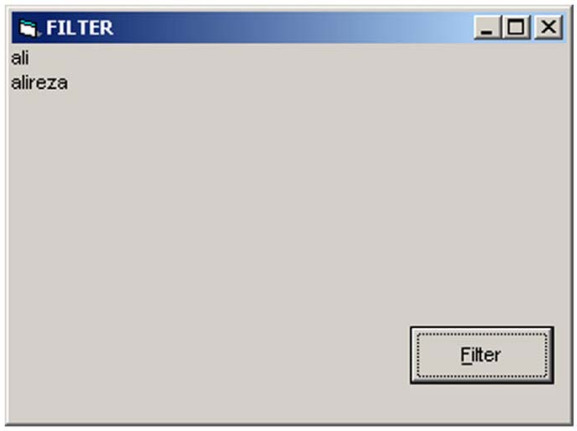
```
Dim  strname(4)  As  String,  strresult( )  As  String
Dim  strsearch  As  String,  i
strname(0)  =  "ali"
strname(1)  =  "reza"
strname(2)  =  "hamid"
strname(3)  =  "alireza"
strname(4)  =  "nima"
strsearch  =  "ali"
strresult  =  Filter(strname,  strsearch)
For  i  =  0  To  UBound(strresult)
                Print  strresult(i)
Next  i
```

همان‌طور که در دستورات قبل مشاهده می‌کنید ابتدا دو آرایه رشته‌ای تعریف شده‌اند. آرایه strname که اسامی ۵ نفر در آن ذخیره می‌شود. آرایه strresult که این آرایه جهت نگهداری مقادیری که توسط تابع بازگشت داده می‌شوند مورد استفاده قرار می‌گیرد. متغیر رشته‌ای strsearch نیز رشته مورد جستجو را در خود نگهداری می‌کند.

پس از تعریف آرایه‌ها و متغیرهای مورد نیاز، ۵ نام در آرایه strname ذخیره می‌شوند و سپس کلمه Ali در متغیر strsearch می‌شود.

در واقع هدف از این دستورات پیدا کردن مقدار strsearch در آرایه strname است. پس از مقداردهی آرایه strname و strsearch به‌وسیله تابع Filter مقدار strsearch در آرایه strname جستجو می‌شود و در پایان به‌وسیله یک حلقه For مقادیر موجود در آرایه strresult نمایش داده می‌شود. اجرای دستورات فوق سبب می‌شود تا کلمات ali و alireza نمایش داده شوند، در واقع تابع Filter مقادیر آرایه strname را یکی یکی بررسی می‌کند و در صورتی که مقدار متغیر strsearch در اعضای آرایه strname موجود باشد مقدار آن عضو را در آرایه strresult ذخیره می‌کند. مثلاً چون کلمه ali در آرایه strname (0) قرار دارد مقدار این عضو در آرایه strresult ذخیره می‌شود. اما کلمه مزبور در آرایه strname (1)

قرار ندارد و در نتیجه از مقدار این عضو در آرایه صرف‌نظر می‌شود و به همین شکل سایر اعضا نیز بررسی می‌شوند (شکل ۸-۱۰).




**شکل ۸-۱۰**

اکنون فرض کنید که در متغیر strsearch کلمه Ali ذخیره شده باشد در این صورت یک آرایه خالی بازگشت داده خواهد شد زیرا به‌طور پیش فرض a با A برابر نیست اما اگر در این حالت فرمان Filter را به‌صورت زیر تغییر دهیم :

```
strresult = Filter (strname , strsearch , , 1)
```

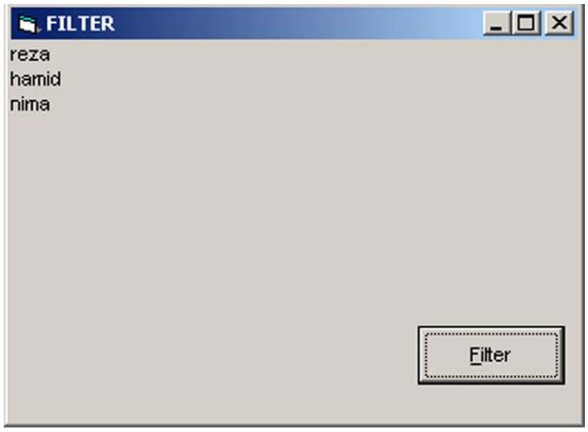
چون آرگومان compare، یک انتخاب شده است بنابراین تابع Filter بین حروف کوچک و بزرگ تفاوتی قایل نخواهد شد و مجدداً کلماتali و alireza را مشاهده خواهید کرد.

حال فرض کنید که متغیر strsearch کلمه ali را نگهداری می‌کند. تابع Filter را به‌صورت زیر استفاده می‌کنیم:

```
strresult = Filter (strname , strsearch , False)
```

در این صورت اعضایی که کلمه ali در آن‌ها به کار نرفته است در آرایه strresult ذخیره می‌شوند و کلمات reza ، hamid وnima نمایش داده خواهد شد (شکل ۹-۱۰).

در این مثال توجه شما را به نحوه تعریف آرایه strresult جلب می‌کنیم این آرایه به‌صورت Dynamic تعریف شده است. علت این است که چون تعداد اعضایی که توسط تابع Filter بازگشت داده می‌شوند قابل تغییر است استفاده از آرایه از نوع Dynamic مناسب‌تر است در صورت استفاده از آرایه با ابعاد ثابت باید ابعاد آرایه strresult حداقل با ابعاد آرایه strname یکسان باشد.




**شکل ۹-۱۰**

| نکته |
| --- |
| آرایه‌های رشته‌ای که به‌وسیله تابع Filter استفاده می‌شوند نباید از نوع رشته‌ای با طول ثابت باشند. |

**تمرین :** در صورتی که مقدار strsearch = "Ali" و تابع Filter به‌صورت زیر استفاده شود، خروجی دستورات فوق چیست؟

strresult = Filter (strname , strsearch , False , 0 )

## ۹-۱۰ 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 نیز مانند تابع Filter آرایه‌ای یک بعدی است. شکل کلی این تابع به‌صورت زیر است :

Split (expression [, delimiter [, limit [,compare ]]] )

این تابع دارای یک آرگومان اجباری است. expression می‌تواند یک عبارت رشته‌ای باشد.

آرگومان delimiter یک آرگومان اختیاری است و به‌طور پیش فرض یک کاراکتر فاصله " " است. در واقع این آرگومان کاراکتری را که جهت جدا کردن زیر رشته‌ها در آرگومان expression مورد استفاده قرار می‌گیرد معین می‌کند.

آرگومان limit دومین آرگومان اختیاری است و تعداد زیر رشته‌های مورد نظر را که باید بازگشت داده شوند تعیین می‌کند در صورت عدم استفاده از این آرگومان مقدار آن ۱- خواهد بود که

به معنی بازگشت دادن تمامی زیررشته‌هاست.

آرگومان compare نیز اختیاری بوده و تعیین می‌کند که آیا تابع بین حروف بزرگ و کوچک آرگومان delimiter با کاراکترهای expression تفاوت قایل شود یا خیر؟

مقادیری که این آرگومان می‌تواند دریافت کند در جدول ۲-۱۰ ارایه شده‌اند.

**جدول ۲-۱۰ مقادیر مربوط به آرگومان Compare**

| توضیح | ثابت عددی | ثابت رشته‌ای |
|---|---|---|
| تابع بین حروف کوچک و بزرگ تفاوت قایل می‌شود. | ۰ | vbBinaryCompare |
| تابع بین حروف کوچک و بزرگ تفاوت قایل نمی‌شود. | ۱ | vbTextCompare |

**نکته**

مقدار پیش فرض برای آرگومان compare صفر است.

برای روشن شدن بهتر مطلب به این دستورات توجه کنید:

```
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
```

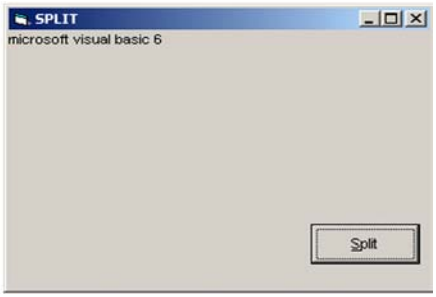
در دستورات فوق دستور Split بر اساس فضاهای خالی " " موجود در رشته strname، کلمات موجود در رشته را به‌صورت جداگانه در آرایه strresult ذخیره می‌کند که به‌وسیله یک حلقه For مطابق شکل ۱۰-۱۰ نمایش داده می‌شود.


**شکل ۱۰-۱۰**

اکنون فرض کنید تابع Split را به‌صورت زیر تغییر می‌دهیم:

strresult = Split ( strname , " A " )

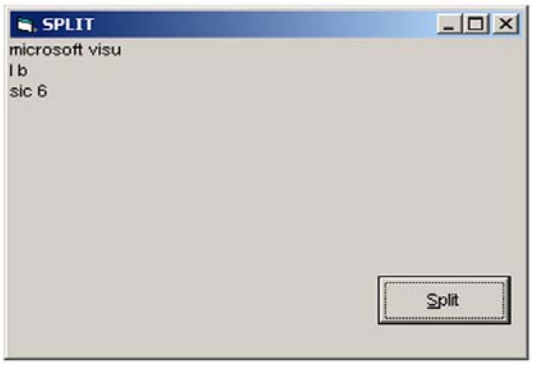
در این صورت کاراکتری که برای جدا کردن زیر رشته‌ها استفاده می‌شود کاراکتر A خواهد بود.




**شکل ۱۱-۱۰**

اما همان‌طور که در شکل ۱۱-۱۰ 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.

اکنون اگر فرمان قبل را به‌صورت strresult = Split (strname,"A",,1) استفاده کنید خروجی برنامه به‌صورت شکل ۱۲-۱۰ خواهد بود یعنی تابع بین کاراکتر A در آرگومان دوم با کاراکتر a در رشته strname تفاوتی قایل نمی‌شود و strname به سه زیر رشته تقسیم می‌شود.

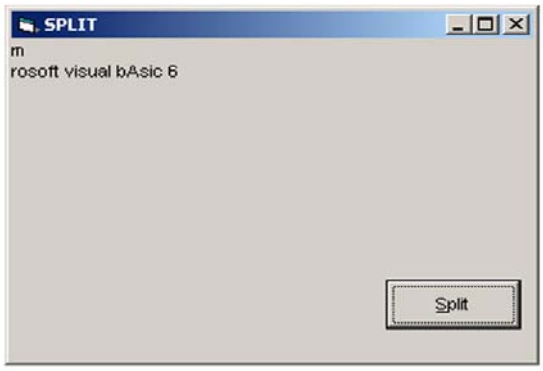



**شکل ۱۲-۱۰**

در صورتی که بخواهید تعداد معینی از زیر رشته‌ها را جدا کنید می‌توانید از آرگومان سوم در تابع Split استفاده کنید. مثلاً اگر فرمان قبل را به‌صورت زیر تغییر دهید:

```
strresult = Split (strname,"ic",2,1)
```

خروجی برنامه مطابق شکل ۱۳-۱۰ خواهد بود.




**شکل ۱۳-۱۰**

**نکته**

در صورتی که آرگومان limit در تابع Split، ۱ باشد تمام رشته expression به‌صورت یک زیر رشته بازگشت داده خواهد شد.

## ۱۰-۱۰ تابع Join

به‌وسیله این تابع می‌توانید اعضای یک آرایه رشته‌ای را به‌صورت یک مقدار رشته‌ای واحد در آورید. عملکرد این تابع برعکس تابع Split است. 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 جداکننده‌ای را که 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 :

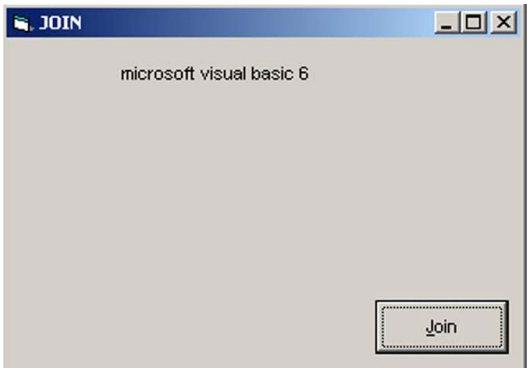
```
Dim  strname  As  String
Dim  strresult()  As  String
strname(0)  =  "Microsoft"
strname(1)  =  "Visual"
strname(2)  =  "Basic"
strname(3)  =  "6"
strresult  =  Join(strname)
Print
Print  ,  strresult
```

در صورت اجرای دستورات فوق خروجی مشابه شکل ۱۰-۱۴ 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.

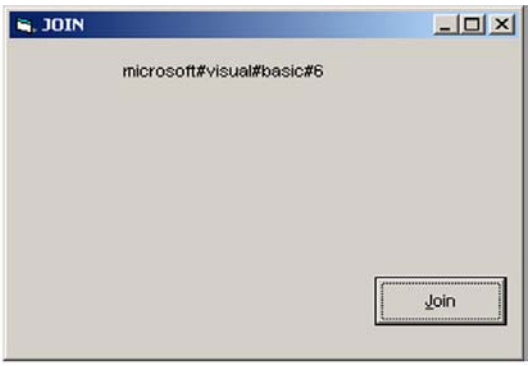
اکنون تابع Join را به‌صورت زیر استفاده کنید:

strresult = Join ( strname , " # " )

در این صورت خروجی دستورات به‌صورت شکل ۱۵-۱۰ خواهد بود.




**شکل ۱۴-۱۰**




**شکل ۱۵-۱۰**

## خلاصه مطالب

- از آرایه‌ها برای ذخیره‌سازی و نگهداری آسان‌تر داده‌ها، مرتب‌سازی و جستجوی سریع اطلاعات استفاده می‌شود.
- آرایه‌ها به دو دسته کلی، آرایه‌های 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 فرمان Erase می‌توان آرایه‌های پویا را حذف کرد.
- مقادیر موجود در یک آرایه با ابعاد ثابت به‌وسیله فرمان Erase 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.
- برای فراخوانی یک رویه با تعداد آرگومان‌های نامعین در تعریف رویه قبل از نام آرایه از کلمه کلیدی ParamArray استفاده کنید.
- روش‌های مرتب‌سازی آرایه‌ها عبارتند از : مرتب‌سازی حبابی، مرتب‌سازی به روش Shell Sort، روش مرتب‌سازی Quick Sort و روش مرتب‌سازی Insertion Sort.
- برای جستجوی اطلاعات در یک آرایه می‌توان از روش جستجوی خطی و یا دودویی استفاده کرد.
- روش مرتب‌سازی حبابی کندترین روش مرتب‌سازی آرایه‌هاست.
- توابع UBound و LBound به ترتیب دامنه بالایی و پایینی آرایه مورد نظر را محاسبه می‌کنند.
- به‌وسیله تابع Filter می‌توان یک مقدار رشته‌ای را در یک آرایه رشته‌ای جستجو کرد.
- تابع Split می‌تواند یک عبارت رشته‌ای را به‌صورت زیر رشته‌های جداگانه در یک آرایه یک بعدی ذخیره کند.
- تابع Join بر خلاف تابع Split، می‌تواند مقادیر رشته‌ای موجود در یک آرایه رشته‌ای را به‌صورت یک عبارت رشته‌ای واحد درآورد.

# آزمون پایانی

**۱- در ویژوال بیسیک اندیس اول یک آرایه به‌طور پیش‌فرض از چه شماره‌ای آغاز می‌شود؟**

۱- ۱- ۲- صفر ۳- ۱ ۴- ۲

**۲- پس از تعریف یک آرایه پویا به‌وسیله کدام دستور آرایه قابل استفاده می‌شود؟**

۱- Dim ۲- Static ۳- Erase ۴- ReDim

**۳- به‌وسیله کدام فرمان می‌توان شماره اندیس اولین عضو آرایه را تعیین کرد؟**

۱- Option Compare Text ۲- Option Explicit

۳- Option Base ۴- Option Keyword

**۴- آرایه ( 10,5 ) No چند عضو دارد؟ ( Option Base 2 )**

۱- ۳۶ ۲- ۵۰ ۳- ۴۵ ۴- ۴۰

**۵- فرمان Erase می‌تواند آرایه‌های..... را حذف کند.**

۱- پویا ۲- با ابعاد ثابت

۳- رشته‌ای ۴- گزینه‌های ۱و۲ صحیح هستند.

**۶- اگر ( 10 ) name باشد حاصل عبارت ( name ) UBound چیست؟**

۱- ۸ ۲- ۹ ۳- ۱۰ ۴- ۱۱

**۷- کدام روش مرتب‌سازی از کم‌ترین سرعت برخوردار است؟**

۱- Bubble Sort ۲- Shell Sort

۳- Quick Sort ۴- Insertion Sort

**۸- کدام تابع می‌تواند اعضای یک‌آرایه رشته‌ای را به یک عبارت رشته‌ای واحد تبدیل کند؟**

۱- Split ۲- Join

۳- Ubound ۴- Preserve

**۹- کدام تابع می‌تواند عملیات جستجو را در یک آرایه رشته‌ای انجام دهد؟**

۱- Split ۲- Filter ۳- Join ۴- Preserve

**۱۰- کدام تابع می‌تواند یک عبارت رشته‌ای را به‌صورت کلمات جداگانه‌ای در یک آرایه ذخیره کند؟**

۱- Split ۲- Join ۳- Preserve ۴- Filter

# دستور کار آزمایشگاه

۱- رویه‌ای بنویسید که یک ماتریس ۵×۵ را با اعضای تصادفی ایجاد کرده سپس مجموع هر سطر و هر ستون ماتریس را به‌طور جداگانه به همراه خود ماتریس نمایش دهد.

۲- رویه‌ای بنویسید که دو ماتریس دلخواه را دریافت کرده و ماتریس حاصل‌ضرب آن دو را به همراه دو ماتریس نمایش دهند.

۳- رویه‌ای بنویسید که دو ماتریس یک بعدی دلخواه را دریافت کرده و سپس ماتریس 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.

۵- یک پروژه از نوع Standard EXE طراحی کنید که بتواند نام و نام خانوادگی و شماره دانشجویی ۵۰ دانشجو را به همراه نمرات ۱۰درس آن‌ها دریافت کند و امکان ارایه گزارشات زیر وجود داشته باشد:

**الف**ـ اطلاعات ورودی مربوط به تمام دانشجویان به ترتیب شماره دانشجویی

**ب**ـ کاربر بتواند با استفاده از شماره دانشجویی یک فرد اطلاعات ورودی وی را مشاهده کند و در صورت عدم وجود دانشجوی مورد نظر پیام مناسبی نمایش داده شود.

**ج**ـ کاربر بتواند با استفاده از شماره دانشجویی یک فرد کارنامه وی را مشاهده کند و در صورت عدم وجود دانشجوی مربوطه پیام مناسبی نمایش داده شود.

**د**ـ کاربر بتواند اسامی و معدل دانشجویانی را که بالاترین و پایین‌ترین معدل را کسب کرده‌اند مشاهده نماید.

**توجه :** برای نمایش اطلاعات در هر قسمت از مسأله فوق از یک فرم جداگانه استفاده شود.

## پاسخ پیش‌آزمون

| | | | |
|---|---|---|---|
| ۱ـ ۱ | ۲ـ ۳ | ۳ـ ۴ | ۴ـ ۳ |
| ۵ـ ۲ | ۶ـ ۴ | ۷ـ ۲ | ۸ـ ۲ |
| ۹ـ ۲ | ۱۰ـ ۳ | | |

## پاسخ آزمون پایانی

| | | | |
|---|---|---|---|
| ۱ـ ۲ | ۲ـ ۴ | ۳ـ ۳ | ۴ـ ۱ |
| ۵ـ ۱ | ۶ـ ۳ | ۷ـ ۱ | ۸ـ ۲ |
| ۹ـ ۲ | ۱۰ـ ۱ | | |

فصل یازدهم

## هدف کلی

# توانایی استفاده از جلوه‌های گرافیکی و چاپ در ویژوال بیسیک

| زمان (ساعت) | |
|---|---|
| نظری | عملی |
| ۳ | ۶ |

### هدفهای رفتاری ▼

**پس از مطالعه این فصل از فراگیر انتظار می‌رود که:**

۱- مفاهیم مربوط به سیستم مختصات را بداند.

۲- توانایی تغییر مختصات را جهت انجام ترسیمات داشته باشد.

۳- توانایی به کارگیری متدهای گرافیکی زیر را داشته باشد :

PSet , Line , Circle , Point , Cls , Print , TextHeight , TextWidth

۴- توانایی استفاده از خواص 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، خواص و متدهای آن را در برنامه‌ها داشته باشد.

# پیش آزمون

**۱- در صورتی که بخواهیم در زمان تغییر ابعاد یک آرایه پویا مقادیر قبلی آرایه از بین نرود از کدام کلمه کلیدی استفاده می‌کنیم؟**

۱- Dim ۲- ReDim ۳- Para ۴- Preserve

**۲- در صورتی که Option Base 1 باشد تعداد اعضای آرایه A (5) چقدر خواهد بود؟**

۱- ۵ ۲- ۶ ۳- ۷ ۴- ۸

**۳- کدام روش جستجو از سرعت بالاتری برخوردار است؟**

۱- Liner ۲- Binary ۳- Quick ۴- Shell

**۴- کدام تابع می‌تواند مقدار دامنه بالایی یک آرایه را محاسبه کند؟**

۱- LBound ۲- UBound

۳- LUBound ۴- ULBound

**۵- در صورت نامعین بودن تعداد آرگومان‌های ارسالی به یک رویه از کدام کلمه کلیدی می‌توان استفاده کرد؟**

۱- Preserve ۲- ReDim ۳- ParamArray ۴- Erase

**۶- آرایه ( 1 to 5 و 3 to 6 ) Color چند عضو دارد؟**

۱- ۱۵ ۲- ۲۰ ۳- ۲۵ ۴- ۳۰

**۷- اگر (10 to 17 , 2 to 9 ) grade آن‌گاه حاصل LBound ( grade , 2) چیست؟**

۱- ۱۰ ۲- ۱۷ ۳- ۲ ۴- ۹

**۸- درهنگام ارسال آرایه‌ها به رویه‌ها به‌طور پیش‌فرض ازکدام روش ارسال استفاده می‌شود؟**

۱- مرجع ۲- مقدار

۳- بستگی به انتخاب کاربر دارد. ۴- گزینه‌های ۱و۲ صحیح هستند.

## مقدمه

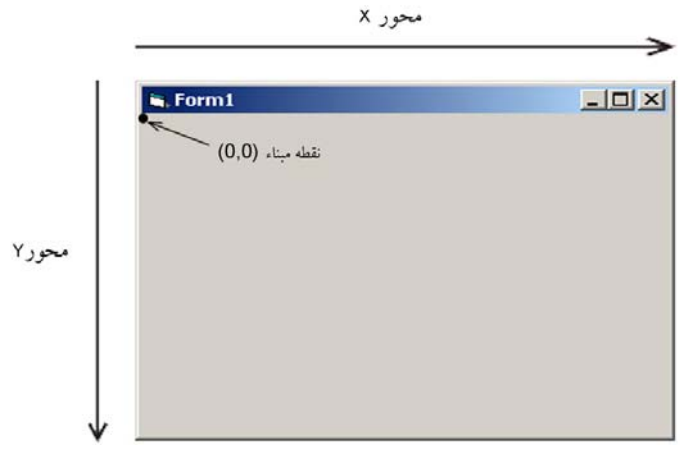
تاکنون چگونگی طراحی و ساخت رابط‌های گرافیکی را با استفاده از فرم‌ها و بعضی از کنترل‌ها آموختید؛ اما گاهی اوقات وجود فرم‌ها و کنترل‌ها بدون جلوه‌های گرافیکی محیط نرم‌افزار را خسته‌کننده و غیر قابل استفاده می‌کند. با استفاده از جلوه‌های گرافیکی نظیر رنگ، نمودارهای گرافیکی و تصاویر متحرک می‌توانید به کاربر در مشاهده و درک بهتر گزارشات و نتیجه محاسبات کمک کنید.

ویژوال بیسیک قابلیت بالایی در استفاده از جلوه‌های گرافیکی در اختیار شما قرار می‌دهد و می‌توانید این جلوه‌های گرافیکی را با دو روش ایجاد کنید: کنترل‌های گرافیکی مثل Shape، Line و متدهای گرافیکی مثل Circle ، Pset ، Line و...

با کنترل‌های گرافیکی نظیر Shape و Line قبلاً آشنا شده‌اید در این فصل ابتدا به معرفی سیستم مختصات در ویژوال بیسیک می‌پردازیم، سپس خواص گرافیکی مربوط به فرم‌ها و کنترل‌ها و نحوه استفاده از متدهای گرافیکی و چگونگی تنظیم پارامترهای چاپ در چاپگرها را بررسی می‌کنیم.

## ۱-۱۱ مفهوم سیستم مختصات در ویژوال بیسیک

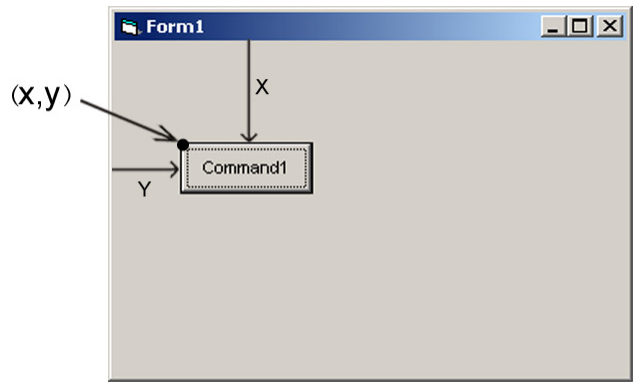
در ویژوال بیسیک نیز مانند ریاضیات و هندسه برای انجام هر نوع ترسیمی‌از سیستم مختصات استفاده می‌شود، هم‌چنین از دو بعد یا محور مختصات جهت تعیین موقعیت ترسیمات یا چاپ آن‌ها استفاده می‌شود (شکل ۱-۱۱).




**شکل ۱-۱۱**

مختصات هر نقطه به‌صورت (X,Y) تعیین می‌شود مقدار X موقعیت نقطه را در طول محور X و مقدار Y موقعیت نقطه را در طول محور Y بیان می‌کند که مقدار شروع در هر یک از محورها صفر می‌باشد. دقت داشته باشید که محورهای مختصات در ویژوال بیسیک با محورهای مختصات در ریاضیات متفاوت است. در شکل ۱-۱۱ نحوه قرار گرفتن محورها بر نقطه مبنا قابل مشاهده است. مختصات نقطه مبنا ( 0 و 0 ) است که در گوشه بالا و چپ شئ مربوطه که معمولاً فرم است قرار دارد.

وقتی کنترلی را جابه‌جا کرده و 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 ۲-۱۱ 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 ویژوال بیسیک هر محور در سیستم مختصات می‌تواند مقیاس خاص خود را داشته باشد.

## ۲-۱۱ تغییر سیستم مختصات

می‌توانید سیستم مختصات یک شئ خاص مثل فرم را به‌وسیله خاصیت Scale و 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 و یا این که یک مقیاس جدید را ایجاد کنید. با تغییر مقیاس سیستم مختصات می‌توانید اندازه و موقعیت ترسیمات گرافیکی را در روی فرم با توجه به

نیازتان به آسانی تنظیم کنید.

هر فرم یا بعضی از کنترل‌ها مانند PictureBox چندین خاصیت 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 ۱-۱۱ را برای خاصیت ScaleMode فرم یا کنترل مورد نظر خود در نظر بگیرید.

**جدول ۱-۱۱ مقادیری که خاصیت ScaleMode کسب می‌کند.**

| ثابت رشته‌ای | ثابت عددی | توضیح (نوع مقیاس) |
|---|---|---|
| vbUser | 0 | مقادیر تعریفی کاربر در خواص ScaleHeight، ScaleWidth، ScaleTop، ScaleLeft استفاده می‌شوند. |
| vbTwips | 1 | twip |
| vbPoints | 2 | Point ( ۷۲Point = ۱Inch ) |
| vbPixels | 3 | Pixel ( یک pixel کوچک‌ترین واحد نمایشی در صفحه نمایش یا چاپگر است و تعداد آن‌ها در هر اینچ به مقدار وضوح تصویر یا چاپ بستگی دارد ) |
| vbCharacters | 4 | Character |
| vbInches | 5 | Inch |
| vbMillimeters | 6 | Milimeter |
| vbCentimeters | 7 | Centimeter |

اگر بخواهید مختصات نقطه مبنا را تغییر دهید و یا مقیاس جدید را در یک کنترل یا فرم ایجاد کنید می‌توانید از خواص ScaleHeight، ScaleWidth، ScaleTop و ScaleLeft استفاده کنید. خواص ScaleLeft و ScaleTop با دریافت مقادیر عددی، مختصات نقطه مبنا را در کنترل، فرم و یا چاپگر معین می‌کنند. مقدار پیش‌فرض برای این دو خاصیت صفر است. مقدار این خواص را می‌توانید از طریق پنجره خواص 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 خواص به‌صورت زیر است :

[ object. ] ScaleLeft [ = value ]

[ object. ] ScaleTop [ = value ]

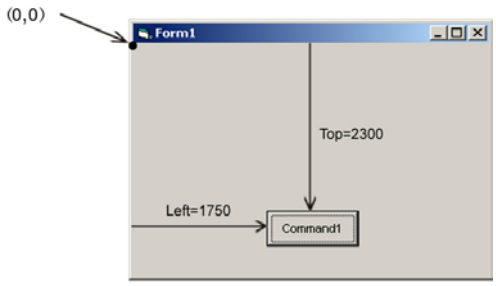
منظور از object نام یک فرم، کنترل یا چاپگر است و استفاده از آن اختیاری است و value یک عبارت عددی است که مختصات نقطه مبنا را تعیین می‌کند و استفاده از آن اختیاری است. در صورت عدم استفاده از value می‌توانید مقدار فعلی مختصات نقطه مبنا را به‌دست آورید. به‌عنوان مثال فرض کنید در یک فرم همراه با یک کنترل دکمه فرمان مقادیر زیر تنظیم شده است:

```
Form1. ScaleMode = 1
Command1.Top = 2300
Command1.Left = 1750
```

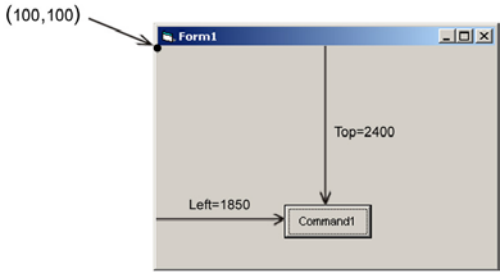
اگر بخواهیم در این فرم مختصات نقطه مبنا را (100 , 100) قرار دهیم در این صورت خواص مربوطه را به‌صورت زیر تنظیم می‌کنیم:

```
Form1.ScaleTop = 100
Form1.ScaleLeft = 100
```

در این صورت مقدار خاصیت ScaleMode در Form1 به‌طور خودکار روی مقدار صفر قرار می‌گیرد و در ضمن خاصیت Top و Left در دکمه فرمان به‌طور خودکار به ترتیب روی مقادیر ۲۴۰۰ و ۱۸۵۰ تنظیم 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 (100 , 100) تنظیم می‌شود. دو حالت قبل را می‌توانید در شکل‌های ۳-۱۱ و ۴-۱۱ مشاهده کنید.




**شکل ۳-۱۱**




**شکل ۴-۱۱**

همان‌طور که گفته شد به‌وسیله این دو خاصیت می‌توانید از مقدار مختصات نقطه مبنا مطلع شوید در رویه زیر پس از تغییر مقدار این دو خاصیت، 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 )
```

**نکته**

در صورت عدم استفاده از بخش object، فرمی که فوکوس دارد در نظر گرفته خواهد شد.

علاوه بر تغییر مختصات نقطه مبنا، می‌توانید مقیاس سیستم مختصات را تغییر دهید. خواص ScaleWidth و ScaleHeight واحد اندازه‌گیری در محورهای X و Y را تعیین می‌کنند. مقدار این خواص را می‌توانید از طریق پنجره خواص و یا با نوشتن کد مناسب تغییر دهید. شکل کلی استفاده از این خواص به‌صورت زیر است:

```
[ object. ] ScaleHeight [ = value ]
[ object. ] ScaleWidth [ = value ]
```

منظور از object نام یک فرم، کنترل یا چاپگر است و value یک عبارت عددی است که واحد اندازه‌گیری را در محورها تعیین می‌کند. استفاده از این دو بخش اختیاری است. در صورتی که object تعیین نشود، فرمی‌که فوکوس دارد در نظر گرفته خواهد شد و در صورت عدم استفاده از بخش value می‌توانید مقادیر مربوط به این دو خاصیت 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) فرم و در محور عمودی (Y)، یک پانصدم ( $\frac{۱}{۵۰۰}$ ) ارتفاع داخلی (Height) فرم خواهد بود.

**نکته**

خواص ScaleWidth و ScaleHeight واحدها را با توجه به ابعاد داخلی فرم یا شئ مربوط تعیین می‌کنند. این ابعاد شامل حاشیه‌ها یا منوها یا نوار عنوان نمی‌شوند. این دو خاصیت همواره در رابطه با بخش قابل دسترس در 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
```

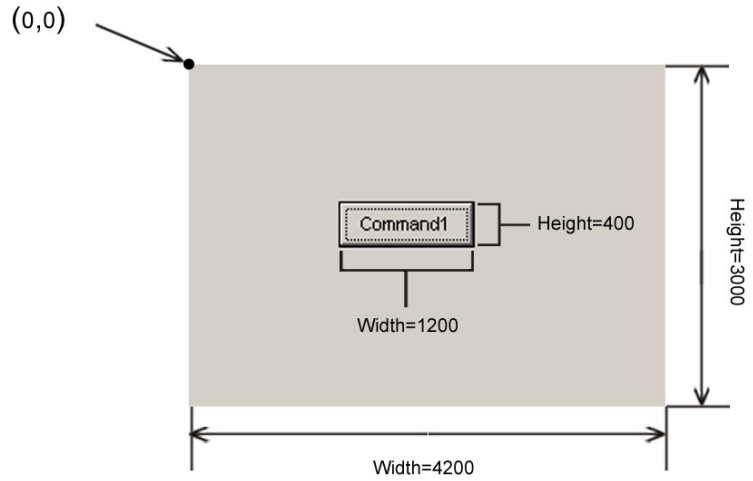
اگر تنظیمات فوق را به‌ترتیب انجام دهید و سپس مقدار خواص ScaleHeight و ScaleWidth را ملاحظه کنید خواهید دید که مقدار این دو خاصیت مانند مقدار ارتفاع و عرض فرم است زیرا فرم شما بدون حاشیه و نوار عنوان است و تمام فضای فرم، فضای قابل دسترس محسوب می‌شود. اما اگر مقدار خاصیت BorderStyle را Sizeable قرار دهید مقدار دو خاصیت ScaleHeight و ScaleWidth مقادیر کمتری خواهند بود زیرا بخشی از ابعاد فرم به نوار عنوان و حاشیه‌ها داده شده است که جزء فضاهای قابل دسترس نیستند. هم‌چنین معیار اندازه‌گیری در محورها توسط خاصیت ScaleMode تعیین می‌شود که از نوع twip است.

اکنون خواص ScaleHeight و ScaleWidth فرم را روی ۵۰۰ و ۱۰۰۰ تنظیم کنید همان‌طور که مشاهده کردید با تغییر یکی از این خواص، خاصیت ScaleMode به‌طور خودکار روی مقدار User-0 تنظیم می‌شود پس از تغییر دو خاصیت فوق، خواص Height و Width کنترل Command1 را مشاهده کنید. همان‌طور که می‌بینید مقدار آن‌ها به ترتیب به ۸۳/۳۳۳ و ۲۳۸/۰۹۵ تغییر کرده است. در واقع برای محاسبه هر یک از روش زیر استفاده شده است.

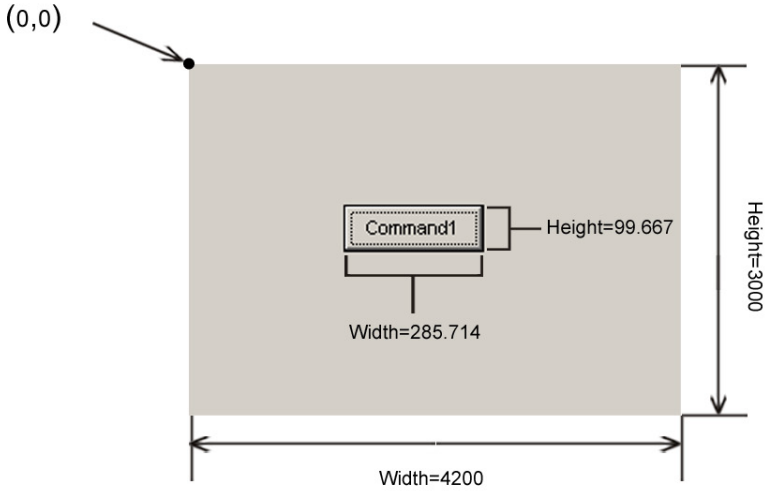
مقدار خاصیت Command1.Height $۴۰۰ \times \frac{۵۰۰}{۳۰۰۰} = ۶۶/۶۶۷$

مقدار خاصیت Command1.Width $۱۲۰۰ \times \frac{۱۰۰۰}{۴۲۰۰} = ۲۸۵/۷۱۴$

در دو شکل زیر:




**شکل ۵-۱۱**




**شکل ۶-۱۱**

**نکته**

تنظیم هر یک از خواص Scale، مقدار خاصیت ScaleMode را به‌طور خودکار روی 0 تنظیم می‌کند.

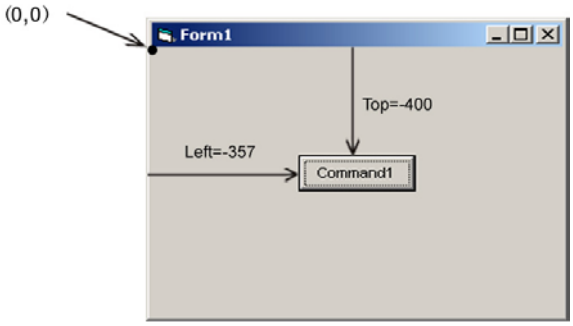
**نکته**

انتخاب مقیاس بزرگ‌تر از صفر برای خاصیت ScaleMode مقدار خواص ScaleHeight و ScaleWidth را به‌طور خودکار روی مقادیر جدید تنظیم می‌کند و مقدار خواص ScaleTop و ScaleLeft را روی صفر تنظیم می‌کند.

**نکته**

هر چهار خاصیت Scale می‌توانند مقادیر اعشاری و حتی منفی داشته باشند. در صورت استفاده از اعداد منفی برای خواص ScaleWidth و ScaleHeight، جهت محورها در سیستم مختصات عوض می‌شود مثلاً برای فرم و کنترلی با مشخصات زیر فرمی‌مطابق شکل ۷-۱۱ خواهید داشت:

```
Form1.ScaleWidth  =  -1000
Form1.ScaleHeight  =  -1000
Form1.ScaleLeft  =  0
Form1.ScaleTop  =  0
Command1.Left  =  -357
Command1.Top  =  -400
```




**شکل ۷-۱۱**

به‌عنوان آخرین مثال در رابطه با دو خاصیت 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
```

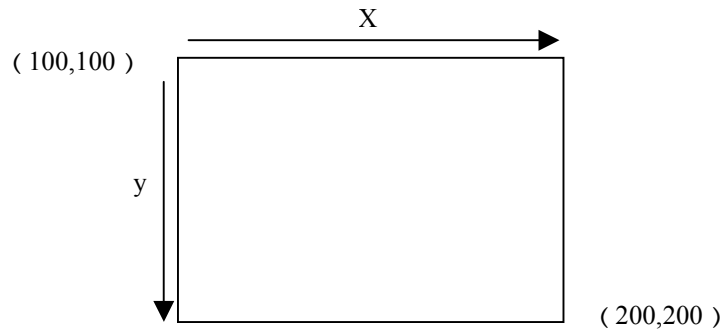
همان‌طور که مشاهده کردید ابتدا نوع مقیاس اندازه‌گیری twip در نظر گرفته شده است و سپس عرض و ارتفاع فرم مقدار دهی می‌شود. اولین دستور Print ابتدا عرض و ارتفاع فرم یعنی ۴۲۰۰ و ۳۰۰۰ و دو فرمان بعدی نیز مقادیر ۴۰۸۰ و ۲۵۹۵ را برای خواص ScaleWidth و ScaleHeight نمایش می‌دهد. در واقع این دو خاصیت ابعاد فرم را بدون در نظر گرفتن حاشیه و نوار عنوان فرم نمایش می‌دهند. اما در خطوط بعدی مقدار دو خاصیت ScaleWidth و ScaleHeight روی ۱۰۰۰ و ۱۵۰۰ تنظیم می‌شود و همان‌طور که قبلاً نیز گفته شد در این حالت خاصیت ScaleMode نیز به‌طور خودکار روی صفر تنظیم می‌شود. پس از مقداردهی دو خاصیت ScaleWidth و ScaleHeight، سه دستور Print دیگر اجرا می‌شوند اولین Print همان ابعاد قبلی یعنی ۴۲۰۰ و ۳۰۰۰ را برای ابعاد فرم نمایش می‌دهند اما دو دستور Print بعد مقادیر جدید دو خاصیت ScaleWidth و ScaleHeight یعنی ۱۰۰۰ و ۱۵۰۰ را نمایش خواهند داد.

آخرین روشی را که در رابطه با تغییر سیستم مختصات به آن می‌پردازیم 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 x1,y1 مختصات گوشه بالایی و سمت چپ شئ و مقادیر عدد x2,y2 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 فرمان زیر سیستم مختصات را در فرم به‌صورت شکل ۸-۱۱ در می‌آورد.

Form1. Scale ( 10,10 )-( 200,200 )




**شکل ۸-۱۱**

در واقع فرمان فوق چهار خاصیت Scale را به این صورت تنظیم می‌کند:

ScaleWidth = 190
ScaleHeight = 190
ScaleTop = 10
ScaleLeft = 10

## ۳-۱۱ خواص 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 خواص گرافیکی نیاز گرافیکی شما را در پروژه‌های برنامه‌نویسی برآورده می‌کند.

### ۱-۳-۱۱ متد PSet

به‌وسیله این متد می‌توانید نقاط مورد نظرخود را در مکان‌های مناسب قرار دهید شکل کلی این متد به‌صورت زیر است:

[ object. ] PSet [ Step ] (x,y) , [ color ]

**توجه**

بخش‌هایی که در بین علامت [ ] قرار دارند اختیاری هستند. منظور از object، شئ است که نقطه روی آن رسم می‌شود. اگر از ذکر آن خودداری کنید فرمی‌که فوکوس را در اختیار دارد مکان رسم نقطه است. ذکر نام شئ اختیاری است.

کلمه کلیدی step نیز اختیاری است و در صورت استفاده از آن هنگام رسم نقطه، مکان ترسیم با توجه به مقدار خواص فعلی CurrentX و CurrentY در شیئی که در آن ترسیم انجام می‌شود، انتخاب می‌شود. در مورد این دو خاصیت در آینده توضیح داده خواهد شد.

x , y مقادیر عددی از نوع Single هستند که مختصات محل ترسیم نقطه را مشخص می‌کنند. علاوه بر موارد فوق می‌توانید رنگ نقطه مورد نظر را به‌وسیله بخش color تعیین کنید در صورت عدم استفاده از این قسمت، رنگی که در خاصیت ForeColor شیئی که نقطه در آن رسم می‌شود (مثلاً فرم) در نظر گرفته خواهد شد. جدول مربوط به مقادیر رنگ‌ها در جدول ۲-۱۱ آورده شده است. می‌توانید از ثابت‌های رشته‌ای یا از ثابت‌های عددی در مبنای ۱۶ استفاده کنید.

**جدول ۲-۱۱ 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 فرمان یک نقطه به رنگ آبی در روی فرم نمایش می‌دهد:

```
Private  Sub  cmdpset_Click( )
        Form1.ForeColor  =  vbBlue
        PSet  (1000,  200)
End  Sub
```

اکنون فرمان زیر را در نظر بگیرید:

```
PSet ( 500 , 700 ) , vbCyan
```

فرمان فوق نقطه‌ای را با رنگ فیروزه‌ای در مختصات ۵۰۰ و ۷۰۰ رسم می‌کند حال اگر پس از اجرای این دستور فرمان زیر اجرا شود:

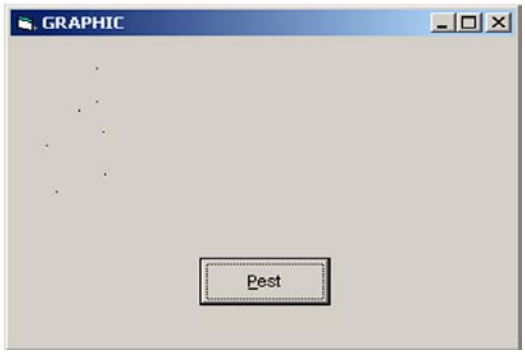
```
PSet Step ( 500 , 700 ) , vbGreen
```

مختصات نقطه مربوطه با توجه به مختصات نقطه رسم شده قبلی محاسبه می‌شود بنابراین دو نقطه در روی یکدیگر قرار نخواهند گرفت.

به‌عنوان آخرین مثال به دستورات زیر توجه 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
```

اجرای این دستورات هر بار ۱۰ نقطه را به‌صورت تصادفی رسم می‌کند (شکل ۹-۱۱).




**شکل ۹-۱۱**

## ۲-۳-۱۱ 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) [ step ] - ( x2,y2 ) , [ color ] , [B] [F]

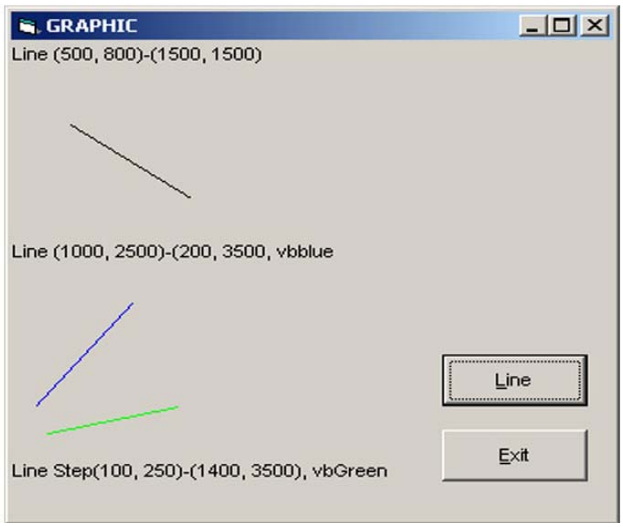
عملکرد گزینه‌های object و step مانند متد PSet است. مقادیر x1,y1 مختصات نقطه ابتدای خط و مقادیر x2,y2 مختصات نقطه انتهایی را در خط تعیین می‌کنند. به‌وسیله بخش color نیز می‌توانید رنگ خط را مشخص کنید.

استفاده از حرف B سبب‌تولید یک مستطیل خالی و استفاده ازحرف F به همراه حرفB ، (BF) یک مستطیل توپر ایجاد می‌کند، البته استفاده از این دو کاراکتر اختیاری است. برای مشاهده مثال به جدول ۳-۱۱ مراجعه کنید.

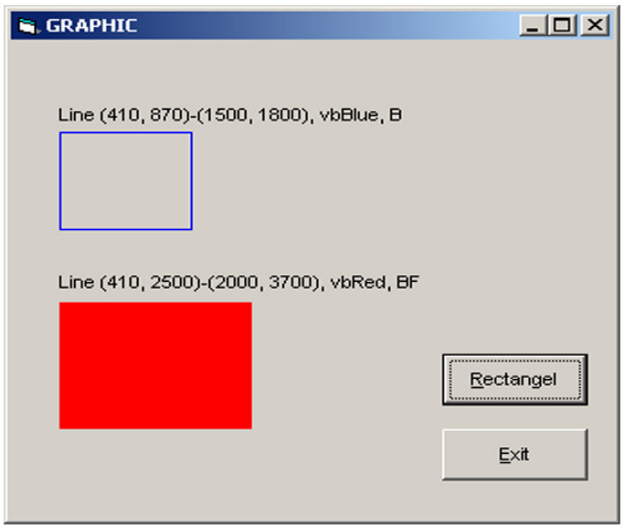
**جدول۳-۱۱**

| توضیح | مثال |
|---|---|
| یک خط با رنگ قرمز ایجاد می‌کند. | Line (500,500)-(1800,1800) ,vbRed |
| یک مستطیل با رنگ ForeColor فرم ایجاد می‌کند. | Line (500,500)-(1800,1800), , B |
| یک مستطیل توپر با رنگ زرد رسم می‌کند. | Line (500,500)-(1800,1800) , vbYellow, BF |

در شکل ۱۰-۱۱ و ۱۱-۱۱ انواع ترسیمات را با متد Line مشاهده می‌کنید.




**شکل ۱۰-۱۱**




**شکل ۱۱-۱۱**

همان‌طور که در شکل ۱۰-۱۱ می‌بینید سومین خط از متد Line به همراه کلمه کلیدی Step استفاده شده است در نتیجه مختصات ابتدای خط سوم از انتهای خط قبل محاسبه خواهد شد.

### ۳-۳-۱۱ 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 می‌باشد. مقادیر عددی x,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 جهت ترسیم معین می‌کند. مقدار مجاز برای این دو مقدار از $2\pi$- رادیان تا $2\pi$ رادیان است. در صورت عدم استفاده از این دو مقدار، کمان ترسیمی از صفر تا $2\pi$ رادیان در نظر گرفته می‌شود.

**نکته**

جهت ترسیم کمان، جهت خلاف 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.

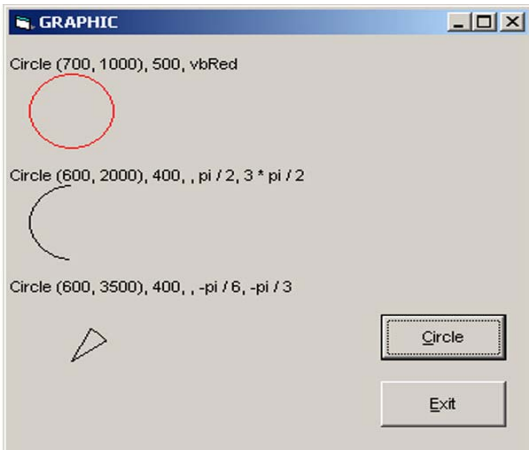
جهت مشاهده مثال‌هایی از متد Circle به شکل‌های ۱۲-۱۱ و ۱۳-۱۱ مراجعه کنید.

**نکته**

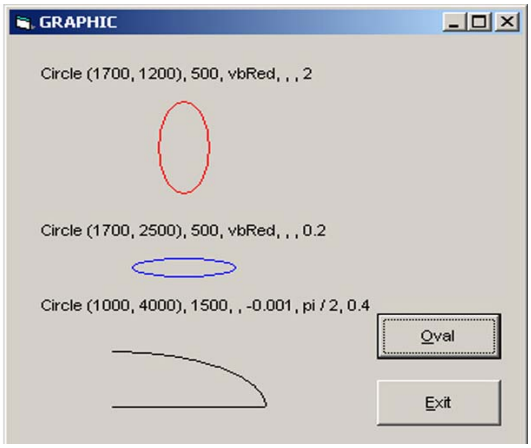
منظور از عدد $\pi$، ثابت ۳/۱۴۱۵۹۲۶۵۳۵۸۹۷۹ است.

**نکته**

برای رسم قطاع‌های بیضی و یا دایره همراه با خطوط شعاع آن‌ها از مقادیر منفی استفاده کنید.




**شکل ۱۲-۱۱**




**شکل ۱۳-۱۱**

## ۱۱-۳-۴ متد Point

این متد با دریافت مختصات یک نقطه، شماره رنگ آن را به‌صورت یک عدد صحیح از نوع Long باز می‌گرداند. شکل کلی این متد به‌صورت زیر است:

[ object. ] Point (x,y)

| نکته |
| --- |
| در صورتی که مختصاتی که به متد Point داده می‌شود از محدوده سیستم مختصات شئ که تصویر در آن قرار دارد تجاوز کند این متد مقدار ۱- را باز می‌گرداند. |

به‌عنوان مثال فرض کنید که با استفاده از متد Line، یک مستطیل توپر با رنگ قرمز رسم کرده‌ایم به دستورات زیر توجه کنید:

```
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.

## ۵-۳-۱۱ خواص CurrentX و CurrentY

به‌وسیله این دو خاصیت می‌توانید موقعیت جاری مکان نما در صفحه ترسیمات را تغییر دهید. خاصیت CurrentX مختصات مکان نما را در جهت محور X و خاصیت CurrentY مختصات مکان نما را در جهت محور Y تعیین می‌کنند. شکل کلی نحوه استفاده از این دو خاصیت به‌صورت زیر است:

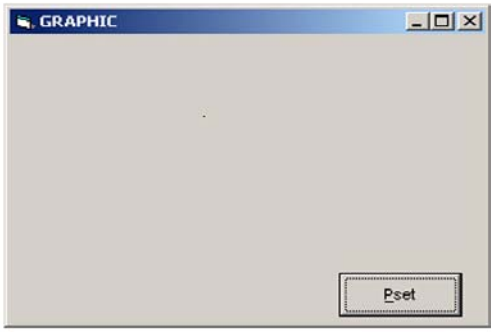
```
[ object. ] CurrentX [ = x ]
[ object. ] CurrentY [ = y ]
```

مقادیر x,y مقادیر عددی هستند که موقعیت مکان نما را در سطح شئ که ترسیمات گرافیکی روی آن انجام می‌گیرد مشخص می‌کنند.

object می‌تواند نام یک فرم، کنترل PictureBox و یا شئ چاپگر باشد. اگر مقادیر x,y استفاده نشود می‌توان مختصات فعلی مکان نما را به‌دست آورد. به دستورات زیر توجه کنید:

```
Private  Sub  cmdcurrentxy_Click( )
        Form1.CurrentX  =  2000
        Form1.CurrentY  =  1000
        PSet  (CurrentX,  CurrentY)
End  Sub
```

در رویه قبل ابتدا مختصات مکان نما به نقطه ( ۱۰۰۰ و ۲۰۰۰ ) تغییر پیدا می‌کند سپس به‌وسیله مقدار این دو خاصیت نقطه‌ای در همین مختصات رسم می‌شود(شکل ۱۴-۱۱).

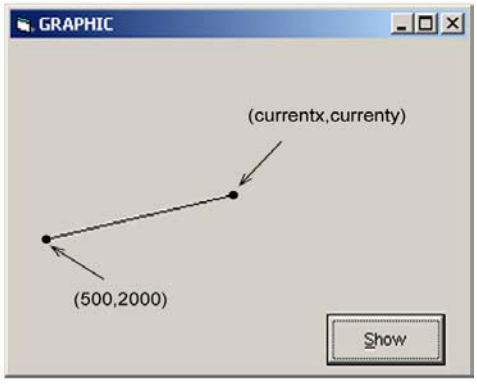



**شکل ۱۴-۱۱**

در دستورات زیر یک خط از وسط فرم تا نقطه ( ۲۰۰۰ و ۵۰۰ ) رسم می‌شود.

```
Private  Sub  cmdshow_Click(  )
        Form1.CurrentX  =  ScaleWidth  /  2
        Form1.CurrentY  =  ScaleHeight  /  2
        Line  (CurrentX,  CurrentY)-(500,  2000)
End  Sub
```

در رویه فوق با استفاده از خواص 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 ( ۲۰۰۰ و ۵۰۰ ) رسم می‌کند.




**شکل ۱۵-۱۱**

### ۱۱-۳-۶ 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 (۰و۰) 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 خاصیت CurrentX و CurrentY در کادر پیغام دیگری به نمایش در می‌آید که نشان‌دهنده عملکرد متد Cls در تغییر موقعیت جاری مکان نماست.

```
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
```

### ۱۱-۳-۷ متد Print

تاکنون بارها از این متد استفاده کرده‌اید. این متد می‌تواند هرگونه اطلاعات اعم از متن، مقادیر عددی، مقادیر مربوط به خواص 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 در روی آن نمایش داده می‌شوند. outputlist اطلاعاتی است که توسط متد Print در روی شئ 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. استفاده از کاراکتر ; در متد Print باعث خواهد شد تا اطلاعات به‌صورت چسبیده به هم و بدون فاصله از هم نمایش داده شوند، اما استفاده از کاراکتر کاما ( , ) سبب می‌شود که اطلاعات بعدی با فاصله معینی از اطلاعات قبلی به نمایش درآیند و استفاده از آن‌ها اختیاری است. ( علامت ¦ در شکل کلی متد Print یک نماد به معنی استفاده از یکی از علایم « ; » و « , » می‌باشد و جزیی از پارامترهای متد Print نیست).

به‌عنوان مثال، به این رویه توجه کنید:

```
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
```

رویه قبل مقادیر مربوط به سه متغیر رشته‌ای را به‌صورت زیر نمایش خواهد داد:

```
Visual Basic 6
Visual          Basic    6
VisualBasic     6
```

در این‌جا ذکر این نکته لازم است که در صورت استفاده از کاراکتر « , » اطلاعات را با فاصله ۱۴ ستون از یکدیگر نمایش می‌دهد و عرض هر ستون با توجه به اندازه قلم محاسبه می‌شود.

برای نمایش فضاهای خالی توسط متد Print، می‌توانید از تابع Spc استفاده کنید. شکل کلی این تابع به این صورت است: Spc (n)

که در آن n یک مقدار عددی است که تعداد فاصله‌ها را معین می‌کند مثلاً دستور زیر قبل از نمایش کلمه BASIC، ۵ فضای خالی ایجاد می‌کند.

```
Print  Spc (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 نمایش‌کلمه 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. |

## ۸-۳-۱۱ متدهای TextWidth و TextHeight

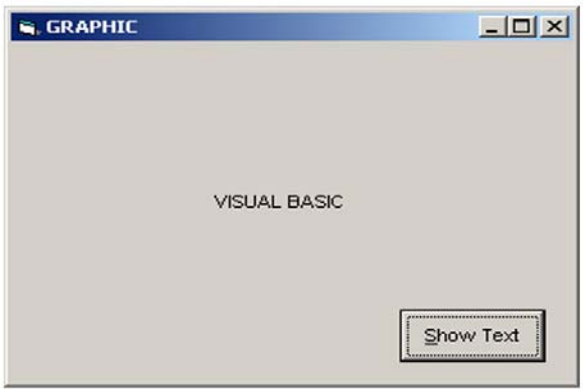
به‌وسیله این دو متد می‌توان اطلاعات نمایشی را به‌دست آورد. متد TextWidth عرض متن مورد نظر و متد TextHeight ارتفاع متن مورد نظر جهت نمایش را معین می‌کنند. شکل کلی این دو متد به‌صورت زیر است:

```
[ object. ] TextWidth ( string )
[ object. ] TextHeight ( string )
```

مقدار object اختیاری بوده و می‌تواند نام یک فرم، کنترل PictureBox و یا چاپگر باشد و 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 از دستورات بعدی استفاده می‌شود.

```
Private  Sub  cmdtext_Click(  )

        CurrentX  =  (ScaleWidth  -  TextWidth _
("VISUAL  BASIC"))  /  2

        CurrentY  =  (ScaleHeight  -  TextHeight _
("VISUAL  BASIC"))  /  2

        Print  "VISUAL  BASIC"

End  Sub
```

در نتیجه اجرای رویه قبل، خروجی برنامه مشابه شکل (۶-۱۱) خواهد بود:



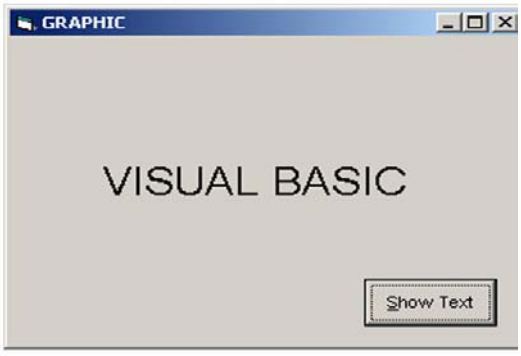

**شکل ۱۶-۱۱**

**نکته**

متدهای TextWidth و TextHeight محاسبات خود را بر اساس اندازه و نوع قلم در شیئی که نمایش اطلاعات در آن صورت می‌گیرد انجام می‌دهند. به‌عنوان مثال به رویه زیر و نتیجه اجرای آن در شکل ۱۷-۱۱ توجه کنید.

```
Private  Sub  cmdtext_Click(  )

        Form1.FontSize  =  20

        CurrentX  =  (ScaleWidth  -
TextWidth("VISUAL  BASIC"))  /  2

        CurrentY  =  (ScaleHeight  -
TextHeight("VISUAL  BASIC"))  /  2
```

```
        Print  "VISUAL  BASIC"

End  Sub
```

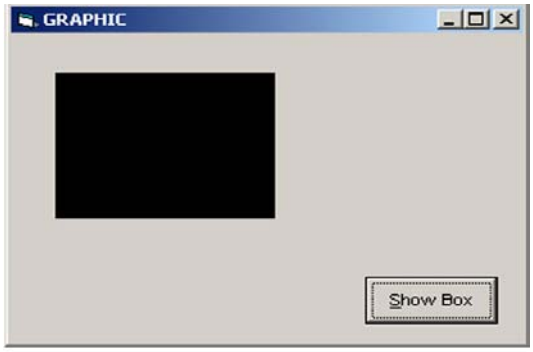



**شکل ۱۷-۱۱**

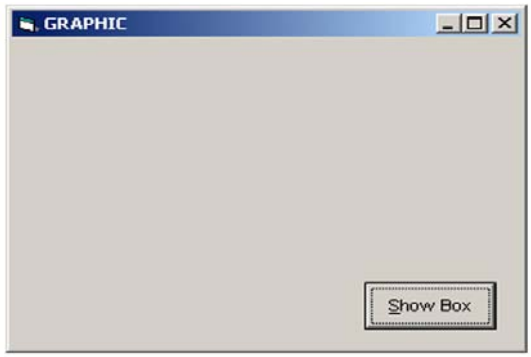
همان‌طور که در دو مثال اخیر مشاهده کردید به‌وسیله دو متد TextWidth و TextHeight و محاسبه عرض و ارتفاع متن و با کم کردن این مقادیر از عرض و ارتفاع فرم که به‌وسیله خواص ScaleWidth و ScaleHeight به‌دست می‌آیند و تقسیم مقدار به‌دست آمده بر عدد ۲، محل نمایش متن را به گونه‌ای که در وسط صفحه قرار گیرد محاسبه کردیم. 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 در وسط صفحه قرار می‌گیرد.

## ۹-۳-۱۱ خاصیت 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 و 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 نواروظیفه (Taskbar) کلیک کنید تا پنجره برنامه به حالت قبل بازگردد. اکنون فرم را به‌دقت مشاهده کنید. آیا اثری از شکل رسم شده در حالت قبل از به حداقل رسانی پنجره دیده می‌شود؟ پاسخ روشن است، خیر (شکل‌های ۱۸-۱۱ و ۱۹-۱۱).




**شکل ۱۸-۱۱**




**شکل ۱۹-۱۱**

در سیستم عامل ویندوز وقتی پنجره‌ای منتقل می‌شود یا پنجره‌ها روی یکدیگر قرار گرفته و یا تغییر اندازه پیدا می‌کنند ویندوز محتویات آن‌ها را به خاطر سپرده و در صورت نیاز محتویات هر پنجره را در اختیار آن قرار می‌دهد. تاکنون مشاهده کرده‌اید که در ویژوال بیسیک این امکان برای کنترل‌ها وجود دارد اما آیا برای سایر ترسیمات هم این نیاز برآورده می‌شود؟ بله.

فرم‌ها و بعضی از کنترل‌ها مانند PictureBox خاصیتی به نام AutoRedraw دارند که در حالت پیش‌فرض مقدار این خاصیت False است اگر این خاصیت 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 PictureBox نمایش داده می‌شوند.

در مثالی که در این قسمت انجام دادید خاصیت 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 دسک تاپ نمایش دهید. آیا در این حالت مستطیل رسم شده قابل مشاهده است؟ بله.

شکل کلی استفاده از این خاصیت به‌صورت زیر است:

[ object. ] AutoRedraw [ = Boolean ]

object یک مقدار اختیاری است که می‌تواند نام فرم، کنترل PictureBox باشد و در صورت عدم استفاده از آن، نام فرمی‌که فوکوس دارد استفاده می‌شود.

مقدار Boolean یک مقدار منطقی است که می‌تواند True و یا False باشد. در صورت عدم استفاده از مقدار Boolean، مقدار خاصیت بازگشت داده می‌شود. مقدار پیش‌فرض این خاصیت False است.

## ۱۰-۳-۱۱ خاصیت DrawMode

به‌وسیله این خاصیت می‌توان شکل ظاهری ترسیماتی ( از نظر رنگ ) را که به‌وسیله متدهای گرافیکی نظیر Circle و Line و Pset و غیره انجام می‌شوند تعیین کرد. شکل کلی نحوه استفاده از این خاصیت به‌صورت زیر است:

[ object. ] DrawMode [ = value ]

مقدار object اختیاری بوده و می‌تواند نام فرم، یا کنترل PictureBox باشد و در صورت عدم استفاده از آن نام فرمی‌که فوکوس دارد استفاده می‌شود. مقدار value نیز اختیاری بوده و شکل ظاهری ترسیمات را مشخص می‌کند. درصورت عدم استفاده از این مقدار، مقدار خاصیت بازگشت داده می‌شود.

مقادیری که بخش value می‌تواند کسب کند در جدول ۴-۱۱ آورده شده‌اند.

**جدول ۴-۱۱ مقادیر مربوط به خاصیت DrawMode**

| توضیح | ثابت عددی | ثابت رشته‌ای |
|---|---|---|
| رنگ سیاه | 1 | vbBlackness |
| عکس حالت 15-vbMergePen | 2 | vbNotMergePen |
| ترکیب رنگ بر اساس رنگ زمینه و رنگ معکوس pen | 3 | vbMaskNotPen |
| عکس حالت 13-CopyPen | 4 | vbNotCopyPen |
| ترکیب رنگ‌های عمومی با رنگpen و رنگ معکوس، رنگی که نمایش داده شده است. | 5 | vbMaskPenNot |
| رنگ معکوس، رنگی که نمایش داده شده است. | 6 | vbInvert |
| ترکیب رنگ‌ها بر اساس رنگ pen و یا رنگی که نمایش داده شد. | 7 | vbXorPen |
| عکس حالت 9-MaskPen | 8 | vbNotMaskPen |
| ترکیب رنگ‌های‌عمومی با رنگ pen و رنگی که نمایش داده شد. | 9 | vbMaskPen |

| ثابت رشته‌ای | ثابت عددی | توضیح |
|---|---|---|
| vbNotXorPen | 10 | عکس حالت 7-vbXorPen |
| vbNop | 11 | ترسیمی انجام نمی‌شود. |
| vbMergeNotePen | 12 | ترکیب رنگ معکوس، رنگ pen و رنگی‌که نمایش‌داده شده‌است. |
| vbCopyPen | 13 | به‌طور پیش‌فرض رنگ pen و اگر رنگ pen تعیین نشود رنگ ForeColor استفاده می‌شود. |
| vbMergePenNot | 14 | ترکیب رنگ pen و رنگ معکوس، رنگی‌که نمایش‌داده شده‌است. |
| vbMergePen | 15 | ترکیب رنگ pen و رنگی که نمایش داده شده است. |
| vbWhiteness | 16 | رنگ سفید |

توجه داشته باشید که در جدول ۴-۱۱منظور از رنگ pen، رنگی است که متد گرافیکی جهت رسم شکل گرافیکی از آن استفاده می‌کند. در واقع این خاصیت به ویژوال بیسیک می‌گوید که چگونه رنگ یک pixel نمایشی روی صفحه را تعیین کند؛ به‌عبارت دیگر ویژوال بیسیک 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 خاصیت ScaleMode تعیین می‌کند. به‌عنوان مثال به رویه زیر توجه کنید:

```
Private  Sub  cmdbox_Click( )
        Form1.DrawMode  =  vbInvert
        Form1.BackColor  =  vbRed
        Line  (500,  500)-(2000,  2500),  vbGreen,  BF
End  Sub
```

همان‌طور که در رویه ملاحظه می‌کنید باید یک مستطیل توپر به‌وسیله متد Line با رنگ سبز رسم شود اما چون مقدار خاصیت 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.

**توجه :** مقدار پیش‌فرض خاصیت 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 خاصیت ForeColor تعیین شده استفاده شود.

## ۱۱-۳-۱۱ خاصیت DrawStyle

به‌وسیله این خاصیت می‌توان نوع و حالت خطوط را در ترسیمات گرافیکی تعیین کرد. شکل کلی نحوه استفاده از این خاصیت به‌صورت زیر است:

[ object. ] DrawStyle [ = value ]

مقدار object اختیاری بوده و می‌تواند نام فرم، یا کنترل PictureBox باشد. در صورت عدم استفاده از آن، نام فرمی‌که فوکوس دارد استفاده می‌شود. مقدار value اختیاری بوده و نوع و حالت ترسیمات را مشخص می‌کند. در صورت عدم استفاده از این مقدار، مقدار خاصیت بازگشت داده می‌شود. مقادیری که value می‌تواند کسب کند در جدول ۵-۱۱ ارایه شده است.

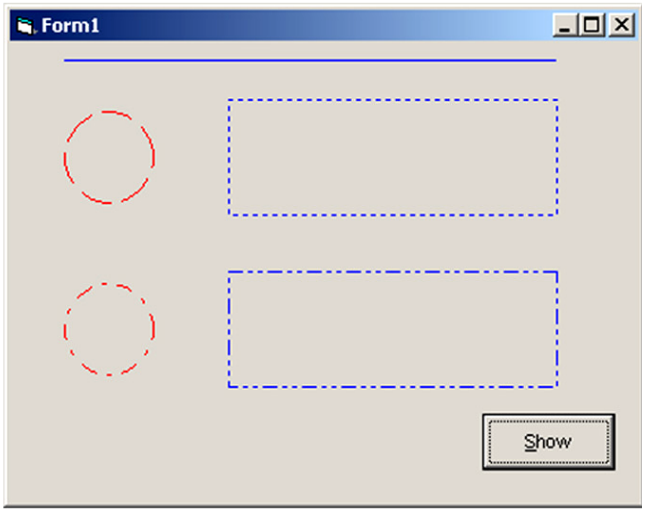
**جدول ۵-۱۱ مقادیر مربوط به خاصیت DrawStyle**

| ثابت رشته‌ای | ثابت عددی | حالت نمایشی |
|---|---|---|
| vbsolid | 0 | ———— |
| vbDash | 1 | —— —— |
| vbDot | 2 | ------ |
| vbDashDot | 3 | —— .—— . |
| vbDashDotDot | 4 | ——.. —— |
| vbInvisible | 5 | خط نامریی |
| vbInsideSolid | 6 | لبه بیرونی حاشیه بر لبه بیرونی شکل منطبق است |

در این رویه حالت‌های مختلف ترسیم را برای مقادیر صفر تا ۴ مشاهده می‌کنید (شکل ۲۰-۱۱).

```
Private  Sub  cmdshow_Click( )
        DrawStyle  =  vbSolid
        Line  (500,  150)-(500,  150),  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 ۲۰-۱۱**

## ۱۲-۳-۱۱ خاصیت DrawWidth

به‌وسیله این خاصیت می‌توانید ضخامت خطوط را برای ترسیمات گرافیکی معین کنید. شکل کلی نحوه استفاده از این خاصیت به‌صورت زیر است:

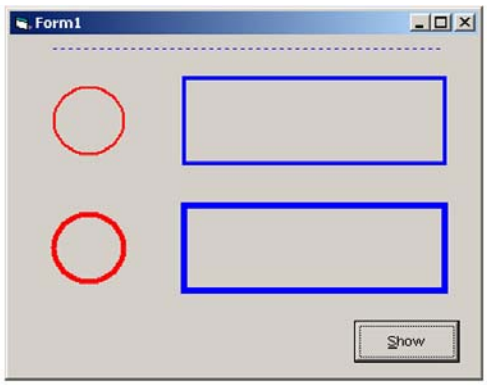
[ object. ] DrawWidth [ = size ]

مقدار object اختیاری بوده و می‌تواند نام فرم یا کنترل PictureBox باشد. در صورت عدم استفاده از آن، نام فرمی‌که فوکوس دارد استفاده می‌شود. مقدار size اختیاری بوده و یک عبارت عددی

است که اندازه قلم را برای ترسیمات معین می‌کند و در صورت عدم استفاده از این مقدار، مقدار خاصیت بازگشت داده می‌شود. مقدار size می‌تواند مقداری بین ۱ تا ۳۲۷۶۷ باشد.

در رویه زیر حالت‌های مختلف ترسیم را با اندازه‌های مختلف قلم مشاهده می‌کنید (شکل ۲۱-۱۱).

```
Private  Sub  cmdshow_Click(  )
        DrawStyle  =  vbDot
        DrawWidth  =  1
        Line  (500,  150)-(500,  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 ۲۱-۱۱**

**نکته**

در صورتی که خاصیت DrawWidth روی عدد بزرگ‌تر از یک تنظیم شود مقادیر ۱ تا ۴ برای خاصیت DrawStyle در زمان اجرای متدهای گرافیکی عملکردی از خود نشان نمی‌دهند و مانند مقدار vbSolid عمل می‌کنند.

## ۱۳-۳-۱۱ خاصیت FillStyle

به‌وسیله این خاصیت می‌توانید ترسیماتی نظیر دایره، بیضی و مستطیل را با حالت‌های مختلف پر کنید. شکل کلی نحوه استفاده از این خاصیت به‌صورت زیر است:

[ object. ] FillStyle [ = number ]

مقدار object اختیاری بوده و می‌تواند نام فرم یا کنترل PictureBox باشد. در صورت عدم استفاده از آن، نام فرمی‌که فوکوس دارد استفاده می‌شود. 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 خاصیت بازگشت داده می‌شود. مقادیر مجاز برای number در جدول ۶-۱۱ ارایه شده است:

**جدول ۶-۱۱ مقادیر مربوط به خاصیت FillStyle**

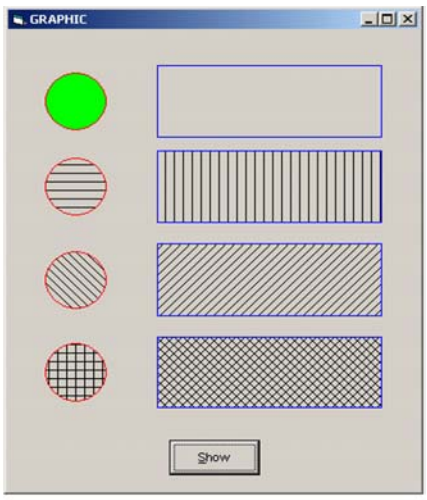
| ثابت رشته‌ای | ثابت عددی | توضیح |
|---|---|---|
| vbFsSolid | 0 | داخل شکل با رنگی که در خاصیت FillColor تعیین شده پر می‌شود. |
| vbFStransparent | 1 | داخل شکل با رنگ زمینه پر می‌شود. |
| vbHorizontalLine | 2 | داخل شکل با خطوط افقی پر می‌شود. |
| vbVerticalLine | 3 | داخل شکل با خطوط عمودی پر می‌شود. |
| vbUpwardDiagonal | 4 | داخل‌شکل‌با خطوط مایل پر می‌شود( از چپ به راست ). |
| vbDownwardDiagonal | 5 | داخل‌شکل‌با خطوط مایل پر می‌شود( از راست به چپ ). |
| vbCross | 6 | داخل‌شکل با خطوط عمودی و افقی پر می‌شود ( حالت شطرنجی ). |
| vbDiagonalCross | 7 | داخل شکل با خطوط عمودی و افقی مایل پر می‌شود ( حالت شطرنجی مایل ). |

در رویه زیر حالت‌های مختلف ترسیم را برای مقادیر متفاوتی از خاصیت FillStyle مشاهده می‌کنید.

```
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 رویه فوق پس از تعیین مقادیر مورد نظر برای خواص DrawStyle و DrawWidth، مقدار خاصیت FillColor روی سبز تنظیم شده است. خاصیت FillColor رنگ قلم را برای ترسیماتی که به‌وسیله خاصیت FillStyle ایجاد می‌شود معین می‌کند. به‌عنوان مثال در دایره اول چون مقدار

خاصیت FillStyle، صفر است دایره با رنگ FillColor یعنی سبز پر می‌شود. در مورد سایر مقادیر FillStyle نیز خاصیت FillColor رنگ خطوط عمودی، افقی، مایل و غیره که شکل را پر می‌کند تعیین می‌کند. نتیجه اجرای رویه فوق را می‌توانید در شکل ۲۲-۱۱ مشاهده کنید.




**شکل ۲۲-۱۱**

## ۱۱-٤ 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 باز می‌گرداند. شکل کلی این تابع به‌صورت زیر است:

QBColor ( color )

آرگومان color یک عدد از نوع صحیح است که براساس جدول۷-۱۱ قابل استفاده خواهند بود.

**جدول ۷-۱۱ مقادیر قابل استفاده برای آرگومان color**

| رنگ | مقدار عددی | رنگ | مقدار عددی |
|---|---|---|---|
| خاکستری | 8 | سیاه | 0 |
| آبی روشن | 9 | آبی | 1 |
| سبز روشن | 10 | سبز | 2 |
| فیروزه‌ای روشن | 11 | فیروزه‌ای | 3 |
| قرمز روشن | 12 | قرمز | 4 |

| رنگ | مقدار عددی | رنگ | مقدار عددی |
|---|---|---|---|
| بنفش روشن | 13 | بنفش | 5 |
| زرد روشن | 14 | زرد | 6 |
| سفید روشن | 15 | سفید | 7 |

به‌عنوان مثال به رویه زیر توجه کنید در این رویه با استفاده از تابع QBColor ۵۰۰۰ نقطه با رنگ‌های متفاوت نمایش داده خواهد شد البته این عمل زمانی روی می‌دهد که کاربر روی فرم عمل کلیک انجام دهد.

```
Private  Sub  Form_Click( )
      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
End  Sub
```

## ۵-۱۱ تابع RGB

در ویژوال بیسیک تابع دیگری به نام RGB وجود دارد که می‌تواند ترکیبات رنگی شما 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 ۸-۱۱ ارایه شده است.

**جدول ۸-۱۱ مقادیر سه رنگ اصلی برای رنگ‌های استاندارد**

| رنگ | مقدار آرگومان red | مقدار آرگومان green | مقدار آرگومان blue |
|---|---|---|---|
| سیاه | 0 | 0 | 0 |
| آبی | 0 | 0 | 255 |
| سبز | 0 | 255 | 0 |
| فیروزه‌ای | 0 | 255 | 255 |
| قرمز | 255 | 0 | 0 |
| بنفش | 255 | 0 | 255 |
| زرد | 255 | 255 | 0 |
| سفید | 255 | 255 | 255 |

به‌عنوان مثال رویه زیر پانصد دایره با ابعاد و مختصات و رنگ‌های تصادفی ایجاد می‌کند.

```
Private  Sub  Form_Click( )
      Dim  i  As  Integer
      Dim  XPos  As  Single,  YPos  As  Single
      DrawWidth  =  4
      Randomize
      For  i  =  1  To  1000
            XPos  =  Rnd  *  ScaleWidth
            YPos  =  Rnd  *  ScaleHeight
            Circle  (XPos,  YPos),  Rnd  *  800,  RGB(Rnd * _
255,  Rnd  *  255,  Rnd  *  255)
```

```
      Next  i
End  Sub
```

## ۱۱-۶ شئ چاپگر PRINTER OBJECT

تاکنون کلیه عملیاتی که انجام داده‌اید در روی فرم و صفحه نمایش انجام شده است اما گاهی اوقات لازم است تا اطلاعات مورد نیاز خود را به‌وسیله چاپگر روی کاغذ چاپ کنید. ویژوال بیسیک در این زمینه نیز امکانات لازم را مهیا کرده است. شما با استفاده از شئ چاپگر علاوه بر انجام عملیات چاپ می‌توانید به‌وسیله خواص این شئ عملیات چاپ را به نحو مناسبی مدیریت کنید. در این بخش به معرفی متدهای چاپ و معرفی خواص شئ چاپگر می‌پردازیم.

### ۱۱-۶-۱ متدهای چاپ

تا کنون متدهای مختلفی را جهت نمایش اطلاعات و ترسیمات آموختید. همان‌طور که در معرفی متدها در این فصل نیز توضیح دادیم می‌توانید به جای پارامتر object در متدهای معرفی شده، به جای نام یک فرم یا کنترل از شئ چاپگر استفاده کنید؛ بنابراین به آسانی می‌توانید از متدهای Circle، Line، PSet، Scale، TextHeight و TextWidth و Print استفاده کنید. فقط کافی است برای معرفی شئ چاپگر از کلمه Printer استفاده کنید. به‌عنوان مثال دستور زیر پیغام IN THE NAME OF GOD را روی کاغذ چاپ می‌کند.

```
Printer. Print  " IN THE NAME OF GOD. "
```

در این‌جا لازم است به بعضی از متدهای ویژه برای چاپ اطلاعات اشاره کنیم.

#### ۱۱-۶-۱-۱ متد EndDoc

این متد سبب می‌شود تا عملیات چاپ متوقف شده و تا زمانی که چاپگر آماده چاپ شود اطلاعات مربوط به چاپ در روی دیسک یا حافظه کامپیوتر ذخیره می‌شود. شکل کلی نحوه استفاده از این متد به‌صورت زیر است:

```
Printer. EndDoc
```

#### ۱۱-۶-۱-۲ 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
```

### ۱۱-۶-۱-۳ 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

## ۱۱-۶-۲ خواص شئ چاپگر

تنظیمات مربوط به چاپگرها نیز مانند اشیای دیگر به‌وسیله تعدادی از خاصیت‌ها قابل دست‌یابی و تغییر است. در این بخش مهم‌ترین خواص شئ چاپگر را مورد بررسی قرار می‌دهیم. بعضی از خواص نیز قبلاً در همین فصل توضیح داده شده‌اند مانند :CurrentY، DrawMode، DrawStyle ، CurrentX، FillStyle ، DrawWidth، FillColor، خواص مربوط به قلم‌ها ( Fonts ) و...

### ۱۱-۶-۲-۱ خاصیت ColorMode

به‌وسیله این خاصیت می‌توان نوع چاپگر را از نظر چاپ رنگی یا سیاه سفید تعیین کرد. شکل کلی نحوه استفاده از این خاصیت 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 ۹-۱۱ باشد. در صورت عدم استفاده از بخش value، مقدار فعلی خاصیت بازگشت داده خواهد شد.

**جدول ۹-۱۱ مقادیر مربوط به خاصیت colormode**

| توضیح | ثابت عددی | ثابت رشته‌ای |
|---|---|---|
| چاپ سیاه سفید | 1 | vbPRCMMonochrome |
| چاپ رنگی | 2 | vbPRCMColor |

**نکته**

استفاده از چاپ رنگی و یا سیاه سفید به امکانات چاپگر بستگی دارد.

### ۱۱-۶-۲-۲ خاصیت Copies

به‌وسیله این خاصیت می‌توانید تعداد نسخه‌هایی که چاپگر چاپ می‌گیرد تعیین کنید. شکل کلی نحوه استفاده از این خاصیت 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 خاصیت بازگشت داده خواهد شد.

### ۱۱-۶-۲-۳ خاصیت DeviceName

این خاصیت 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 وبه‌صورت پیش‌فرض نصب کرده‌باشید فرمان زیر نام چاپگر یعنی myprinter را نمایش می‌دهد.

Print Printer. DeviceName

### ۱۱-۶-۲-۴ خاصیت DriverName

این خاصیت 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

### ۱۱-۶-۲-۵ خاصیت Orientation

به‌وسیله این خاصیت می‌توانید جهت انجام عملیات چاپ را در روی صفحه کاغذ تعیین کنید.

عملیات چاپ می‌تواند به‌صورت portrait و landscape باشد. در شکل ۱۱-۲۳ تفاوت این دو حالت نمایش داده شده است.

**شکل ۱۱-۲۳**

شکل کلی نحوه استفاده از این خاصیت به‌صورت زیر است:

Printer. Orientation [ = value ]

value یک ثابت عددی یا رشته‌ای است که جهت چاپ اطلاعات را در روی کاغذ معین می‌کند. این مقدار می‌تواند یکی از مقادیر موجود در جدول ۱۱-۱۰ باشد. در صورت عدم استفاده از مقدار value، مقدار فعلی خاصیت بازگشت داده خواهد شد.

**جدول ۱۰-۱۱ مقادیر مربوط به خاصیت Orientation**

| توضیح | ثابت عددی | ثابت رشته‌ای |
|---|---|---|
| چاپ به‌صورت portrait | 1 | vbPRORPortrait |
| چاپ به‌صورت landscape | 2 | vbPRORLandscape |

### ۶-۲-۶-۱۱ خاصیت Page

این خاصیت شماره صفحه در حال چاپ را در اختیار برنامه قرار می‌دهد. شکل کلی نحوه استفاده از این خاصیت به این صورت است:

```
Printer.Page
```

### ۷-۲-۶-۱۱ خاصیت PaperSize

به‌وسیله این خاصیت می‌توانید نوع و ابعاد کاغذ چاپ را تنظیم کنید. شکل کلی نحوه استفاده از این خاصیت 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 ۱۱-۱۱ باشد. در صورت عدم استفاده از مقدار value، مقدار فعلی خاصیت بازگشت داده خواهد شد.

**جدول ۱۱-۱۱ مقادیر مربوط به اندازه کاغذ در خاصیت PageSize**

| ابعاد کاغذ | ثابت عددی | ثابت رشته‌ای |
|---|---|---|
| Inch ۸½ ×۱۱ | 1 | vbPRPSLetter |
| Inch ۸½ ×۱۱ ( اندازه کوچک ) | 2 | vbPRPSLetterSmall |
| Inch ۱۱ × ۱۷ | 3 | vbPRPSTabloid |
| Inch ۱۷ ×۱۱ | 4 | vbPRPSLedger |
| Inch ۸½ ×۱۴ | 5 | vbPRPSLegal |
| Inch ۵½ × ۸½ | 6 | vbPRPSStatement |
| Inch ۷½ × ۱۰½ | 7 | vbPRPSExecutive |
| A3 (۲۹۷×۴۲۰ mm) | 8 | vbPRPSA3 |
| A4 (۲۱۰×۲۹۷ mm) | 9 | vbPRPSA4 |
| A4 ( اندازه کوچک ) | 10 | vbPRPSA4Small |

| ابعاد کاغذ | ثابت عددی | ثابت رشته‌ای |
|---|---|---|
| A5 ( ۱۴۸×۲۱۰ mm) | 11 | vbPRPSA5 |

### ۱۱-۶-۲-۸ خاصیت Port

به‌وسیله این خاصیت می‌توان نام پورت مربوط به چاپگری را که چاپ به آن ارسال می‌شود به‌دست آورد. شکل کلی نحوه استفاده از این خاصیت به‌صورت زیر است:

```
Printer.Port
```

**نکته**

چاپگرها معمولاً از پورت‌های موازی LPT1 و LPT2 استفاده می‌کنند.

### ۱۱-۶-۲-۹ خاصیت PrintQuality

این خاصیت کیفیت وضوح چاپ را در چاپگر معین می‌کند. شکل کلی نحوه استفاده از این خاصیت 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 خاصیت بازگشت داده خواهد شد. مقدار value می‌تواند یکی از مقادیر موجود در جدول ۱۲-۱۱ باشد.

**جدول ۱۲-۱۱ مقادیر مربوط به کیفیت چاپ در خاصیت PrintQuality**

| توضیح | ثابت عددی | ثابت رشته‌ای |
|---|---|---|
| چاپ با کیفیت Draft | -1 | vbPRPQDraft |
| چاپ با کیفیت پایین | -2 | vbPRPQLow |
| چاپ با کیفیت متوسط | -3 | vbPRPQMedium |
| چاپ با کیفیت بالا | -4 | vbPRPQHigh |

## خلاصه مطالب

- ترسیمات گرافیکی را می‌توان در روی فرم یا کنترل PictureBox و یا چاپگر ایجاد کرد.
- به‌وسیله خاصیت ScaleMode می‌توان مقیاس را در محورهای مختصات تنظیم کرد.
- به‌وسیله خاصیت ScaleLeft و ScaleTop می‌توان مختصات نقطه مبنا را تنظیم کرد.
- خواص ScaleHeight و ScaleWidth مقیاس را می‌توان در سیستم مختصات فرم یا کنترل PictureBox تنظیم کرد.
- به‌وسیله خاصیت 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 مکان نما را در صفحه ترسیمات معین می‌کنند. خاصیت CurrentX موقعیت جاری مکان نما 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، مقدار متغیرها و خواص 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 خاصیت AutoRedraw می‌توان از حذف ترسیمات موجود در یک پنجره جلوگیری به عمل آورد.
- خاصیت DrawMode می‌تواند رنگ ترسیمات گرافیکی را تعیین کند.
- به‌وسیله خاصیت DrawStyle می‌توان نوع خطوط را در متدهای گرافیکی نظیر Line و Circle تعیین کرد.

- با استفاده از خاصیت DrawWidth می‌توان ضخامت خطوط و نقاط را در متد Line، Circle و PSet مشخص کرد.
- خاصیت FillStyle، نوع و حالت خطوطی که سطح یک شکل گرافیکی نظیر مستطیل، دایره و یا بیضی را پر می‌کند تعیین می‌کند.
- به‌وسیله خاصیت FillColor می‌توان رنگ مورد نظر را برای پوشاندن سطح یک شکل گرافیکی نظیر مستطیل، دایره و 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 خواص و متدهای این شئ، می‌توانید اطلاعات مورد نظر را به‌وسیله چاپگر روی کاغذ چاپ کرده و بر نحوه انجام عملیات چاپ نظارت کرد.

# آزمون پایانی

**۱- به‌وسیله کدام خاصیت می‌توان یکی از مقیاس‌های استاندارد را در ویژوال بیسیک انتخاب کرد؟**

۱- ScaleMode ۲- ScaleHeight
۳- ScaleWidth ۴- Scale

**۲- کدام گزینه برای رسم یک مستطیل توپر توسط متد Line مناسب است؟**

۱- B ۲- BF ۳- F ۴- FB

**۳- کدام تابع در متد Print می‌تواند اطلاعات نمایشی را در ستون مشخصی نمایش دهد؟**

۱- Spc ۲- Point ۳- Tab ۴- Spcb

**۴- خروجی فرمان Circle (150,200) , 1000 ,,, 2 به صورت..... است.**

۱- بیضی افقی ۲- بیضی عمودی
۳- نیم‌دایره ۴- ربع دایره

**۵- به‌وسیله کدام خاصیت می‌توان ضخامت ترسیمات را تعیین کرد؟**

۱- DrawMode ۲- DrawStyle
۳- DrawWidth ۴- FillStyle

**۶- کدام متد می‌تواند عملیات چاپ را خاتمه دهد؟**

۱- EndDoc ۲- Port ۳- NewPage ۴- KillDoc

**۷- کدام تابع می‌تواند ترکیبات رنگی را بر اساس رنگ‌های اصلی ایجاد کند؟**

۱- QBColor ۲- BackColor ۳- RGB ۴- ColorMode

**۸- کدام خاصیت، رنگ مورد نظر جهت پر کردن یک مستطیل یا دایره را معین می‌کند؟**

۱- FillStyle ۲- FillColor
۳- ColorMode ۴- ForeColor

**۹- به‌وسیله کدام خاصیت موقعیت جاری مکان نما در صفحه ترسیمات در جهت محور افقی معین می‌شود؟**

۱- CurrentX ۲- CurrentY
۳- Point ۴- گزینه‌های ۱و۲ صحیح هستند.

**۱۰- به‌وسیله کدام خاصیت می‌توان نام راه‌انداز دستگاه چاپگر پیش‌فرض را به‌دست آورد؟**

۱- DeviceName ۲- DriverName

۳- Orientation ۴- ColorMode

## دستور کار آزمایشگاه

۱- رویه‌ای را بنویسید که با دریافت ضرایب معادله خط، آن را در روی یک فرم نمایش دهد (همراه با محورهای مختصات).

۲- پروژه‌ای از نوع Standard EXE 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.

## پاسخ پیش‌آزمون

| | | | |
|---|---|---|---|
| ۱ـ۴ | ۲ـ۱ | ۳ـ۲ | ۴ـ۲ |
| ۵ـ۳ | ۶ـ۲ | ۷ـ۴ | ۸ـ۱ |

## پاسخ آزمون پایانی

| | | | |
|---|---|---|---|
| ۱ـ۱ | ۲ـ۲ | ۳ـ۳ | ۴ـ۲ |
| ۵ـ۳ | ۶ـ۴ | ۷ـ۳ | ۸ـ۲ |
| ۹ـ۱ | ۱۰ـ۲ | | |

فصل دوازدهم

## هدف کلی

# نحوه استفاده از کنترل‌های ذاتی و ActiveX ویژوال بیسیک

| زمان (ساعت) | |
|---|---|
| نظری | عملی |
| ۵ | ۱۰ |

### هدفهای رفتاری ▼

**پس از مطالعه این فصل از فراگیر انتظار می‌رود که:**

۱- توانایی استفاده از کنترل‌های زیر را در زمان‌طراحی یک‌برنامه‌کاربردی داشته‌باشد:
ListBox , ComboBox , DriveListBox , DirListBox
FileListBox , Monthview , Dtpicker , Flatscrollbar
Hscrollbar , Vscrollbar , Image List , ImageCombo
MaskedEdit , RichTextBox , Slider , UpDown

۲- توانایی استفاده از رابط‌های گرافیکی MDI و SDI را داشته باشد و تفاوت‌های آن‌ها را بداند.

۳- توانایی استفاده از فرم‌های آماده را داشته باشد.

# پیش آزمون

**۱- به‌وسیله کدام‌یک از خواص زیر می‌توان مختصات نقطه مبنا را تغییر داد؟**

۱- ScaleLeft, ScaleMode ۲- ScaleTop, ScaleLeft
۳- ScaleTop, ScaleMode ۴- ScaleMode, ScaleHeight

**۲- فرمان circle (50,50), 20,,,, 0. 5 چه شکلی را رسم می‌کند؟**

۱- دایره ۲- نیم‌دایره ۳- بیضی افقی ۴- بیضی عمودی

**۳- به‌وسیله کدام متد می‌توان یک مستطیل را رسم کرد؟**

۱- Circle ۲- Pset ۳- Line ۴- Point

**۴- به‌وسیله کدام‌گزینه می‌توان موقعیت جاری مکان‌نما را در صفحه‌ترسیمات در جهت محور عمودی تعیین کرد؟**

۱- CurrentX ۲- CurrentX
۳- Point ۴- گزینه‌های ۱و۲ صحیح هستند.

**۵- کدام کاراکتر در متد Print سبب می‌شود اطلاعات نمایشی بدون فاصله در کنار هم قرار گیرند؟**

۱- ; ۲- , ۳- ! ۴- #

**۶- کدام گزینه در متد Line برای رسم یک مستطیل مناسب است؟**

۱- F ۲- BF ۳- FB ۴- B

**۷- به‌وسیله کدام متد می‌توان رنگ یک نقطه از تصویر را به‌دست آورد؟**

۱- PSet ۲- Point ۳- Tab ۴- Spc

**۸- به‌وسیله کدام خاصیت می‌توان نوع خطوط را برای متدهای گرافیکی تعیین کرد؟**

۱- DrawMode ۲- DrawStyle ۳- DrawWidth ۴- FillColor

**۹- کدام خاصیت کیفیت عملیات چاپ را در چاپگر معین می‌کند؟**

۱- PaperSize ۲- Orientation ۳- PrintQuality ۴- Copies

**۱۰- واحد اندازه‌گیری پیش‌فرض در سیستم مختصات ویژوال بیسیک عبارت است از:**

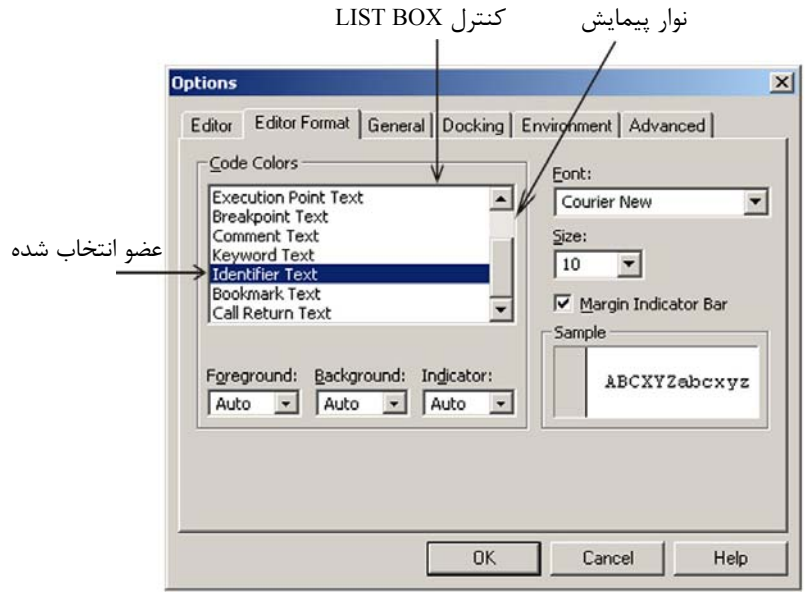
۱- سانتی‌متر ۲- اینچ ۳- pixel ۴- twip

## مقدمه

در فصل ۷ از جلد اول با بعضی از کنترل‌های ذاتی در ویژوال بیسیک آشنا شدید. در این فصل به معرفی سایر کنترل‌های ذاتی و بعضی از کنترل‌های ActiveX خواهیم پرداخت. در ضمن با نحوه طراحی و ساخت انواع منو (Menu Bar) در ویژوال بیسیک آشنا می‌شوید.

## ۱-۱۲ کنترل کادر لیست (ListBox)

یکی از کنترل‌های ذاتی در ویژوال بیسیک کنترل ListBox است. به‌وسیله این کنترل می‌توان لیستی از موارد مختلف را در رابطه با موضوع مربوطه، در اختیار کاربر قرار داد تا وی قادر به انتخاب یکی از موارد موردنظر خود باشد. این‌کار باعث می‌شود تا کاربر به‌جای نوشتن داده‌ها مثلاً در TextBox آن‌ها را به‌صورت آماده از یک لیست انتخاب کند. مزیت استفاده از این کنترل، سرعت و دقت بیشتر در ورود داده‌هاست. در شکل ۱-۱۲ شکل ظاهری یک کنترل ListBox را مشاهده می‌کنید. همان‌طور که در شکل می‌بینید یک کنترل ListBox مجموعه‌ای از اعضایی از پیش آماده است و کاربر به‌وسیله نوار پیمایش توانایی حرکت به بالا و پایین لیست و مشاهده اسامی موجود در آن را دارد. کاربر پس از پیدا کردن عضو مورد نظر در لیست، می‌تواند آن را به‌وسیله کلیک ماوس یا کلیدهای حرکت مکان‌نما در صفحه کلید انتخاب کند.




**شکل ۱-۱۲ اجزای مختلف در کنترل کادر لیست ListBox**

## ۱-۱-۱۲ خواص کنترل کادر لیست (ListBox)

بعضی از خواص این کنترل به‌صورت مشترک با سایر کنترل‌ها در فصل ۷ معرفی شده‌اند. در این‌جا فقط به ذکر خواص مخصوص این کنترل خواهیم پرداخت.

### ۱-۱-۱-۱۲ خاصیت List

این خاصیت اسامی اعضای موجود در لیست را نگهداری می‌کند. برای قرار دادن مقادیر مورد نظر در کنترل ListBox، پس از قرار دادن کنترل در روی فرم مقادیر مورد نظر را در این خاصیت بنویسید. مقادیر با همان ترتیبی که در این خاصیت نوشته می‌شوند در کنترل مشاهده می‌شوند.

به‌عنوان مثال می‌خواهیم لیستی از اسامی رنگ‌ها را ایجاد کنیم برای این کار عملیات زیر را به ترتیب انجام دهید :

۱- یک پروژه از نوع Standard EXE ایجاد کرده و سپس از جعبه ابزار، کنترل ListBox را روی فرم قرار دهید.

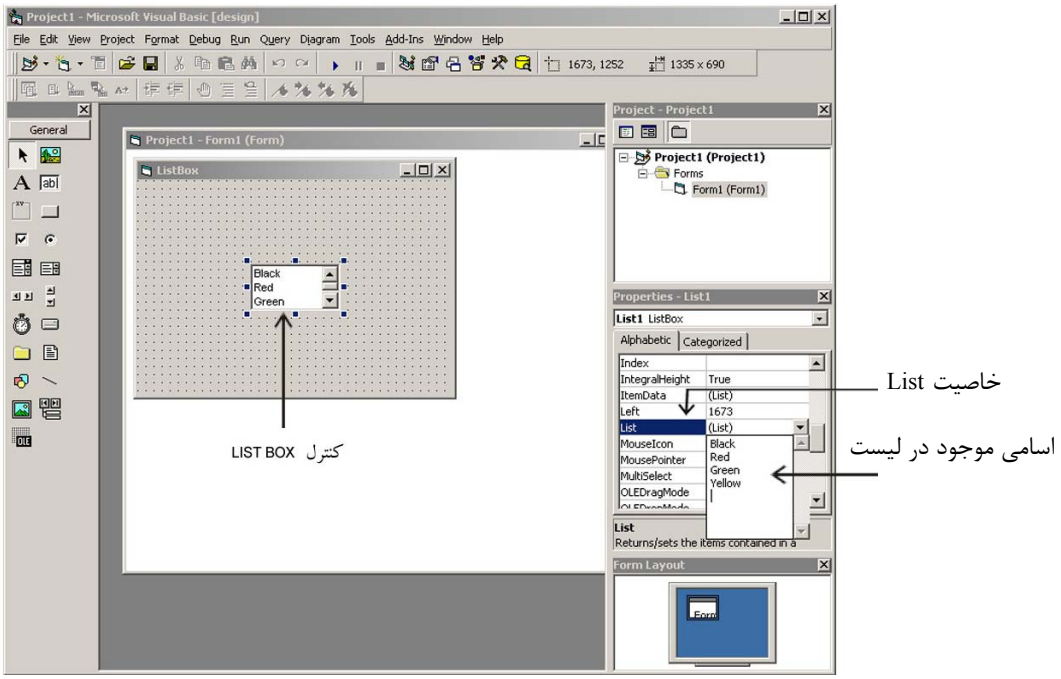
| توجه |
| --- |
| برای مشاهده محل قرارگیری کنترل ListBox در جعبه ابزار به فصل سوم شکل ۹-۳ مراجعه کنید. |

۲- در پنجره خواص کنترل ListBox را از لیست انتخاب کرده و روی خاصیت List آن را کلیک کنید.

۳- سپس روی دکمه در روبه‌روی خاصیت List کلیک کنید تا باز شود.

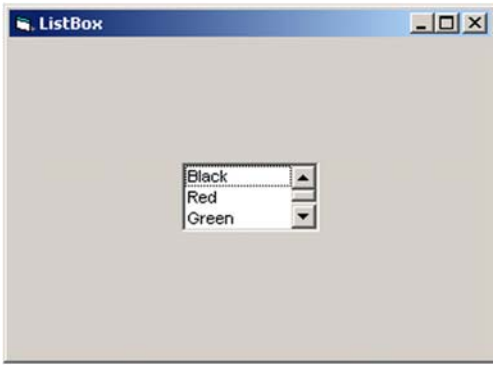
۴- کلمه Black را در لیست مربوطه بنویسید و سپس کلید ترکیبی Enter + Ctrl را فشار دهید.

۵- مانند مرحله ۴ کلمات Red، Green و Yellow را به لیست کنترل اضافه کنید (شکل ۲-۱۲).




**شکل ۲-۱۲ نحوه ورود اعضای یک کنترل ListBox به‌وسیله خاصیت List**

۶- پس از انجام مراحل فوق برنامه را اجرا کنید و به‌وسیله ماوس یا صفحه کلید روی اسامی موجود در کنترل ListBox حرکت کنید، نتیجه اجرای برنامه مطابق شکل ۳-۱۲ است.




**شکل ۳-۱۲**

۷- اجرای برنامه را به‌وسیله دکمه ■ در نوار ابزار پایان دهید.

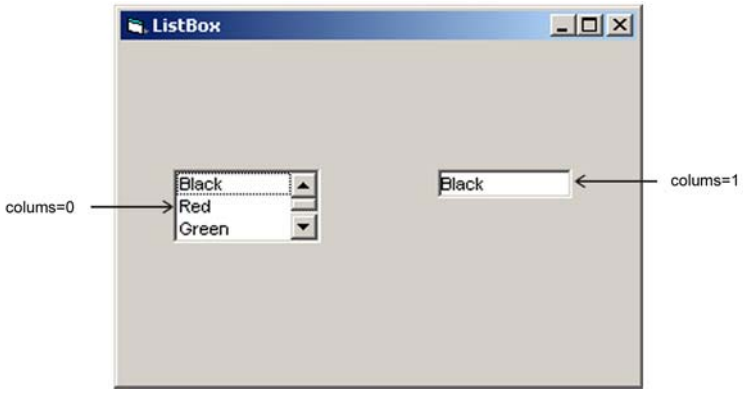
### ۱۲-۱-۱-۲ خاصیت Columns

به‌وسیله این خاصیت می‌توان تعداد ستون‌های موجود در کنترل را تعیین کرد اگر مقدار این خاصیت صفر باشد کنترل دارای یک لیست و در صورتی که مقدار آن ۱ یا بزرگ‌تر از یک باشد اعضای تشکیل‌دهنده لیست به‌صورت چند ستونی نمایش داده می‌شوند.

برای تنظیم کنترل در این حالت باید ابعاد کنترل از ابعاد لازم جهت نمایش تمامی اعضا کوچک‌تر باشد تا حالت چند ستونی مشاهده شود.

به‌عنوان مثال یک کنترل ListBox دیگر مانند شکل ۴-۱۲ در روی فرم مثال قبل اضافه کنید. خاصیت List این کنترل را مانند کنترل قبل تنظیم کنید.

خاصیت Columns کنترل ListBox دوم را روی مقدار ۱ تنظیم کنید. برنامه را اجرا کرده و در بین اعضای موجود در دو لیست حرکت کنید. تفاوت دو کنترل کاملاً قابل مشاهده است.




**شکل ۴-۱۲**

### ۱۲-۱-۱-۳ خاصیت MultiSelect

به‌وسیله این خاصیت می‌توانید نحوه انتخاب اعضای موجود در کنترل ListBox را تعیین کنید. این خاصیت می‌تواند یکی از سه مقدار زیر را کسب کند.

اگر مقدار این خاصیت 0-None باشد کاربر می‌تواند فقط یکی از اعضای موجود در کنترل را به‌وسیله کلیک ماوس یا کلیدهای مکان‌نما انتخاب کند.

اگر مقدار این خاصیت 1-Simple باشد کاربر می‌تواند به‌وسیله کلید spacebar یا کلیک ماوس چند عضو را در کنترل انتخاب کند.

اگر مقدار خاصیت فوق روی مقدار 2-Extended تنظیم شود کاربر می‌تواند به‌وسیله پایین نگه داشتن کلید Shift و به‌طور هم‌زمان کلیک ماوس و یا فشردن کلیدهای حرکت مکان‌نما در صفحه

کلید، اعضای مجاور را هم به‌صورت گروهی انتخاب کند یا با فشردن هم‌زمان کلید Ctrl و کلیک ماوس اعضای غیر مجاور را به‌صورت گروهی انتخاب کند. برای تشخیص بهتر تفاوت‌های این سه مقدار در خاصیت MultiSelect به مثال زیر توجه کنید :

در مثال قبل کنترل List1 را در روی فرم انتخاب کنید سپس در پنجره خواص خاصیت MultiSelect مربوط به این کنترل را بیابید و مقدار آن را روی 0-None تنظیم کنید سپس برنامه را اجرا کنید.

در مثال قبل عملیات زیر را به ترتیب انجام دهید :

۱- روی کنترل List1 در روی فرم کلیک کنید.

۲- در پنجره خواص خاصیت MultiSelect را برای کنترل مزبور پیدا کنید و مقدار آن‌را روی 0-None تنظیم کنید.

۳- برنامه را اجرا کرده و روی کلمه Red و بعد روی کلمه Yellow کلیک کنید.

۴- مرحله ۳ را با کلیدهای حرکت مکان‌نما انجام دهید. همان‌طور که می‌بینید در هر لحظه فقط یک انتخاب صورت می‌گیرد. اجرای برنامه را متوقف کنید.

۵- اکنون مقدار خاصیت MultiSelect را روی مقدار 1-Simple تنظیم کنید و برنامه را مجدداً اجرا کنید.

۶- در کنترل List1 روی کلمه Red و بعد روی Yellow کلیک کنید.

۷- مرحله ۶ را با کلیدهای spacebar و کلیدهای حرکت مکان‌نما انجام دهید. همان‌طور که مشاهده می‌کنید در هر لحظه می‌توانید چندین انتخاب داشته باشید.

۸- حالا مقدار خاصیت MultiSelect را روی مقدار 2-Extended تنظیم کنید و برنامه را مجدداً اجرا کنید.

۹- در کنترل List1 روی کلمه Black کلیک کنید سپس کلید Shift را پایین نگه دارید و هم‌زمان روی کلمه Green کلیک کنید. چه اتفاقی روی می‌دهد؟ تمام اعضای موجود بین Black و Green انتخاب می‌شوند.

۱۰- اکنون روی کلمه Yellow کلیک کنید و بعد کلید Ctrl را پایین نگه دارید و به‌طور هم‌زمان ابتدا روی Black و بعد روی Green کلیک کنید همان‌طور که مشاهده کردید سه عضو غیر مجاور در کنترل به‌طور هم‌زمان انتخاب می‌شوند.

۱۱- اجرای برنامه را متوقف سازید و به محیط طراحی بازگردید.

#### ۴-۱-۱-۱۲ خاصیت Sorted

این خاصیت از نوع منطقی است و در صورتی که مقدار آن روی True تنظیم شود اعضای

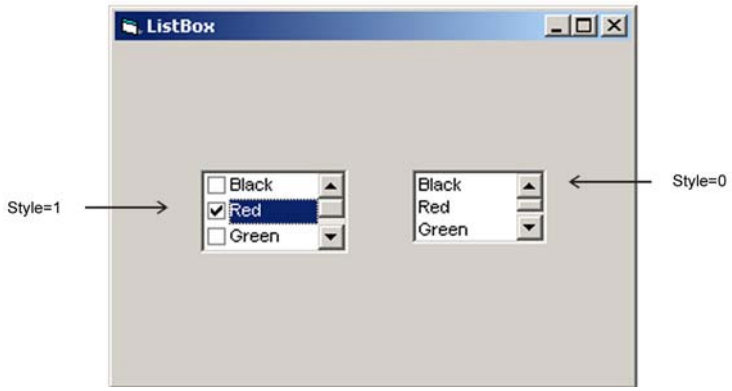
موجود در کنترل را به‌صورت صعودی مرتب می‌کند در غیر این صورت (یعنی تنظیم خاصیت روی False) اعضای موجود در کنترل با همان ترتیبی که در خاصیت List قرار گرفته‌اند در زمان اجرا دیده می‌شوند. به‌عنوان نمونه در مثال قبل روی کنترل List1 در روی فرم کلیک کنید، سپس در پنجره خواص خاصیت Sorted را برای این کنترل بیابید و مقدار آن را روی True تنظیم کنید. در پنجره طراحی فرم دقت کنید آیا تفاوتی به‌وجود آمده است؟

اکنون برنامه را اجرا کنید آیا تفاوتی دیده می‌شود؟ همان‌طور که می‌بینید اعضای موجود در کنترل بر اساس حروف الفبا مرتب شده‌اند.

### ۵-۱-۱-۱۲ خاصیت Style

این خاصیت را می‌توان روی دو مقدار تنظیم‌کرد. مقدار 0-Standard که اعضای‌کنترل ListBox را مشابه آن‌چه تا کنون گفته شد نمایش می‌دهد. اما اگر این خاصیت روی مقدار 1-CheckBox تنظیم کنید هر یک از اعضا به شکل یک کنترل CheckBox در داخل کنترل دیده می‌شوند.

دو حالت فوق را می‌توانید در شکل ۵-۱۲ مشاهده کنید.




**شکل ۵-۱۲**

در جدول ۱-۱۲ مقادیر مختلف را برای این خاصیت مشاهده می‌کنید.

**جدول ۱-۱۲**

| ثابت رشته‌ای | ثابت عددی | توضیح |
|---|---|---|
| vbListBoxStandard | 0 | اعضا با حالت استاندارد نمایش داده می‌شوند. |
| vbListBoxCheckbox | 1 | اعضا به‌صورت کنترل Checkbox نمایش داده می‌شوند. |

### ۱۲-۱-۱-۶ خاصیت Text

این خاصیت را نمی‌توان در پنجره خواص مشاهده کرد و از طریق کد نویسی قابل دسترس است. این خاصیت مقدار عضوی را که در کنترل ListBox انتخاب شده است نگهداری می‌کند. به‌عنوان مثال اگر در برنامه مثال قبل کاربر کلمه Red را انتخاب کند و سپس دستور زیر اجرا شود کلمه Red در روی فرم نمایش داده می‌شود.

```
Print  List1. text
```

شکل کلی نحوه استفاده از این خاصیت به‌صورت زیر است:

```
نام کنترل ListBox . Text [ = string ]
```

استفاده از string اختیاری بوده و در صورت استفاده از آن، عضو پیش‌فرض در کنترل تعیین می‌شود. string یک عبارت رشته‌ای است که نام یکی از اعضای موجود در کنترل می‌باشد.

### ۱۲-۱-۱-۷ خاصیت ListIndex

این خاصیت شماره اندیس عضوی را که در کنترل ListBox انتخاب شده است نگهداری می‌کند شماره اندیس‌ها از صفر شروع می‌شوند. شکل کلی نحوه استفاده از این خاصیت به‌صورت زیر است:

```
نام کنترل ListBox . ListIndex [ = index ]
```

مقدار index یک عبارت عددی است که می‌تواند عضو پیش‌فرض را در کنترل مشخص کند.

استفاده از این بخش اختیاری بوده و در صورت عدم استفاده از آن شماره اندیس عضو انتخاب شده بازگشت داده می‌شود. این خاصیت را نمی‌توان در پنجره خواص مشاهده کرد.

### ۱۲-۱-۱-۸ خاصیت ListCount

به‌وسیله این خاصیت می‌توان تعداد اعضای موجود در کنترل ListBox را به‌دست آورد. شکل کلی نحوه استفاده از این خاصیت به‌صورت زیر است:

```
نام کنترل ListBox . ListCount
```

این خاصیت را نمی‌توان در پنجره خواص مشاهده کرد.

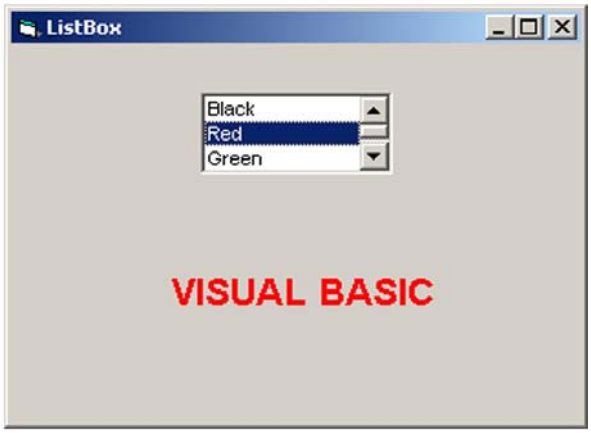
### ۱۲-۱-۱-۹ خاصیت Selected

به‌وسیله این خاصیت می‌توان از انتخاب یک عضو در کنترل ListBox مطلع شد. شکل کلی نحوه استفاده از این خاصیت به‌صورت زیر است:

```
نام کنترل ListBox . Selected ( index ) [ = boolean ]
```

استفاده از بخش Boolean اختیاری است و شما می‌توانید از مقدار True یا False استفاده کنید در صورتی‌که مقدار این خاصیت True باشد عضوی که شماره اندیس آن (index) در خاصیت Selected معین شده است به‌وسیله کاربر انتخاب شده است و در غیر این صورت کاربر آن را انتخاب نکرده است. در صورت عدم استفاده از بخش boolean، مقدار خاصیت Selected برای عضو مورد نظر بازگشت داده می‌شود (index یک عدد از نوع صحیح است). این خاصیت را نمی‌توان در پنجره خواص مشاهده کرد.

**مثال :** می‌خواهیم پروژه‌ای از نوع Standard EXE ایجاد کنیم که به‌وسیله یک کنترل ListBox بتوان رنگ عبارت VISUAL BASIC را در روی فرم مربوطه تغییر داد. در ضمن رنگ پیش‌فرض قرمز باشد و در صورت انتخاب یک رنگ، نام رنگ، شماره اندیس آن و تعداد اعضای موجود در کنترل نمایش داده شوند و در صورت انتخاب رنگ سیاه در لیست، برنامه خاتمه یابد. فرم برنامه را مطابق شکل ۶-۱۲ طراحی کنید.




**شکل ۶-۱۲**

پس از طراحی شکل ظاهری فرم، به تنظیم رویدادها می‌پردازیم. ابتدا رنگ پیش‌فرض را با استفاده از خاصیت Text کنترل در رویداد Load فرم تنظیم کنید برای این کار رویداد Load فرم را مطابق شکل زیر تنظیم کنید.

```
Private Sub Form_Load()

lstcolor.Text = "Red"

End Sub
```

برای آن که سایر نیازهای برنامه در زمان انتخاب یک رنگ برآورده شود رویداد Click کنترل

ListBox را به این صورت تنظیم کنید.

```
Private Sub cmdfont_Click()
    dlg.DialogTitle = "MyFont"
    dlg.Flags = cdlCFBoth + cdlCFEffects
    dlg.Color = vbBlack
    dlg.FontSize = 20
    dlg.FontName = "Arial"
    lbltext.FontName = dlg.FontName
    lbltext.FontSize = dlg.FontSize
    lbltext.ForeColor = dlg.Color

    dlg.ShowFont

    lbltext.FontName = dlg.FontName
    lbltext.FontSize = dlg.FontSize
    lbltext.FontBold = dlg.FontBold
    lbltext.ForeColor = dlg.Color
    lbltext.FontItalic = dlg.FontItalic
    lbltext.FontStrikethru = dlg.FontStrikethru
    lbltext.FontUnderline = dlg.FontUnderline
End Sub

Private Sub lstcolor_Click()
    If lstcolor.Selected(0) = True Then End
    Select Case lstcolor.ListIndex
        Case 1
            lbltext.ForeColor = vbRed
        Case 2
            lbltext.ForeColor = vbGreen
        Case 3
            lbltext.ForeColor = vbYellow
    End Select
    Print lstcolor.ListIndex
    Print lstcolor.ListCount
End Sub
```

همان‌طور که در رویه فوق مشاهده می‌کنید ابتدا به‌وسیله یک If مقدار خاصیت Selected کنترل بررسی می‌شود اگر مقدار این خاصیت برای رنگ سیاه که شماره اندیس آن صفر است True باشد به معنی انتخاب این رنگ خواهد بود در نتیجه مقایسه درست بوده و برنامه خاتمه می‌یابد اما در صورت انتخاب رنگ‌های دیگر بلاک Select Case اجرا می‌شود و با توجه به مقدار خاصیت ListIndex که برای رنگ Red، ۱ برای رنگ Green، ۲ و برای رنگ yellow، ۳ می‌باشد رنگ قلم را در کنترل برچسب تعیین می‌کند.

شماره اندیس‌ها و تعداد اعضا نیز به‌وسیله دو متد Print نمایش داده می‌شوند. برنامه را اجرا کرده و روی گزینه‌های مختلف در کنترل لیست کلیک کنید آیا عبارت VISUAL BASIC تغییر رنگ می‌دهد. در پایان روی رنگ سیاه Black کلیک کنید تا برنامه خاتمه یابد.

## ۱۲-۱-۲ متدهای کنترل ListBox

این کنترل دارای سه متد معروف AddItem، RemoveItem و Clear است که به توضیح جداگانه هر یک می‌پردازیم :

### ۱۲-۱-۲-۱ متد AddItem

این متد می‌تواند اعضای مورد نظر را به کنترل ListBox اضافه کند. شکل کلی نحوه استفاده از آن به‌صورت زیر است :

نام کنترل List Box . AddItem item

در واقع item نام عضوی است که به کنترل اضافه می‌شود. مثلاً در پروژه مثال قبل می‌توانید اسامی رنگ‌ها را در زمان اجرای برنامه و در رویداد Load فرم به‌وسیله این متد به کنترل اضافه کنید. در ادامه این رویه را مشاهده می‌کنید:

```
Private Sub Form_Load()
    lstcolor.AddItem "Black"
    lstcolor.AddItem "Red"
    lstcolor.AddItem "Green"
    lstcolor.Text = "Red"
End Sub
```

### ۱۲-۱-۲-۲ متد Clear

این متد تمام اعضای موجود در کنترل List Box را حذف می‌کند. شکل کلی نحوه استفاده از این متد به این صورت است:

نام کنترل ListBox . Clear

### ۱۲-۱-۲-۳متد RemoveItem

به‌وسیله این متد می‌توانید هر یک از اعضای موجود در کنترل ListBox را حذف کنید. شکل کلی نحوه استفاده از این متد به این صورت است:

نام کنترل ListBox . Remove Item (index )

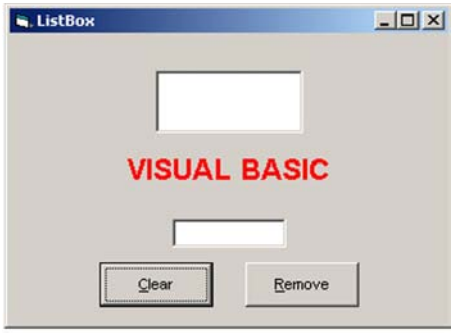
index یک عدد صحیح است که شماره اندیس عضو مورد نظر را در کنترل معین می‌کند.

همان‌طور که قبلاً هم اشاره شد شماره اندیس‌ها از صفر شروع می‌شوند.

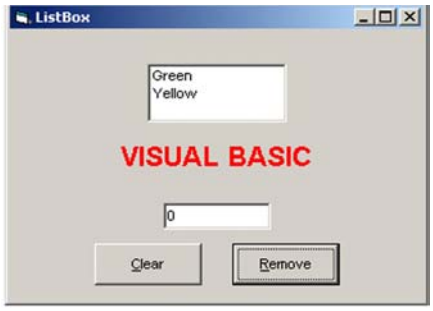
**مثال:** در پروژه مثال قبل دو کنترل دکمه فرمان با عنوان Clear و Remove و یک کنترل TextBox اضافه کنید. سپس این رویه‌ها را در پروژه کد نویسی کنید.

```
Private Sub cmdclear_Click()
    lstcolor.Clear
End Sub


Private Sub cmdremove_Click()
    If Val(txtindex.Text)>= 0 Then lstcolor.RemoveItem _
(txtindex.Text)
End Sub
```

رویداد اول کاملاً واضح است و در رویداد دوم با توجه به شماره اندیسی که کاربر در کنترل TextBox می‌نویسد عضو متناظر با آن اندیس حذف خواهد شد. شکل‌های ۷-۱۲ و ۸-۱۲ نحوه اجرای برنامه را پس از فشردن دکمه فرمان Clear و Remove نشان می‌دهند.




**شکل ۷-۱۲**




**شکل ۸-۱۲**

### ۱۲-۱-۳ رویدادهای کنترل ListBox

رویدادهای این‌کنترل‌نظیر Click، Dblclick، Gotfocus و LostFocus در فصل۷ در رویدادهای مشترک کنترل‌ها توضیح داده شده‌اند.

## ۱۲-۲ کنترل ComboBox

در ویژوال بیسیک علاوه بر کنترل ListBox کنترل دیگری وجود دارد که به شما اجازه می‌دهد انواع دیگری از لیست‌ها را طراحی کنید. کنترل ComboBox علاوه بر ایجاد یک لیست آماده از داده‌ها می‌تواند اجازه ورود داده‌هایی را که در لیست وجود ندارند نیز فراهم کند.

### ۱۲-۲-۱ خواص کنترل ComboBox

این کنترل نیز مانند سایر کنترل‌ها علاوه بر خواص مشترک، خواص ویژه‌ای نیز دارد که به توضیح هر یک می‌پردازیم.

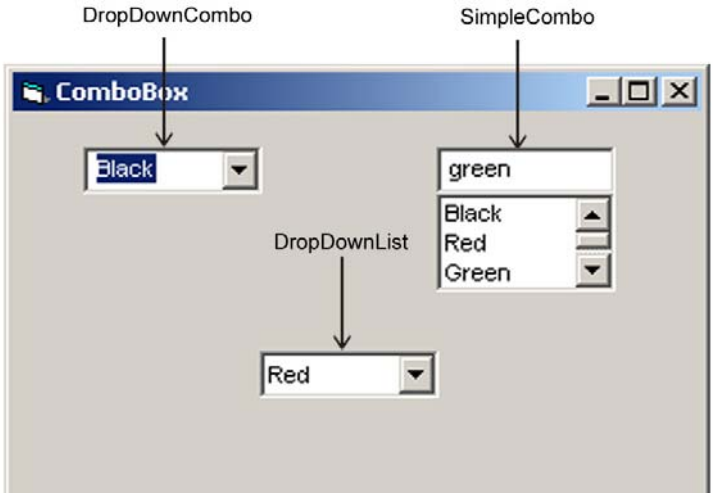
#### ۱۲-۲-۱-۱ خاصیت Style

مهم‌ترین خاصیت کنترل ComboBox، خاصیت Style است. به‌وسیله این خاصیت می‌توان سه نوع مختلف از این کنترل را ایجاد کرد. در جدول ۱۲-۲ مقادیر مربوط به این خاصیت را مشاهده کنید.

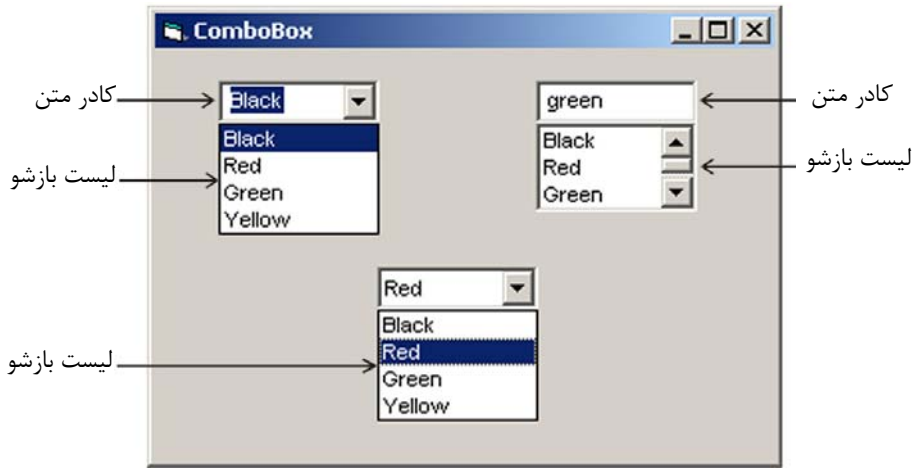
**جدول ۱۲-۲ مقادیر مربوط به خاصیت Style**

| ثابت رشته‌ای | ثابت عددی | توضیح |
|---|---|---|
| Vbcombo DropDown | 0 | مقدار پیش‌فرض است و یک لیست باز شو به همراه یک TextBox ایجاد می‌کند. |
| Vbcombo simple | 1 | یک لیست combo ساده که شامل یک لیست ساده به‌همراه TextBox است. |
| Vbcombo Drop-Downlist | 2 | کاربر فقط یک لیست به‌صورت باز شو ایجاد می‌کند. |

در شکل ۱۲-۹ و ۱۲-۱۰ کنترل ComboBox در حالت‌های مختلف دیده می‌شوند.




**شکل ۹-۱۲ انواع مختلف ComboBox**




**شکل ۱۰-۱۲ اجزای تشکیل دهنده کنترل ComboBox**

همان‌طور که در شکل ۱۰-۱۲ مشاهده می‌کنید نوع اول ComboBox از یک لیست باز شو که به‌وسیله دکمه قابل مشاهده است و از یک کادر متن که کاربر می‌تواند به‌طور مستقل اطلاعات خود را در آن بنویسد، تشکیل شده است.

نوع دوم از یک لیست ساده و یک کادر متن تشکیل می‌شود و نوع سوم فقط از یک لیست باز شو که به‌وسیله دکمه قابل مشاهده است تشکیل می‌شود.

**تمرین :** روی یک فرم سه کنترل ComboBox با مقادیر مختلف برای خاصیت style ایجاد کنید و در خاصیت List هر یک اسامی دلخواهی بنویسید سپس موارد فوق را در رابطه با آن‌ها تحقیق کنید.

### ۱۲-۲-۱-۲ خاصیت Locked

این خاصیت در رابطه با انواع DropDownCombo و SimpleCombo قابل استفاده است و در صورتی که مقدار آن روی True تنظیم شده باشد کاربر قادر به ورود یا ویرایش داده و اعضای کنترل نخواهد بود. مقدار پیش‌فرض این خاصیت False است. این خاصیت هم در پنجره خواص و هم به‌وسیله کدنویسی قابل تنظیم است.

### ۱۲-۲-۱-۳ خاصیت Text

این خاصیت در رابطه با انواع DropDownCombo و SimpleCombo قابل استفاده است و امکان تغییر مقدار این خاصیت برای نوع DropDownList امکان‌پذیر نیست. به‌وسیله این خاصیت می‌توان مقدار پیش‌فرض را در کنترل تعیین کرد. علاوه بر این شما می‌توانید از این خاصیت به عضو انتخاب شده در کنترل نیز دست‌یابی پیدا کنید. در این مورد می‌توانید از خاصیت Text در هر سه نوع کنترل ComboBox استفاده کنید. شکل کلی نحوه استفاده از این خاصیت به‌صورت زیر است :

نام کنترل ComboBox . Text [ = string ]

استفاده از string اختیاری بوده و در صورت استفاده از آن عضو پیش‌فرض در کنترل تعیین می‌شود. string یک عبارت رشته‌ای است که نام یکی از اعضای موجود در کنترل است.

| نکته |
| --- |
| در صورتی‌که بخواهید مقدار پیش‌فرض را در کنترل نوع DropDownList انتخاب کنید می‌توانید از خاصیت ListIndex استفاده کنید. |

### ۱۲-۲-۱-۴ خاصیت List

این خاصیت اعضای مربوط به کنترل را در خود نگهداری می‌کند. شما می‌توانید با انتخاب این خاصیت در پنجره خواص و نوشتن موارد مورد نیازتان در این خاصیت، کنترل خود را آماده کنید. توجه داشته باشید پس از نوشتن نام هر یک از اعضا در این خاصیت، کلید ترکیبی Ctrl + Enter را بفشارید سپس نام عضو بعدی را بنویسید.

### ۱۲-۲-۱-۵ خاصیت Sorted

این خاصیت از نوع منطقی بوده و در صورتی که مقدار آن روی True تنظیم شود اعضای

موجود درکنترل رابه‌صورت صعودی مرتب می‌کند در غیراین‌صورت (یعنی تنظیم خاصیت روی False) اعضای موجود در کنترل با همان ترتیبی که در خاصیت List قرار گرفته‌اند در زمان اجرا دیده می‌شوند.

#### ۱۲-۲-۱-۶ خاصیت ListIndex

این خاصیت شماره اندیس عضوی را که در کنترل ComboBox انتخاب شده است نگهداری می‌کند. شماره اندیس‌ها از صفر شروع می‌شوند. شکل کلی نحوه استفاده از این خاصیت به‌صورت زیر است:

نام کنترل ComboBox . ListIndex [ = index ]

مقدار index یک عبارت عددی است که می‌تواند عضو پیش‌فرض را در کنترل مشخص کند استفاده از این بخش اختیاری بوده و در صورت عدم استفاده از آن شماره اندیس عضو انتخاب شده بازگردانده می‌شود. از این خاصیت در پنجره خواص نمی‌توانید استفاده کنید.

#### ۱۲-۲-۱-۷ خاصیت ListCount

به‌وسیله این خاصیت می‌توانید تعداد اعضای یک کنترل ComboBox را به‌دست آورید. این خاصیت در پنجره خواص قابل استفاده نیست و شکل کلی نحوه استفاده از این خاصیت به این صورت است:

نام کنترل ComboBox . ListCount

### ۱۲-۲-۲ متدهای کنترل ComboBox

این کنترل دارای سه متد AddItem ، RemoveItem و Clear است که به توضیح هر یک از آن‌ها می‌پردازیم.

#### ۱۲-۲-۲-۱ متد AddItem

به‌وسیله این متد می‌توان اعضای مورد نظر خود را به کنترل اضافه کرد. شکل کلی نحوه استفاده از این متد به‌صورت زیر است:

نام کنترل ComboBox . AddItemitem

item نام عضوی است که به کنترل ComboBox اضافه می‌شود. مثلاً فرمان‌های زیر می‌تواند دو عضو جدید به کنترل مربوطه اضافه کند.

```
Cbocolor. Add Item    " Red "
```

```
Cbocolor. Add Item    " Green "
```

### ۱۲-۲-۲-۲ متد Clear

این متد می‌تواند تمام اعضای موجود در کنترل را حذف کند. شکل کلی نحوه استفاده از این متد به‌صورت زیر است:

نام کنترل ComboBox . Clear

مثلاً فرمان زیر تمام اعضای کنترل مربوطه را حذف می‌کند.

```
Cbocolor. Clear
```

### ۱۲-۲-۲-۳ متد RemoveItem

به‌وسیله این متد می‌توان هر یک از اعضای موجود در کنترل را حذف کرد. شکل کلی نحوه استفاده از این متد به‌صورت زیر است:

نام کنترل ComboBox . RemoveItem (index )

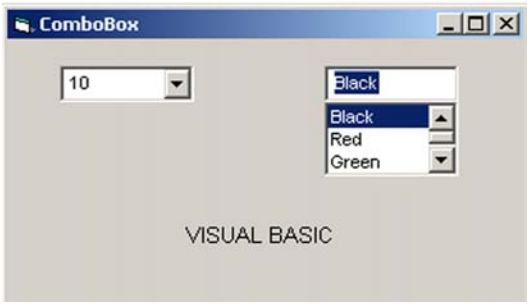
Index یک عدد صحیح است که شماره اندیس عضو مورد نظر را در کنترل معین می‌کند. شماره اندیس‌ها از صفر شروع می‌شود.

### ۱۲-۲-۲-۴ رویدادهای کنترل ComboBox

این کنترل نیز مانند سایر کنترل‌هایی که تاکنون معرفی شده‌اند رویدادهای متعددی دارد که به‌صورت مشترک توضیح داده شده‌اند در این‌جا لازم است رویداد Change این کنترل را مورد بررسی قرار دهیم.

رویداد Change فقط در رابطه با انواع DropDownCombo و SimpleCombo قابل استفاده است و زمانی اجرا می‌شود که محتویات موجود در بخش TextBox کنترل ComboBox تغییر کند.

**مثال:** می‌خواهیم پروژه‌ای طراحی کنیم که شامل دو کنترل ComboBox و یک کنترل برچسب باشد و به‌وسیله یکی از کنترل‌های ComboBox بتوان اندازه قلم و به‌وسیله کنترل دوم رنگ قلم را در کنترل برچسب تنظیم کرد. در ضمن کاربر قادر باشد در صورت نیاز اندازه‌های قلم مورد نیاز خود را که در کنترل ComboBox مربوطه موجود نیست به آن اضافه کند. فرم پروژه به‌صورت شکل ۱۱-۱۲ است.




**شکل ۱۱-۱۲**

پس از طراحی فرم رویه‌های پروژه را به‌صورت زیر تنظیم کنید:

```
Private Sub Form_Load()
    cbosize.AddItem "6"
    cbosize.AddItem "8"
    cbosize.AddItem "10"
    cbosize.AddItem "12"
    cbosize.AddItem "14"
    cbocolor.AddItem "Black"
    cbocolor.AddItem "Red"
    cbocolor.AddItem "Green"
    cbocolor.AddItem "Yellow"
    cbosize.ListIndex = 2
    cbocolor.ListIndex = 0
    lbltext.AutoSize = True
    lbltext.ForeColor = vbBlack
End Sub


Private Sub cbosize_Click()
    lbltext.FontSize = Val(cbosize.Text)
End Sub


Private Sub cbocolor_click()
    Select Case cbocolor.ListIndex
        Case 0
            lbltext.ForeColor = vbBlack
        Case 1
            lbltext.ForeColor = vbRed
        Case 2
            lbltext.ForeColor = vbGreen
        Case 3
            lbltext.ForeColor = vbYellow
```

```
    End Select
End Sub


Private Sub cbosize_LostFocus()
    cbosize.AddItem cbosize.Text
    lbltext.FontSize = cbosize.Text
End Sub
```

همان‌طور که در رویه‌های قبل مشاهده می‌کنید در رویداد Load فرم با استفاده از متد AddItem اعضای مورد نظر به هر یک از لیست‌ها اضافه می‌شوند سپس با استفاده از خاصیت ListIndex کنترل‌های ComboBox، گزینه‌های پیش‌فرض تنظیم می‌شوند. برای تغییر اندازه قلم و رنگ متن کنترل Label نیز از رویداد Click کنترل‌های ComboBox استفاده شده است. اما برای اضافه شدن اعضای جدید به کنترل cbosize در زمان اجرای برنامه از رویداد LostFocus استفاده شده است. تا وقتی کاربر مقداری را در بخش TextBox کنترل وارد کرد و فوکوس را از کنترل cbosize به کنترل بعدی انتقال داد مقدار وارد شده در لیست اضافه شود.

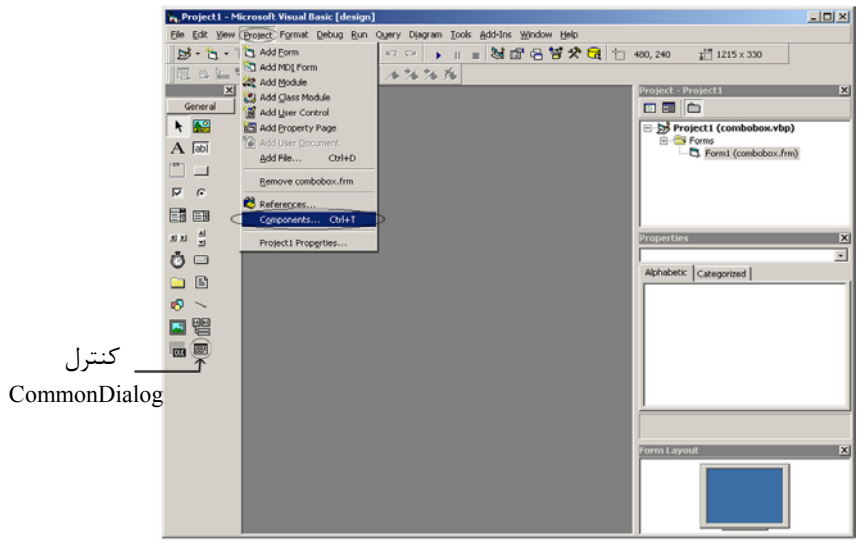
پس از آن که رویدادها را تنظیم کردید برنامه را اجرا کنید و عملکرد برنامه را با انتخاب مقادیر مختلف در کنترل‌ها بررسی کنید. پس از مشاهده صحت عملکرد برنامه، روی لیست مربوط به اندازه قلم، عدد ۱۵ را بنویسید و سپس کلید Tab را در صفحه کلید بزنید سپس مجدداً به لیست قبلی بازگشته و آن را باز کنید. آیا مقدار جدید به لیست اضافه شده است؟

## ۱۲-۳ کنترل CommonDialog

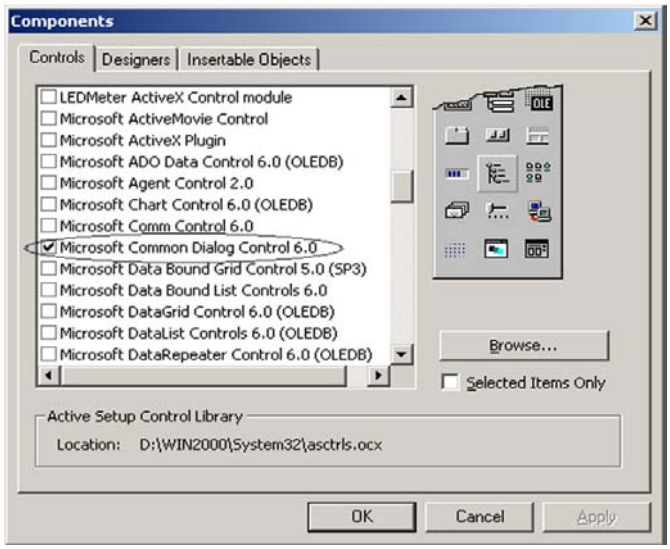
به‌وسیله این کنترل می‌توانید انواع کادرهای محاوره‌ای را مطابق استاندارد شرکت مایکروسافت ایجاد کنید. این کنترل امکان ایجاد کادرهای محاوره‌ای برای باز کردن و ذخیره‌سازی فایل‌ها، انجام عملیات چاپ، انتخاب رنگ‌ها و فونت‌ها و نمایش راهنما را فراهم می‌آورد. آیکن این کنترل به‌طور پیش‌فرض در جعبه ابزار دیده نمی‌شود و در صورت لزوم باید مراحل زیر را انجام دهید:

۱- روی منوی Project در نوار منوی ویژوال بیسیک 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. مراحل انجام عملیات فوق را در شکل ۱۲-۱۲ و ۱۳-۱۲ مشاهده کنید.




**شکل ۱۲-۱۲ نحوه اضافه کردن کنترل CommonDialogبه جعبه ابزار**




**شکل ۱۳-۱۲ پنجره Component جهت اضافه کردن کنترل CommonDialogبه جعبه ابزار**

پس از اضافه شدن آیکن کنترل می‌توانید این کنترل را مانند سایر کنترل‌ها به فرم‌های خود اضافه کنید. این کنترل در هنگام اجرای برنامه در روی فرم مشاهده نمی‌شود و فقط زمانی کادرهای

محاوره‌ای نمایش داده می‌شوند که متدهای مربوط به کادرهای محاوره‌ای فراخوانی شوند. این کنترل به‌وسیله متدهای زیر می‌تواند کادرهای محاوره‌ای مربوطه را نمایش دهد این متدها را در جدول ۳-۱۲ مشاهده می‌کنید.

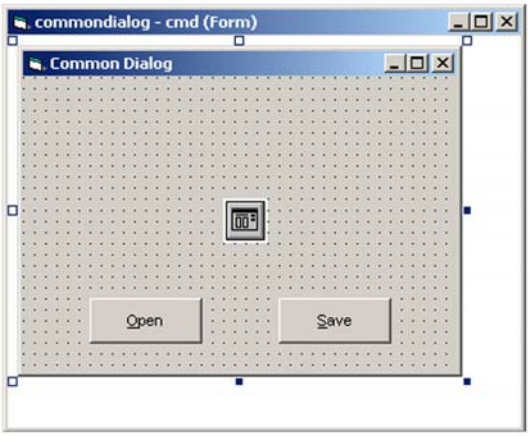
**جدول ۳-۱۲ انواع متدهای کنترل CommonDialog**

| نام کادر محاوره‌ای | نام متد |
|---|---|
| کادر محاوره‌ای باز کردن فایل‌ها | ShowOpen |
| کادر محاوره‌ای ذخیره‌سازی فایل‌ها | showSave |
| کادر محاوره‌ای رنگ‌ها | showColor |
| کادر محاوره‌ای فونت | showFont |
| کادر محاوره‌ای چاپگر | showPrinter |
| کادر محاوره‌ای راهنما | showHelp |

## ۱-۳-۱۲ نحوه ایجاد کادرهای محاوره بازکردن و ذخیره‌سازی فایل‌ها

برای ایجاد این گونه از کادرهای محاوره‌ای ابتدا یک کنترل CommonDialog روی فرم اضافه کنید و سپس به‌وسیله متدهای ShowOpen و ShowSave کادرهای محاوره‌ای مربوطه را ایجاد کنید.

**مثال:** می‌خواهیم یک فرم با یک کنترل CommonDialog و دو دکمه فرمان مطابق شکل ۱۴-۱۲ ایجاد کنیم که با فشردن هر یک از دکمه‌های فرمان یکی از کادرهای محاوره بازکردن و ذخیره‌سازی فایل را نمایش دهد.

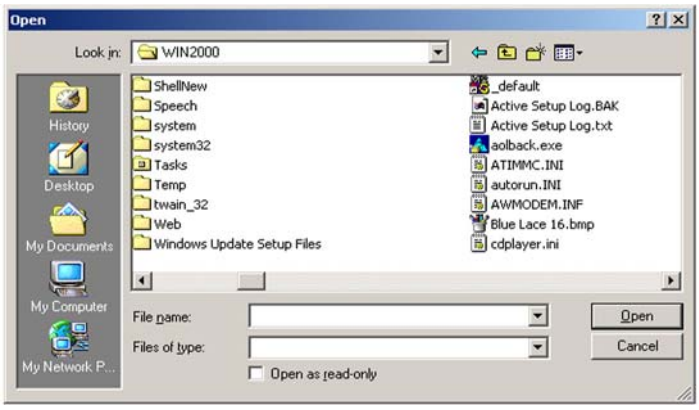



**شکل ۱۴-۱۲**

پس از طراحی فرم، رویدادهای زیر را در بخش ماژول فرم اضافه کنید:

```
Private Sub cmdopen_Click()
    dlg.ShowOpen
End Sub


Private Sub cmdsave_Click()
    dlg.ShowSave
End Sub
```

اکنون برنامه را اجرا کنید؛ ابتدا روی دکمه فرمان Open و بعد روی دکمه Save کلیک کنید. همان‌طور که مشاهده می‌کنید کادرهای محاوره‌ای مربوطه به صورت شکل ۱۵-۱۲ و ۱۶-۱۲ نمایش داده می‌شوند.




**شکل ۱۵-۱۲ پنجره باز کردن فایل**


**شکل ۱۶-۱۲ پنجره ذخیره‌سازی فایل**

کنترل Common Dialog در این دو حالت می‌تواند خواص زیر را در دسترس شما قرار دهد:

**الف‌ـ خاصیت FileName :** این‌خاصیت نام و مسیر فایلی را که توسط کاربر انتخاب شده است، نگهداری می‌کند. شکل کلی نحوه استفاده از این خاصیت به‌صورت زیر است :

CommonDialog. FileName [ = Path name ]

Path name یک عبارت رشته‌ای است که نام و مسیر فایل مورد نظر را معین می‌کند. اگر از این قسمت استفاده نشود نام فایل انتخاب شده در کادر محاوره‌ای برگشت داده می‌شود.

**ب‌ـ خاصیت Filter :** این خاصیت می‌تواند نوع فایل‌هایی را که باید در کادر محاوره نمایش داده شوند تعیین کند. مثلاً اگر بخواهیم فایل‌هایی با پسوند txt و doc را در کادر محاوره‌ای ببینیم، رویداد دکمه فرمان Open در مثال قبل به‌صورت زیر تغییر می‌کند.

```
Private Sub cmdopen_Click()
    dlg.Filter = "Text Files (*.txt)|*.txt|Documents
(*.doc)|*.doc"
    dlg.ShowOpen
End Sub
```

پس از انجام تغییرات فوق برنامه را اجرا کنید و روی دکمه فرمان Open کلیک کنید، کادر محاوره‌ای Open باز می‌شود. در کادر محاوره‌ای 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 Text Files (*.txt) انتخاب شود فقط فایل‌های متنی و در صورتی که گزینه Documents (*.doc) را انتخاب کنید فقط فایل‌های سند با پسوند doc را خواهید دید. برای دیدن تمام فایل‌ها می‌توانید از ترکیب (*.*) استفاده کنید. هر بخش در خاصیت Filter شامل دو قسمت می‌شود؛ قسمت اول یک توضیح است که در لیست Files of types دیده می‌شود و قسمت دوم عبارتی است که می‌تواند نحوه نمایش فایل‌ها را تعیین کند.

**ج‌ـ خاصیت FilterIndex :** در صورتی که بیش از یک فایل را به‌وسیله خاصیت Filter برای نمایش در کادر محاوره تعیین کرده باشید، به‌طور پیش‌فرض اولین نوع فایل در خاصیت Filter در کادر محاوره‌ای در نظر گرفته می‌شود. به‌وسیله خاصیت FilterIndex می‌توانید گزینه پیش‌فرض را انتخاب کنید. همان‌طور که مشاهده کردید در رویداد cmdopen-click قسمت (ب) دو نوع فایل توسط خاصیت Filter قابل انتخاب است که به‌طور پیش‌فرض با اجرای متد ShowOpen کادر محاوره‌ای مربوطه بازشده و فایل‌های متنی رانمایش می‌دهد. در صورتی که بخواهید در زمان نمایش کادر محاوره Open ، فایل‌های با پسوند doc مشاهده شوند، فرمان زیر را قبل از متد cmd. ShowOpen بنویسید.

Cmd. FilterIndex = 2

پس از انجام تغییرات فوق برنامه را اجرا کنید و گزینه انتخاب شده در لیست Files of type را

مشاهده کنید، آیا تغییری به‌وجود آمده است؟

همان‌طور که مشاهده کردید به‌جای Text Files (*.txt) عبارت Documents (*.doc) انتخاب شده است.

خاصیت FilterIndex می‌تواند مقادیر عددی صحیح از ۱ به بالا را کسب کند و بر اساس قرار گرفتن گزینه‌های مورد نظر در خاصیت Filter محاسبه شود. مقدار پیش‌فرض این خاصیت اولین گزینه یا مقدار ۱ است. شکل کلی نحوه استفاده از این خاصیت به‌صورت زیر است:

نام کنترل CommonDialog . FilterIndex [ = number ]

number مقدارعددی است که به شماره ترتیب نوع فایل‌های مربوطه درخاصیت Filter اشاره می‌کند.

**دـ خاصیت FileTitle :** به‌وسیله این خاصیت می‌توان به نام فایل انتخاب شده دسترسی پیدا کرد. این خاصیت فقط نام فایل را بدون مسیر آن نگهداری می‌کند. شکل کلی نحوه استفاده از این خاصیت به‌صورت زیر است:

نام کنترل CommonDialog . FileTitle

**هـ ـ خاصیت InitDir :** زمانی که کادر محاوره Open یا Save As نمایش داده می‌شود اسامی فایل‌ها و پوشه‌های مسیر جاری در دیسک نمایش داده می‌شود. در صورتی که بخواهید مسیر ویژه‌ای را برای کادر محاوره‌ای در نظر بگیرید می‌توانید مسیر مورد نظر را در این خاصیت ذخیره کنید. شکل کلی نحوه استفاده از این خاصیت به‌صورت زیر است:

نام کنترل CommonDialog .InitDir [ = string ]

string یک عبارت رشته‌ای است که مسیر مورد نظر را تعیین می‌کند.

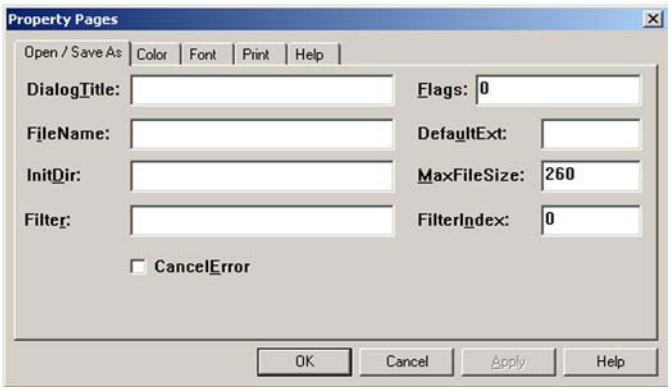
**نکته**

کادرهای محاوره‌ای Open و Save As خواص دیگری نیز دارند که کاربرد کمتری دارد و در این‌جا از ذکر آن‌ها خودداری می‌کنیم. در صورت‌نیاز می‌توانید با مراجعه به‌کتاب‌های مرجع و یا راهنمای MSDN در این رابطه، اطلاعاتی کسب کنید.

**نکته**

خواص فوق را می‌توانید از طریق کادر محاوره Property Pages نیز تنظیم کنید. برای دسترسی به پنجره مزبور، در پنجره خواص، کنترل CommonDialog را انتخاب کرده و روی خاصیت (Custom) آن کلیک کنید، در ادامه در روبه‌روی این خاصیت روی

دکمه ... کلیک کنید، کادر محاوره Property Pages نمایش داده خواهد شد. در ادامه روی زبانه Open / Save As کلیک کنید، شما می‌توانید مقادیر مورد نظر را برای هر خاصیت در این پنجره و در بخش مربوط به هر خاصیت بنویسید.




**شکل ۱۷-۱۲**

**مثال :** می‌خواهیم برنامه مثال قبل را طوری تغییر دهیم که نام و مسیر فایل انتخاب شده توسط کاربر در یک کنترل برچسب نمایش داده شود. برای انجام این کار ابتدا یک کنترل برچسب به فرم پروژه اضافه کنید و سپس رویدادها را به‌صورت زیر بنویسید:

```
Private Sub Form_Load()
    dlg.Filter = "Text File (*.txt)|*.txt|All Files(*.*)|*.*"
    dlg.FilterIndex = 1
    dlg.InitDir = "c:\"
    lblFileName.AutoSize = True
    lblFileName.Alignment = vbCenter
    lblFileName.Visible = False
End Sub


Private Sub cmdopen_Click()
    dlg.DialogTitle = "MyOpen"
    dlg.ShowOpen
    lblFileName.Visible = True
    lblFileName.Caption = dlg.FileName
End Sub


Private Sub cmdsave_Click()
    dlg.DialogTitle = "MySave"
    dlg.ShowSave
```

```
    lblFileName.Visible = True
    lblFileName.Caption = dlg.FileName
End Sub
```

همان‌طور که در رویه‌های بالا مشاهده می‌کنید ابتدا در رویداد Load فرم، خواص Filter، FilterIndex و InitDir مربوط به کنترل CommonDialog تنظیم می‌شود و سپس خواص کنترل برچسب تنظیم می‌شود تا فرم آماده نمایش شود.

در رویداد Click هر دو دکمه فرمان Open و Save ابتدا عنوان کادر محاوره توسط خاصیت DialogTitle تنظیم می‌گردد، سپس با استفاده از متد ShowOpen و ShowSave کادر محاوره متناظر با دکمه فرمان نمایش داده می‌شود و وقتی کاربر فایل خود را انتخاب کند و روی دکمه Open یا Save کلیک کند. خاصیت Visible کنترل برچسب به True تنظیم می‌شود، سپس نام فایل و مسیر آن در خاصیت Caption کنترل برچسب ذخیره می‌شود تا در روی فرم نمایش داده شود.

اکنون برنامه را اجرا کنید و روی دکمه فرمان Open کلیک کنید، کادر محاوره Open باز می‌شود و اسامی فایل‌های متنی موجود در C:\ را نمایش می‌دهد. به عنوان کادر محاوره توجه کنید کلمه MyOpen را در نوار عنوان مشاهده می‌کنید. در بخش File name کلمه Ali را به‌عنوان نام فایل بنویسید و بعد روی دکمه فرمان Open در کادر محاوره کلیک کنید. پنجره برنامه به‌صورت شکل ۱۸-۱۲ نمایش داده می‌شود. اجرای برنامه را متوقف کرده و آن را مجدداً اجرا کنید، سپس روی دکمه فرمان Open دوباره کلیک کرده و عملیات فوق را مجدداً تکرار کنید، اما این بار به‌جای دکمه فرمان Open در کادر محاوره‌ای روی دکمه فرمان Cancel کلیک کنید، آیا نام فایل مورد نظر را در فرم می‌بینید؟

عملکرد برنامه را برای دکمه فرمان Save آزمایش کرده، نتایج را بررسی کنید.

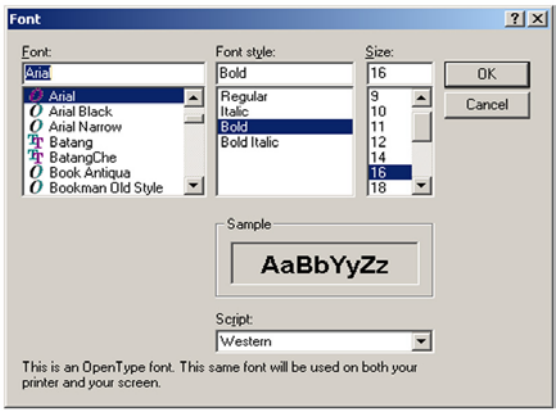

**شکل ۱۸-۱۲**

**نکته**

کادرهای محاوره Open و Save As فایل‌ها را باز یا ذخیره نمی‌کنند. این کادرها و گزینه‌های موجود در آن‌ها مقدمات کار را فراهم می‌آورد. بنابراین وظیفه برنامه‌نویس است تا فایل‌های خود را با دستورات مناسب و با استفاده از این کادرهای محاوره باز کرده و یا ذخیره کند. در این رابطه در فصول بعدی مطالب لازم را فرا خواهید گرفت.

## ۱۲-۳-۲ نحوه ایجاد کادر محاوره قلم (Font)

برای ایجاد این‌گونه از کادرهای محاوره ابتدا یک کنترل CommonDialog روی فرم اضافه کنید و سپس به‌وسیله متد ShowFont کادر محاوره‌ای مربوط به قلم را ایجاد کنید. در شکل ۱۹-۱۲ نمونه‌ای از کادر محاوره Font را ملاحظه می‌کنید.




**شکل ۱۹-۱۲**

کنترل CommonDialog در این حالت خواص زیر را در اختیار شما قرار می‌دهد:

### الف- خاصیت Color

این خاصیت رنگ قلم را تنظیم می‌کند. می‌توانید این خاصیت را از پنجره خواص و یا از طریق کد نویسی تنظیم کنید. شکل کلی نحوه استفاده از این خاصیت به‌صورت زیر است:

نام کنترل CommonDialog . Color [ = number ]

number یک عبارت عددی است که رنگ قلم را معین می‌کند. می‌توانید از مقادیر ثابت عددی، رشته‌ای یا توابع مربوط به رنگ‌ها که در فصل قبل ارایه شده‌اند، استفاده کنید. در صورت عدم استفاده از این بخش رنگ فعلی قلم بازگشت داده می‌شود.

**ب‌ـ خاصیت Flags**

به‌وسیله این خاصیت می‌توانید کادر محاوره قلم (Font) را با توجه به نیاز خود نمایش دهید. شکل کلی نحوه استفاده از این خاصیت به‌صورت زیر است:

نام کنترل CommonDialog . Flags [ = value ]

value مقدار ثابتی است که می‌تواند یکی از مقادیر جدول ۴-۱۲ کسب کند.

**جدول ۴-۱۲ مقادیر مربوط به خاصیت Flags در کادر محاوره Font**

| ثابت رشته‌ای | توضیح |
|---|---|
| cdlCFApply | دکمه فرمانApply در کادر محاوره‌ای نمایش داده می‌شود. |
| cdlCFBoth | نام قلم‌های مربوط به چاپگر وصفحه نمایش رادرکادرمحاوره‌ای نمایش می‌دهد. |
| cdlCFEffects | امکان انتخاب رنگ، خط زیر و سایر جلوه‌ها را امکان‌پذیر می‌کند. |
| cdlCFPrinterFonts | فقط نام قلم‌های مربوط به چاپگر را در کادر محاوره‌ای نمایش می‌دهد. |
| cdlCFScreenFonts | فقط نام قلم‌های مربوط به صفحه نمایش را در کادر محاوره‌ای نمایش می‌دهد. |

**نکته**

قبل از فراخوانی متد ShowFont مقدار خاصیت Flags را روی یکی از سه مقدار cdlCFPrinterFonts ، cdlCFBoth و یا cdlCFScreenFonts تنظیم کنید در غیر این صورت پیام خطایی ظاهر خواهد شد.

**ج‌ـ خواص مربوط به جلوه‌های ویژه در قلم‌ها**

به‌وسیله چهار خاصیت ارایه شده در جدول ۵-۱۲ می‌توانید به قلم‌های خود جلوه‌های ویژه‌ای را اضافه کنید.

**جدول ۵-۱۲ خواص مربوط به ایجاد جلوه‌ها در قلم**

| نام خاصیت | توضیح |
|---|---|
| FontBold | قلم پررنگ |
| FontItalic | قلم به‌صورت مایل |
| FontStrikethru | قلم با خط وسط در هر کاراکتر |
| Fontunderline | قلم با خط زیر در هر کاراکتر |

شکل کلی نحوه استفاده از این چهار خاصیت به‌صورت زیر است:

نام خاصیت . نام کنترل CommonDialog [ =boolean ]

مقدار boolean یک عبارت منطقی True یا False است که فعال یا غیر فعال بودن جلوه مربوطه را معین می‌کند. اگر مقدار هر یک از خواص فوق True باشد جلوه مربوطه فعال و در غیر این صورت غیر فعال خواهد بود.

در صورت عدم استفاده از مقدار boolean، مقدار فعلی خاصیت بازگشت داده می‌شود.

**دـ خاصیت FontName**

وقتی در کادر محاوره Font نام یک قلم انتخاب شود، نام قلم انتخاب شده در این خاصیت نگهداری می‌شود. شکل کلی نحوه استفاده از این خاصیت به‌صورت زیر است:

نام کنترل CommonDialog . FontName [ = font ]

در صورتی که بخواهید قلم مورد نظر در کادر محاوره به‌عنوان قلم پیش‌فرض انتخاب شود نام قلم را در این خاصیت ذخیره کنید. بخش font یک عبارت رشته‌ای است که استفاده از آن اختیاری می‌باشد و در صورت عدم استفاده از این بخش، نام قلم انتخاب شده در کادر محاوره در اختیار شما قرار می‌گیرد.

**هـ ـ خاصیت FontSize**

به‌وسیله این خاصیت می‌توان اندازه قلم را در کادر محاوره‌ای به‌طور پیش‌فرض انتخاب کرد یا اندازه قلم انتخاب شده توسط کاربر را پس از بسته شدن کادر محاوره به‌دست آورد. شکل کلی نحوه استفاده از این خاصیت به‌صورت زیر است:

نام کنترل CommonDialog . FontSize [ = points ]

points یک عبارت عددی است که اندازه پیش‌فرض را برای قلم در کادر محاوره تعیین می‌کند. در صورت عدم استفاده از این بخش اندازه قلم انتخاب شده کاربر به‌دست می‌آید.

**مثال:** یک پروژه از نوع Standard EXE را با یک فرم به همراه یک کنترل دکمه فرمان، برچسب و کنترل CommonDialog مطابق شکل ۲۰-۱۲ ایجاد کنید.


**شکل ۲۰-۱۲**

سپس رویداد click دکمه فرمان را مانند رویداد زیر کد نویسی کنید :

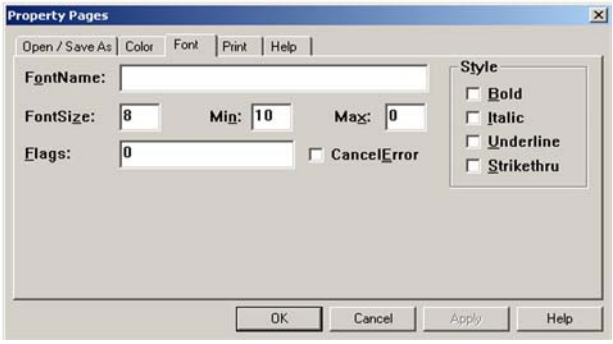
```
Private Sub cmdfont_Click()
    dlg.DialogTitle = "MyFont"
    dlg.Flags = cdlCFBoth + cdlCFEffects
    dlg.Color = vbBlack
    dlg.FontSize = 20
    dlg.FontName = "Arial"
    lbltext.FontName = dlg.FontName
    lbltext.FontSize = dlg.FontSize
    lbltext.ForeColor = dlg.Color
    dlg.ShowFont
    lbltext.FontName = dlg.FontName
    lbltext.FontSize = dlg.FontSize
    lbltext.FontBold = dlg.FontBold
    lbltext.ForeColor = dlg.Color
    lbltext.FontItalic = dlg.FontItalic
    lbltext.FontStrikethru = dlg.FontStrikethru
    lbltext.FontUnderline = dlg.FontUnderline
End Sub
```

همان‌طور که در این رویداد مشاهده می‌کنید ابتدا خواص Flags ، Color ، FontSize و FontName برای کنترل dlg و سپس کنترل برچسب روی مقادیر مورد نظر تنظیم می‌شود، سپس به‌وسیله متد ShowFont، کادر محاوره Font ظاهر می‌شود که در صورت انتخاب مقادیر جدید و کلیک روی دکمه OK در کادر محاوره، دستورات بعد از این متد اجرا شده و خواص متناظر کنترل برچسب را بر اساس انتخاب‌های کاربر در کادر محاوره Font تنظیم می‌کند. در شکل ۲۱-۱۲ نتیجه اجرای برنامه را پس از بسته شدن کادر محاوره Font مشاهده می‌کنید. این شکل را با شکل ۲۰-۱۲ مقایسه کنید.


**شکل ۲۱-۱۲**

**نکته**

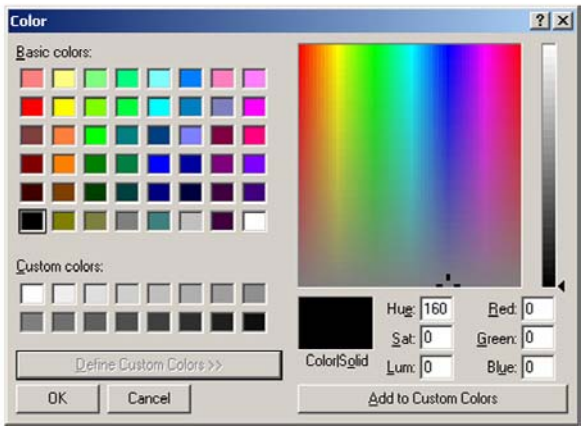
خواص فوق را می‌توانید از طریق زبانه Font در کادر محاوره Property Pages نیز تنظیم کنید. نحوه دسترسی به این کادر محاوره در بخش‌های قبل توضیح داده شده است.




**شکل ۱۲-۲۲**

## ۱۲-۳-۳ نحوه ایجاد کادر محاوره رنگ (Color)

برای ایجاد این‌گونه کادرهای محاوره ابتدا یک کنترل CommonDialog روی فرم اضافه کنید و سپس به‌وسیله متد ShowColor کادر محاوره مربوط به رنگ را نمایش دهید. در شکل ۱۲-۲۳ نمونه‌ای از کادر محاوره Color را مشاهده می‌کنید. کنترل CommonDialog در این حالت خواص زیر را در اختیار شما قرار می‌دهد.




**شکل ۱۲-۲۳**

**الف‌ـ خاصیت Color**

این خاصیت شماره رنگ انتخاب شده به‌وسیله کاربر را نگهداری می‌کند. شکل کلی نحوه استفاده از این خاصیت به‌صورت زیر است:

CommonDialog نام کنترل . Color [ = number ]

number یک عبارت عددی است که رنگ پیش‌فرض را در کادر محاوره تعیین می‌کند و می‌تواند یک مقدار ثابت عددی یا یک ثابت رشته‌ای باشد. در صورت عدم استفاده از آن، رنگ انتخابی کاربر در کادر محاوره را در اختیار شما قرار می‌دهد.

**ب‌ـ خاصیت Flags**

به‌وسیله این خاصیت می‌توانید کادر محاوره‌ای رنگ را با توجه به نیاز خود تنظیم کرده، نمایش دهید. شکل کلی نحوه استفاده از این خاصیت به‌صورت زیر است:

CommonDialog نام کنترل . Flags [ = value ]

value مقدار ثابتی است که می‌تواند یکی از مقادیر موجود در جدول ۶-۱۲ باشد.

**جدول ۶-۱۲ مقادیر مربوط به خاصیت Flags در کادر محاوره رنگ**

| ثابت رشته‌ای | توضیح |
|---|---|
| cdlCCFullOpen | کادر محاوره رنگ به‌طور هم‌زمان همراه با بخش تعریف رنگ نمایش داده می‌شود. در صورت عدم استفاده از این ثابت کاربر باید روی دکمه Define custom colors >> کلیک کند تا بخش تعریف رنگ فعال شود. |
| cdlCCPreventFullOpen | امکان استفاده از دکمه Define custom colors >> جهت نمایش بخش تعریف رنگ وجود نخواهد داشت. |
| cdlCCHelpButton | دکمه Help در کادر محاوره نمایش داده می‌شود. |

**مثال:** مانندمثال قبل یک‌فرم به همراه یک کنترل دکمه فرمان، برچسب و یک‌کنترل CommonDialog طراحی کنید. هدف از این مثال این است که رنگ کاراکترها در کنترل برچسب به رنگ انتخابی کاربر در کادر محاوره رنگ تبدیل شود. پس از طراحی شکل ظاهری برنامه رویداد Click دکمه فرمان color را به‌صورت زیر تنظیم کنید:

```
Private Sub cmdcolor_Click()
    dlg.Flags = cdlCCHelpButton + cdlCCFullOpen
```

```
    dlg.ShowColor
    lbltext.ForeColor = dlg.Color
End Sub
```

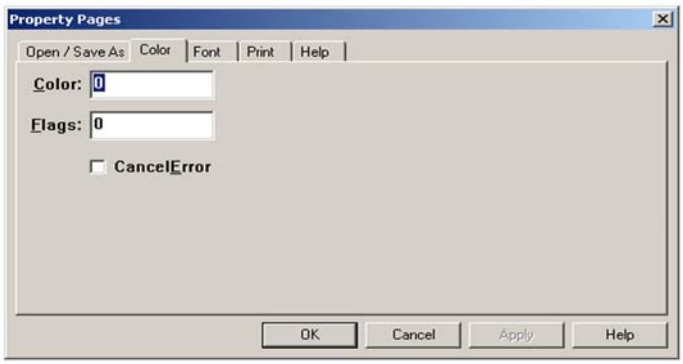
همان‌طور که در رویه رویداد فوق مشاهده می‌کنید ابتدا خاصیت Flags کنترل dlg با توجه به جدول ۶-۱۲ تنظیم شده است. برای استفاده از چند مقدار به‌طور هم‌زمان باید از علامت + در بین ثابت‌های رشته‌ای استفاده کرد.

پس از اجرای متد ShowColor و نمایش کادرمحاوره رنگ، کاربر می‌تواند یک‌رنگ را انتخاب یا تعریف‌کند، سپس روی دکمه OK کلیک‌کند. در این صورت شماره رنگ مربوطه در خاصیت Color کنترل dlg ذخیره شده و پس از متد ShowColor به خاصیت ForeColor کنترل برچسب نسبت داده می‌شود.

برنامه را اجرا کرده و روی دکمه Color کلیک کنید و پس از انتخاب یک رنگ در کادر محاوره روی دکمه OK کلیک کنید. رنگ عنوان کنترل برچسب به رنگ انتخاب شده نمایش داده می‌شود.

**نکته**

خواص فوق را می‌توانید از طریق زبانه Color در کادر محاوره Property Pages نیز تنظیم کنید.

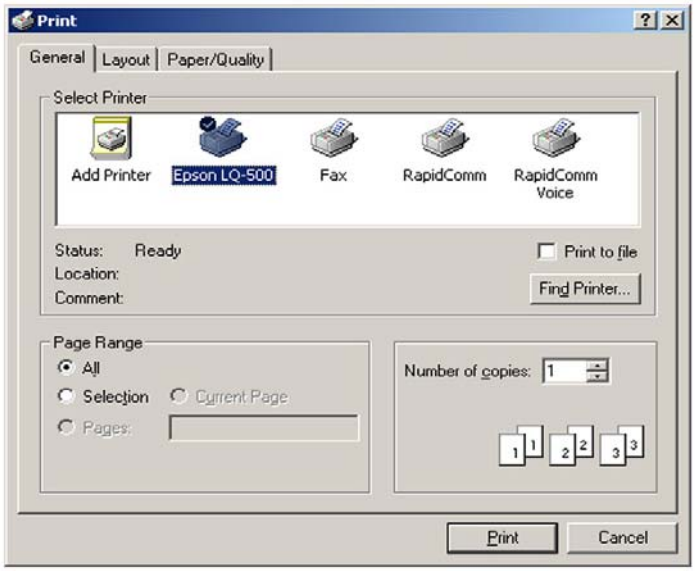



**شکل ۲۴-۱۲**

## ۴-۳-۱۲ نحوه ایجاد کادر محاوره چاپ (Print)

برای ایجاد این نوع کادرهای محاوره ابتدا یک کنترل CommonDialog روی فرم اضافه کنید و سپس به‌وسیله متد ShowPrinter کادر محاوره مربوط به چاپ را فعال کنید. در شکل ۲۵-۱۲ نمونه‌ای از کادر محاوره چاپ را مشاهده می‌کنید. کنترل CommonDialog در این حالت خواص زیر را در

اختیار شما قرار می‌دهد.




**شکل ۱۲-۲۵**

**الف_ خاصیت Copies**

به‌وسیله این خاصیت می‌توان تعداد نسخه‌های مورد نظر برای چاپ را مشخص کرد. شکل کلی نحوه استفاده از این خاصیت به‌صورت زیر است:

نام کنترل CommonDialog . Copies [ = number ]

number یک عبارت عددی از نوع صحیح است که تعداد نسخه‌ها را جهت چاپ تعیین می‌کند. در صورت عدم استفاده از این مقدار، تعداد نسخه‌های چاپ بازگشت داده می‌شود.

**ب_ خاصیت Flags**

به‌وسیله این خاصیت می‌توانید کادر محاوره چاپ را با توجه به نیاز خود تنظیم کنید. شکل کلی نحوه استفاده از این خاصیت به‌صورت زیر است:

نام کنترل CommonDialog . Flags [ = value ]

value مقدار ثابتی است که می‌تواند یکی از مقادیر موجود در جدول ۷-۱۲ باشد. در صورت عدم استفاده از این بخش مقدار خاصیت مزبور بازگردانده می‌شود.

**جدول ۷-۱۲ مقادیر مربوط به خاصیت Flags در کادر محاوره چاپ**

| ثابت رشته‌ای | توضیح |
|---|---|
| cdlPDDisablePrintToFile | کادر علامتPrint to file را در کادر محاوره چاپ در زبانه General غیرفعال می‌کند. |
| cdlPDHidePrintToFile | کادر علامت Print to file را در کادر محاوره چاپ در زبانهGeneral مخفی می‌کند. |
| cdlPDNoSelection | دکمه رادیویی Selection را در بخش Page Range زبانه General غیرفعال می‌کند. |
| cdlPDPrintSetup | به‌جای کادر محاوره چاپ، کادر محاوره Print Setup نمایش داده می‌شود. |
| cdlPDPageNums | کادر علامت Collate را در زبانه General فعال می‌کند. |

**ج- خواص FromPage و ToPage**

به‌وسیله این دو خاصیت می‌توانید شماره صفحه شروع FromPage و صفحه خاتمه ToPage را جهت چاپ تعیین کنید. شکل کلی نحوه استفاده از این دو خاصیت به‌صورت زیر است:

نام کنترل CommonDialog . FromPage [ = number ]

نام کنترل CommonDialog . ToPage [ = number ]

number یک عبارت عددی از نوع صحیح است که شماره اولین صفحه و آخرین صفحه را جهت چاپ معین می‌کند. در صورت عدم استفاده از number مقدار خواص فوق بازگردانده می‌شود.

**نکته**

این‌خواص وقتی درست عمل می‌کنند که‌خاصیت Flags روی مقدار cdlPDPageNums تنظیم شده باشد.

**نکته**

کادر محاوره چاپ و دکمه Print موجود در آن هیچ‌گونه عملیات چاپی را انجام نمی‌دهد و این کادر محاوره فقط مراحل اولیه را در تنظیمات چاپ انجام می‌دهد. برای انجام عملیات چاپ برنامه‌نویس باید کدهای مناسب را بنویسد.

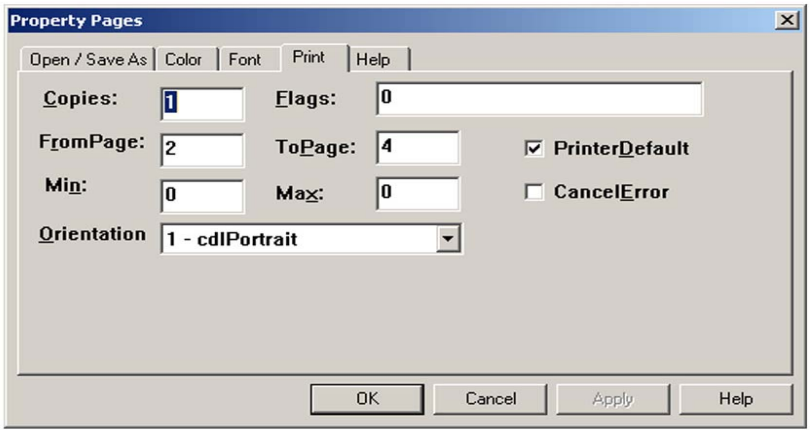
**مثال:** یک برنامه از نوع Standard EXE ایجاد کرده و یک کنترل دکمه فرمان با نام cmdprint و عنوان Print و یک کنترل CommonDialog در روی آن قرار دهید، سپس رویداد Click دکمه فرمان print را به این صورت تنظیم کنید:

```
Private Sub cmdprint_Click()
    dlg.Flags = cdlPDDisablePrintToFile + cdlPDCollate
    dlg.ShowPrinter
End Sub
```

همان‌طور که در رویه رویداد click فوق مشاهده می‌شود قبل از استفاده از متد ShowPrinter ابتدا خاصیت Flags برای کنترل CommonDialog به‌گونه‌ای تنظیم شده است تا کادر علامت Print to File در زبانه General دیده نشود ولی کادر علامت Collate در کادر محاوره چاپ، در زبانه General مشاهده شود.

**نکته**

خواص فوق را می‌توانید از طریق زبانه Print در کادر محاوره Property Pages نیز تنظیم کنید.




**شکل ۲۶-۱۲**

**نکته**

خواص Max و Min می‌توانند بزرگ‌ترین و کوچک‌ترین مقادیری را که کاربر می‌تواند برای خواص Frompage و Topage در کادر محاوره چاپ تایپ کند معین کند.

**نکته**

در کادر لیست Orientation می‌توانید نحوه انجام چاپ را به یکی از دو صورت : Portrait (1-cdlPortrait) و Landscape (2-cdlLandscape) انتخاب کنید.

### ۱۲-۳-۵ نحوه ایجاد کادر محاوره راهنما (Help)

برای ایجاد یک کادر محاوره راهنما علاوه بر استفاده از یک کنترل CommonDialog و متد ShowHelp لازم است تا مطالب بیشتری در رابطه با نحوه ایجاد و طراحی فایل‌های راهنما و چگونگی استفاده از آن‌ها را بدانید. ارایه مطالب فوق از بحث این کتاب خارج است شما می‌توانید جهت دریافت اطلاعات مورد نیاز خود در این زمینه به منابع دیگری مراجعه کنید.

## ۱۲-٤ کنترل DriveListBox

به‌وسیله این کنترل می‌توانید لیستی از درایوهای موجود در یک سیستم را جهت انتخاب کاربر ایجاد کنید. مزیت این کنترل این است که تمام درایوهای HARD DISK، FLOPPY DISK، CD-ROM و حتی درایوهای شبکه را به‌طور خودکار شناسایی کرده و در اختیار کاربر قرار می‌دهد. در شکل ۱۲-۲۷ نمونه‌ای از این کنترل را مشاهده می‌کنید.


**شکل ۱۲-۲۷**

### ۱۲-۴-۱ خواص کنترل DriveListBox

این کنترل علاوه بر خواص مشترکی که با سایر کنترل‌ها دارد، دارای خواص ویژه‌ای است که به توضیح هر یک از آن‌ها می‌پردازیم :

### ۱۲-۴-۱-۱ خاصیت Drive

این خاصیت نام درایوی را که توسط کاربر از لیست انتخاب شده، باز می‌گرداند. در ضمن به‌وسیله این خاصیت می‌توانید درایو پیش‌فرض در لیست را تغییر دهید. به‌طور پیش‌فرض درایو جاری در لیست نمایش داده می‌شود. شکل کلی نحوه استفاده از این خاصیت به‌صورت زیر است:

نام کنترل DriveListBox . Drive [ = drive ]

drive یک عبارت رشته‌ای است که به نام درایو مورد نظر اشاره می‌کند. در صورت عدم استفاده از آن نام درایو انتخاب شده، در اختیار شما قرار می‌گیرد.

### ۱۲-۴-۱-۲ خاصیت ListCount

به‌وسیله این خاصیت می‌توان تعداد درایوهای موجود در لیست را به دست آورد. شکل کلی نحوه استفاده از این خاصیت به‌صورت زیر است:

نام کنترل DriveListBox . ListCount

به‌عنوان مثال مقدار خاصیت ListCount برای کنترل موجود در شکل ۱۲-۲۷، مقدار ۸ است.

### ۱۲-۴-۱-۳ خاصیت ListIndex

در کنترل DriveListBox هر درایو دارای یک شماره منحصر به فرد است. به‌وسیله خاصیت ListIndex می‌توان شماره درایو انتخاب شده در لیست را به‌دست آورد. شماره مربوطه از صفر برای اولین درایو آغاز شده و به ترتیب ۱ و ۲ و به همین صورت تا آخرین درایو ادامه می‌یابد. مثلاً در کنترل موجود در شکل ۱۲-۲۷ اگر کاربر روی درایو a: کلیک کند مقدار ListIndex برابر با صفر و اگر روی درایو f: کلیک کنید مقدار ListIndex برابر با ۴ خواهد بود. شکل کلی نحوه استفاده از این خاصیت به‌صورت زیر است:

نام کنترل DriveListBox . ListIndex [ = index ]

index یک عبارت عددی صحیح است که از صفر شروع می‌شود و به‌وسیله آن می‌توان درایو پیش‌فرض را در لیست انتخاب کرد. مثلاً دستور Drive1.ListIndex = 2 سبب می‌شود در زمان نمایش فرم، درایو d: به‌عنوان درایو پیش‌فرض در کنترل، نمایش داده شود. در صورت عدم استفاده از index شماره درایوی که انتخاب شده است، بازگردانده خواهد شد.

## ۱۲-۴-۲ متدهای کنترل DriveListBox

مهم‌ترین متد این کنترل SetFocus است که پس از اجرا، فوکوس را به کنترل مزبور انتقال

می‌دهد. نحوه استفاده از این کنترل به‌صورت زیر است:

نام کنترل DirListBox . SetFocus

## ۱۲-۴-۳ رویدادهای کنترل DriveListBox

مهم‌ترین رویداد این کنترل، رویداد Change است. این رویداد زمانی اجرا می‌شود که کاربر درایوی را از لیست موجود در کنترل انتخاب کند. سایر رویدادهای این کنترل مانند LostFocus و GotFocus قبلاً در رویدادهای مشترک کنترل‌ها توضیح داده شده‌اند.

**مثال :** می‌خواهیم یک برنامه از نوع Standard EXE را به همراه یک فرم، یک کنترل DriveListBox و چهار کنترل برچسب به گونه‌ای طراحی کنیم که وقتی کاربر یک درایو را از لیست کنترل انتخاب می‌کند نام درایو به همراه شماره ترتیب آن در کنترل‌های برچسب نمایش داده شوند، به‌علاوه در ابتدای نمایش فرم درایو :d در کنترل نمایش داده شود. شکل ۲۸-۱۲ فرم پروژه را در هنگام شروع برنامه نمایش می‌دهد.


**شکل ۲۸-۱۲**

اکنون طراحی فرم برنامه رویداد Load فرم و رویداد Change کنترل DriveListBoxt را به شکل زیر تنظیم کنید:

```
Private Sub Form_Load()
    Drive1.Drive = "D:"
    lblname1.Caption = Drive1.Drive
    lblindex1.Caption = Drive1.ListIndex
End Sub

Private Sub Drive1_Change()
    lblname1.Caption = Drive1.Drive
```

```
    lblindex1.Caption = Drive1.ListIndex
End Sub
```

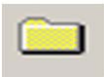
همان‌طور که مشاهده می‌کنید در رویداد Form_Load از سه دستور استفاده شده است دستور اول، درایو :D را به‌عنوان درایو پیش‌فرض در کنترل DriveListBox تعیین می‌کند، سپس نام درایو و شماره درایو :D در لیست، در دو کنترل برچسب نمایش داده می‌شوند. دو دستور آخر در این رویداد دقیقاً در رویداد Change کنترل DriveListBox نیز وجود دارند تا در صورت انتخاب یک مقدار جدید کنترل‌های برچسب مقادیر انتخابی جدید را نمایش دهند.

توجه داشته باشید، در صورتی که درایو جدیدی را در کنترل DriveListBox انتخاب کنید، درایوجاری ازنظر سیستم عامل تغییر نمی‌کند. در صورت نیاز به تغییر درایو جاری باید از دستوراتی که در این زمینه وجود دارد، استفاده کنید. برای تغییردرایو جاری به درایوی که درکنترل DriveListBox وجود دارد، از دستور ChDrive استفاده کنید. شکل کلی نحوه استفاده از این دستور به این صورت است:

ChDrive (drive )

drive یک عبارت رشته‌ای است که درایو مورد نظر را تعیین می‌کند. به‌عنوان مثال دستور ChDrive " c " درایو جاری را از نظر سیستم عامل به درایو :c تغییر می‌دهد. برای تغییر درایو جاری به‌وسیله یک کنترل DriveListBox از دستور زیر استفاده کنید:

ChDrive (DriveListBox نام کنترل . Drive )



## ۱۲-۵ کنترل DirListBox

به‌وسیله این کنترل می‌توانید لیستی از پوشه‌های موجود در یک مسیر در روی HARD DISK، FLOPPY DISK، CD-ROM و نظایر آن‌ها را مشاهده کنید و در صورت نیاز وارد پوشه مورد نظر شوید. کنترل به‌طور خودکار زیر پوشه‌های، پوشه انتخاب شده را نمایش می‌دهد. برای باز کردن هر پوشه و ورود به آن باید روی آیکن پوشه در کنترل دابل کلیک کنید. در شکل ۲۹-۱۲ نمونه‌ای از این کنترل را مشاهده می‌کنید.


**شکل ۲۹-۱۲**

## ۱-۵-۱۲ خواص کنترل DirListBox

این کنترل نیز علاوه بر خواص مشترکی که با سایر کنترل‌ها دارد، دارای خواص ویژه‌ای است که به توضیح هر یک از آن‌ها می‌پردازیم:

### ۱-۱-۵-۱۲ خاصیت ListCount

به‌وسیله این خاصیت می‌توان تعداد زیر پوشه‌های، پوشه باز شده را به‌دست آورد. شکل کلی نحوه استفاده از این خاصیت به این صورت است:

نام کنترل DirListBox . ListCount

به‌عنوان مثال مقدار خاصیت ListCount برای کنترل موجود در شکل ۲۹-۱۲، مقدار ۳ است. در واقع تعداد زیر پوشه‌های پوشه Help به‌عنوان پوشه باز شده در نظر گرفته می‌شود.

### ۲-۱-۵-۱۲ خاصیت ListIndex

در کنترل DirListBox هر پوشه دارای یک شماره منحصر به فرد می‌باشد که مقدار آن با توجه به ترتیب قرار گرفتن پوشه‌ها نسبت به پوشه‌ای که باز شده است، تنظیم می‌شود. نحوه استفاده از این خاصیت به‌صورت زیر است:

نام کنترل DirListBox . ListIndex

به‌عنوان مثال مقدار خاصیت ListIndex برای پوشه Help در شکل ۲۹-۱۲، ۱- است و مقدار این خاصیت برای پوشه WIN2000، ۲- و برای فهرست ریشه یعنی d:\، ۳- است. به‌علاوه مقدار خاصیت ListIndex برای پوشه debug صفر، برای پوشه iisHelp، ۱ و برای پوشه mail، ۲ است.

توجه داشته باشید که مقدار این خاصیت برای پوشه‌هایی تنظیم می‌شود که در بالا و پایین

پوشه‌ای که در حال حاضر باز شده است (روی آن دابل کلیک شده است)، قرار دارند، و همواره مقدار این خاصیت برای پوشه‌ای که باز شده است، ۱- است. در ضمن خاصیت ListIndex در هر لحظه با توجه به پوشه‌ای که در لیست روی آن کلیک شده است، شماره ترتیب پوشه مربوطه را باز می‌گرداند.

#### ۱۲-۵-۱-۳ خاصیت Path

این خاصیت مسیر پوشه‌ای را که در حال حاضر در کنترل DirListBox باز شده است، در اختیار شما قرار می‌دهد. البته به‌وسیله این خاصیت می‌توانید مسیر پوشه‌ای را که در هنگام نمایش کنترل می‌خواهید به‌عنوان پوشه جاری در نظر گرفته شود، تعیین کنید. شکل کلی نحوه استفاده از این خاصیت به‌صورت زیر است :

نام کنترل DirListBox . Path [ = pathname ]

pathname یک عبارت رشته‌ای است که مسیر مورد نظر را برای نمایش در لیست کنترل DirListBox تعیین می‌کند و در صورت عدم استفاده از آن، مسیر پوشه باز شده در کنترل را باز می‌گرداند.

### ۱۲-۵-۲ متدهای کنترل DirListBox

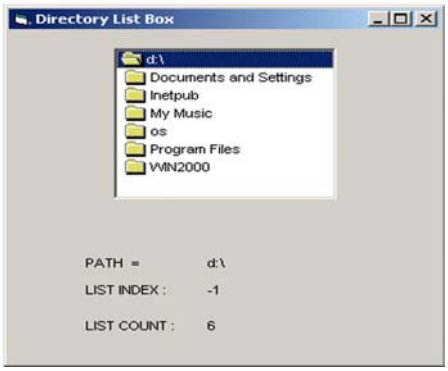
مهم‌ترین متد این کنترل SetFocus است که پس از اجرا، فوکوس را به کنترل مزبور انتقال می‌دهد. نحوه استفاده از این متد به این صورت است :

نام کنترل DirListBox .SetFocus

### ۱۲-۵-۳ رویدادهای کنترل DirListBox

مهم‌ترین رویداد این کنترل، رویداد Change است. این رویداد زمانی اجرا می‌شود که کاربر در لیست روی نام پوشه‌ای دابل کلیک کند. سایر رویدادهای این کنترل مانند LostFocus و GotFocus قبلاً در رابطه با کنترل‌های دیگر توضیح داده شده‌اند.

**مثال:** می‌خواهیم یک برنامه از نوع Standard EXE را به همراه یک فرم، یک کنترل DirListBox و ۶ کنترل برچسب به گونه‌ای طراحی کنیم که وقتی کاربر یک پوشه را باز می‌کند (روی یک پوشه دابل کلیک کند) مسیر پوشه مربوطه به‌طور کامل و به همراه تعداد زیرپوشه‌های آن نمایش داده شود و اگر در کنترل روی هر یک از پوشه‌ها کلیک کند، مقدار خاصیت ListCount به‌وسیله یک کنترل برچسب نمایش داده شود. به‌علاوه مسیری که به‌طور پیش‌فرض در کنترل نمایش داده می‌شود ریشه درایو D: باشد. شکل ۱۲-۳۰ پنجره برنامه را در زمان اجرا نمایش می‌دهد.




**شکل ۳۰-۱۲**

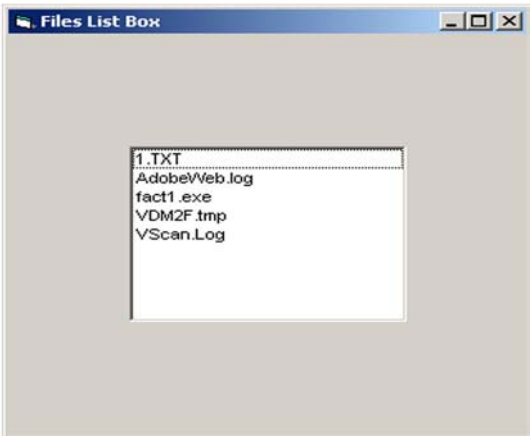
پس از طراحی فرم برنامه رویداد Load، فرم و رویداد Click و Change کنترل DirListBox را به این صورت تنظیم کنید:

```
Private Sub Form_Load()
    Dir1.Path = "d:\"
    lblpath1.Caption = Dir1.Path
    lblindex1.Caption = Dir1.ListIndex
    lblcount1.Caption = Dir1.ListCount
End Sub

Private Sub Dir1_Click()
    lblindex1.Caption = Dir1.ListIndex
End Sub


Private Sub Dir1_Change()
    lblpath1.Caption = Dir1.Path
    lblcount1.Caption = Dir1.ListCount
End Sub
```

همان‌طور که مشاهده می‌کنید در رویداد Form_Load ابتدا دستور " d:\ " = Dir1. Path مسیر پیش‌فرض را در کنترل Dir1 تنظیم کرده و سپس به ترتیب مقادیر مربوط به خواص Path، ListIndex و ListCount مربوط به این‌کنترل درخاصیت Caption مربوط به سه‌کنترل برچسب نشان‌داده می‌شوند. در رویداد کلیک کنترل Dir1 نیز تنها دستور موجود سبب می‌شود تا مقدار خاصیت ListIndex کنترل Dir1 در صورت کلیک کاربر بر روی یکی از پوشه‌ها نمایش داده شود.

در رویداد Change کنترل Dir1 نیز که با باز شدن یک پوشه اجرا می‌شود دو دستور وجود دارند تا مسیر پوشه باز شده و تعداد زیر پوشه‌های آن را به ترتیب نمایش دهند.

**مثال :** می‌خواهیم یک برنامه از نوع Standard EXE را به همراه یک فرم، یک کنترل FileListBox و ۶ کنترل برچسب به گونه‌ای طراحی کنیم که در هنگام اجرای برنامه کنترل FileListBox به‌طور پیش‌فرض اسامی و تعداد فایل‌های موجود در ریشه درایو :D را نمایش دهد. در ضمن اگر کاربر روی نام یک فایل کلیک کند نام فایل، مقدار خاصیت ListIndex و مسیر فایل را به‌وسیله کنترل‌های برچسب نمایش دهد. شکل ۳۱-۱۲ پنجره برنامه را پس از اجرا و کلیک روی فایل fact1.exe نشان می‌دهد.




**شکل ۳۱-۱۲**

پس از طراحی فرم برنامه رویداد Load فرم و رویداد Click کنترل FileListBox را به‌صورت زیر تنظیم کنید:

```
Private Sub File1_Click()
    lblname1.Caption = File1.FileName
    lblindex1.Caption = File1.ListIndex
    lblpath1.Caption = File1.Path
End Sub


Private Sub Form_Load()
    File1.FileName = "d:\"
    lblcount1.Caption = File1.ListCount
End Sub
```

همان‌طور که مشاهده می‌کنید در رویداد Form_Load ابتدا دستور File1. FileName =" d:\" مسیر پیش‌فرض را در کنترل File1 تنظیم کرده و سپس مقدار خاصیت ListCount در روی فرم

نمایش داده می‌شود. در رویداد Click کنترل File1، سه دستور مذکور به ترتیب، نام فایل، مقدار خاصیت ListIndex و مسیر فایلی را که کاربر روی آن کلیک می‌کند به‌وسیله کنترل‌های برچسب نشان می‌دهند.

**نکته**

دستور Kill می‌تواند فایل‌های مورد نظر شما را از روی دیسک پاک کند. شکل کلی این دستور به‌صورت زیر است :

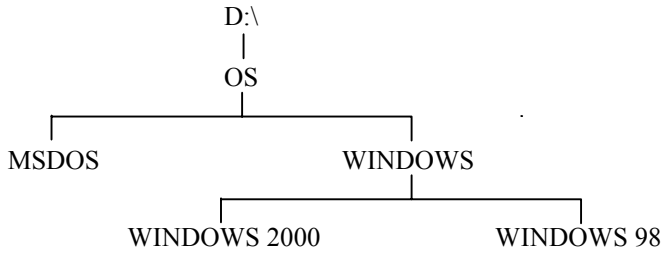
Kill ( pathname )

pathname یک مقدار رشته‌ای است که نام فایل یا فایل‌هایی را که برای حذف در نظر گرفته شده‌اند، تعیین می‌کند. مثلاً در صورتی که بخواهید فایل‌های با پسوند txt را از ریشه درایو :D حذف کنید، دستور زیر را استفاده کنید:

```
Kill ( " D: \*.txt " )
```

در صورتی که مسیر برای دستور Kill تعیین نشود، مسیر جاری به‌عنوان مسیر فایل یا فایل‌ها برای حذف در نظر گرفته می‌شود.

قبل از اجرای برنامه ابتدا این ساختار درختی را روی درایو :D ایجاد کنید:




پس از ایجاد ساختار درختی بالا، برنامه را اجرا کنید. به‌طور پیش‌فرض باید نام پوشه OS را در لیست کنترل Dir1 ببینید. روی پوشه OS دابل کلیک‌کنید. همان‌طور که مشاهده می‌کنید کنترل‌های برچسب D: \ OS را به‌عنوان مسیر جاری و عدد ۲ را به‌عنوان تعداد زیر پوشه‌های موجود در OS یعنی MSDOS و WINDOWS نشان می‌دهند.

اکنون روی OS کلیک کنید همان‌طور که می‌بینید مقدار 1- به‌عنوان ListIndex نمایش داده می‌شود. اگر روی پوشه‌های MSDOS و WINDOWS کلیک کنید به ترتیب مقادیر صفر و ۱ را مشاهده خواهید کرد و اگر روی d: \ کلیک کنید مقدار ۲- را خواهید دید.

## ۱۲-۵-۴ دستورات مدیریت پوشه‌ها

کنترل DirListBox نیز مانند کنترل DriveListBox حالت نمایشی دارد. در صورتی که شما پوشه‌ای را در کنترل باز کنید این پوشه، از نظر سیستم عامل به‌عنوان پوشه باز استفاده نخواهد شد. بنابراین شما باید از دستورات دیگری در این رابطه استفاده کنید که در این‌جا به ذکر بعضی از آن‌ها می‌پردازیم:

### ۱۲-۵-۴-۱ دستور ChDir

به‌وسیله این دستور می‌توان مسیر جاری را از نظر سیستم عامل به مسیر مورد نظر تغییر داد. شکل کلی این دستور به‌صورت زیر است:

ChDir ( path )

path یک عبارت رشته‌ای است که به مسیر مورد نظر اشاره می‌کند. به‌عنوان مثال دستور ChDir ( "D:\OS" ) مسیر جاری را در سیستم عامل به پوشه OS در درایو D: تغییر می‌دهد. برای تغییر مسیر جاری به‌وسیله یک کنترل DirListBox از دستور زیر استفاده کنید:

ChDir ( DirListBox نام کنترل. Path )

### ۱۲-۵-۴-۲ دستور MkDir

به‌وسیله این دستور می‌توان یک پوشه جدید ایجاد کرد. شکل کلی این دستور به این صورت است:

MkDir ( path )

path یک عبارت رشته‌ای است که به مسیر و نام پوشه جدید اشاره می‌کند. به‌عنوان مثال دستور MkDir ( "D: \ GAME" ) یک پوشه جدید با نام GAME در درایو D: \ ایجاد می‌کند.

| نکته |
| --- |
| ایجاد دو پوشه هم نام در یک مسیر با استفاده از این دستور سبب نمایش پیام خطا می‌شود. |

### ۱۲-۵-۴-۳ دستور RmDir

به‌وسیله این دستور می‌توان یک پوشه را حذف کرد. شکل کلی نحوه استفاده از این دستور به‌صورت زیر است:

RmDir ( path )

path یک عبارت رشته‌ای است که به مسیر و نام پوشه مورد نظر جهت حذف اشاره می‌کند. به‌عنوان مثال اگر پوشه GAME موجود باشد، دستور RmDir ( "D: \ GAME" ) آن را از روی دیسک حذف می‌کند.

**نکته**

این دستور فقط قادر به حذف پوشه‌های خالی است و در صورتی که بخواهید پوشه‌ای را که حاوی فایل یا زیر پوشه است حذف کنید پیام خطایی مشاهده خواهید کرد.

### ۱۲-۵-۴-۴ تابع CurDir

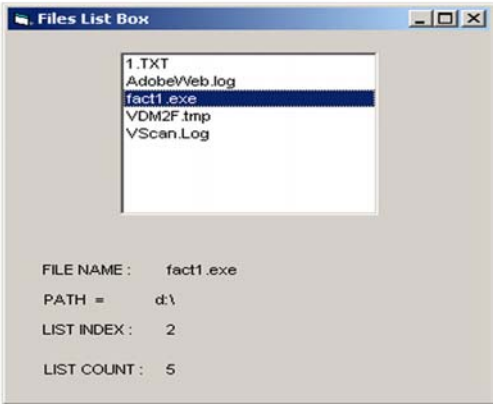
این تابع مسیر جاری را از نظر سیستم عامل به‌صورت یک عبارت رشته‌ای باز می‌گرداند. شکل کلی این تابع به‌صورت زیر است:

CurDir ( [ path ] )

path یک عبارت رشته‌ای و اختیاری است که به نام درایوی که می‌خواهید مسیر جاری در آن را به‌دست آورید، اشاره می‌کند. مثلاً دستور CurDir () مسیر جاری را در درایو جاری باز می‌گرداند و دستور CurDir ("c") مسیر جاری را در درایو c: باز می‌گرداند.

## ۱۲-۶ کنترل FileListBox

به‌وسیله این کنترل می‌توانید لیستی از فایل‌های موجود در یک‌پوشه را در روی HARD Disk، FLOPPY Disk، CD-ROM و نظایر آن‌ها مشاهده کنید و در صورت نیاز یک فایل را انتخاب کنید. در شکل ۳۲-۱۲ نمونه‌ای از این کنترل را مشاهده می‌کنید.




**شکل ۳۲-۱۲**

## ۱-۶-۱۲ خواص کنترل FileListBox

این کنترل نیز علاوه بر خواص مشترکی که با سایر کنترل‌ها دارد دارای خواص ویژه‌ای است که به توضیح هر یک از آن‌ها می‌پردازیم:

### ۱-۱-۶-۱۲ خاصیت FileName

این خاصیت نام فایلی را که در حال حاضر در کنترلFileListBox انتخاب شده است، در اختیار شما قرار می‌دهد. کاربر می‌تواند به‌وسیله کلیک کردن روی نام یک فایل، آن را انتخاب کند. شکل کلی نحوه استفاده از این خاصیت به‌صورت زیر است:

نام کنترل FileListBox . FileName [ = pathname ]

pathname یک عبارت رشته‌ای است که به مسیر و نام فایل‌هایی که باید به‌طور پیش‌فرض در کنترل نمایش داده شوند، اشاره می‌کند. در صورت عدم استفاده از این بخش نام فایل انتخاب شده در کنترل باز گردانده خواهد شد. به‌عنوان مثال در صورتی که بخواهید اسامی فایل‌های با پسوند vbp در ریشه درایو :D به‌طور پیش‌فرض نمایش داده شود، از این دستور استفاده کنید. با توجه به این که نام کنترل FileListBox ، File1 باشد.

```
File1.FileName = "D:\ *.vbp"
```

### ۲-۱-۶-۱۲ خاصیت ListCount

به‌وسیله این خاصیت می‌توان تعداد فایل‌های موجود در پوشه باز را به‌دست آورد. شکل کلی نحوه استفاده از این خاصیت به این صورت است:

نام کنترل FileListBox . ListCount

به‌عنوان مثال مقدار این خاصیت برای کنترل موجود در شکل ۳۲-۱۲، ۵ است.

### ۳-۱-۶-۱۲ خاصیت ListIndex

در کنترل FileListBox هر فایل دارای یک شماره منحصر به‌فرد است که مقدار آن برای اولین فایل، صفر ، برای دومین فایل، ۱ و به همین شکل از بالا به پایین تا آخرین فایل موجود در کنترل افزایش می‌یابد.

### ۴-۱-۶-۱۲ خاصیت Path

این خاصیت مسیر جاری را که اسامی فایل‌های آن در کنترل FileListBox نشان داده شده است، باز می‌گرداند. به‌وسیله این خاصیت می‌توانید مسیر مورد نظر خود را برای نمایش اسامی فایل‌ها

انتخاب کنید. شکل کلی نحوه استفاده از این خاصیت به‌صورت زیر است:

FileListBox نام کنترل . path [ = pathname ]

pathname یک عبارت رشته‌ای است که مسیر مورد نظر را برای نمایش اسامی فایل‌های آن در کنترل FileListBox تعیین می‌کند و در صورت عدم استفاده از آن مسیر جاری در کنترل را باز می‌گرداند.

### ۱۲-۶-۲ متدهای کنترل FileListBox

مهم‌ترین متد این کنترل SetFocus است که پس از اجرا فوکوس را به کنترل مزبور انتقال می‌دهد. نحوه استفاده از این متد به‌صورت زیر است:

FileListBox نام کنترل . SetFocus

### ۱۲-۶-۳ رویدادهای کنترل FileListBox

رویدادهای این کنترل عموماً در رابطه با سایر کنترل‌ها توضیح داده شده‌اند و از مهم‌ترین آن‌ها می‌توان به رویدادهای Click، Dbclick، GotFocus و LostFocus اشاره کرد.

## ۱۲-۷ کنترل MonthView

یکی از کنترل‌هایی که در ویژوال بیسیک جهت کار با تاریخ مورد استفاده قرار می‌گیرد، کنترل MonthView است. جهت اضافه کردن این کنترل به جعبه ابزار ویژوال بیسیک در نوار منوی ویژوال بیسیک روی منوی Project کلیک کرده و گزینه Component را انتخاب کنید؛ سپس در کادر محاوره Components روی زبانه controls کلیک کنید و روی کادر علامت Microsoft windows common controls-2 6.0 کلیک کنید. در پایان روی دکمه OK کلیک کنید تا کادر محاوره بسته شود. اکنون کنترل مزبور به جعبه ابزار اضافه شده است.

یک کنترل MonthView را روی یک فرم قرار دهید، همان‌طور که در شکل ۱۲-۳۳ مشاهده می‌کنید این کنترل دارای اجزای مختلفی است.


**شکل ۱۲-۳۳ کنترل MonthView**

همان‌طور که در شکل ۱۲-۳۳ مشاهده می‌کنید می‌توانید تاریخ مورد نظر را به‌وسیله دکمه‌های فلش‌دار نمایش داده و مشاهده کنید. به‌علاوه می‌توانید یک روز را به‌عنوان تاریخ مورد نظر انتخاب کنید. هم‌چنین تاریخ روز جاری نیز در قسمت پایین کنترل قابل مشاهده است و روز جاری نیز به‌وسیله یک منحنی قرمز رنگ در داخل کنترل نشان داده می‌شود.

## ۱۲-۷-۱ خواص کنترل MonthView

این کنترل نیز دارای خواص ویژه‌ای است که به توضیح هر یک از آن‌ها می‌پردازیم:

### ۱۲-۷-۱-۱ خاصیت Day

به‌وسیله این خاصیت می‌توان روز را به‌صورت یک عدد صحیح بین مقدار یک و ۳۱ تنظیم کرده و یا به‌دست آورد. از این خاصیت به این صورت استفاده می‌شود:

نام کنترل MonthView . Day [ = number ]

number می‌تواند یک مقدار عددی معادل یکی از روزهای ماه باشد. به‌عنوان مثال با توجه به شکل ۱۲-۳۳ فرمان PrintMonthView1.Day مقدار ۲ را نمایش می‌دهد.

### ۱۲-۷-۱-۲ خاصیت DayOfWeek

به‌وسیله این خاصیت می‌توان روز را به‌صورت یک عدد صحیح بین یک و هفت به‌دست آورد یا آن را روی روز مورد نظر تنظیم کرد. نحوه استفاده از این خاصیت به‌صورت زیر است:

نام کنترل MonthView . DayOfWeek [ = number ]

مقدار number می‌تواند یکی از مقادیر موجود در جدول ۸-۱۲ باشد. به‌عنوان مثال با توجه به شکل ۳۳-۱۲ فرمان Print monthview1. DayOfWeek مقدار ۷ را نمایش می‌دهد.

**جدول ۸-۱۲ مقادیر مربوط به خاصیت DayOfWeek**

| روز هفته | ثابت عددی | ثابت رشته‌ای |
|---|---|---|
| یکشنبه (پیش‌فرض) | 1 | mvWsunday |
| دوشنبه | 2 | mvwMonday |
| سه‌شنبه | 3 | mvwTuesday |
| چهارشنبه | 4 | mvwWednesday |
| پنجشنبه | 5 | mvwThursday |
| جمعه | 6 | mvwFriday |
| شنبه | 7 | mvwSaturday |

### ۳-۱-۷-۱۲ خواص MaxDate و MinDate

به‌وسیله این خواص می‌توان بزرگ‌ترین و کوچک‌ترین مقدار تاریخی را برای کنترل تقویم تعیین کرد یا آن‌ها را به‌دست آورد. نحوه استفاده از این خواص به‌صورت زیر است :

نام کنترل MonthView. MaxDate [ = date ]

نام کنترل MonthView. MinDate [ = date ]

date می‌تواند یک عبارت از نوع تاریخ باشد. به‌عنوان مثال این دستورات مقادیر حداکثر و حداقل را برای تقویم مورد نظر تنظیم می‌کنند.

```
MonthView1. MaxDate = # 1/1/2004 #
MonthView1. MinDate = # 1/1/2001 #
```

### ۴-۱-۷-۱۲ خاصیت MaxSelCount

به‌وسیله این خاصیت می‌توان امکان انتخاب چند روز را به‌طور هم‌زمان در تقویم ایجاد کرد. نحوه استفاده از این خاصیت به این صورت است:

نام کنترل MonthView. MaxSelCount [ = number ]

number می‌تواند عبارت عددی که بیانگر تعداد روزهای انتخابی است، باشد.

**نکته**

در صورت استفاده از این خاصیت باید خاصیت MultiSelect کنترل مورد نظر روی مقدار True تنظیم شود.

به‌عنوان مثال دستورات زیر سبب می‌شوند تا کاربر بتواند سه روز متوالی را در تقویم انتخاب کند.

```
MultiSelect = True
MonthView1. MaxSelCount = 3
```

### ۱۲-۷-۱-۵ خاصیت Month

به‌وسیله این خاصیت می‌توان ماه را به‌صورت یک عدد صحیح بین یک تا ۱۲ تنظیم کرد یا مقدار ماه را از تقویم به‌دست آورد. نحوه استفاده از این خاصیت به‌صورت زیر است:

نام کنترل MonthView. Month [ = number ]

number می‌تواند یک مقدار عددی معادل یکی از ماه‌های سال باشد. مقادیر مربوط در جدول ۱۲-۹ ارایه شده‌اند.

**جدول ۱۲-۹**

| ماه | ثابت عددی | ثابت رشته‌ای |
|---|---|---|
| January | 1 | mvwJanuary |
| February | 2 | mvwFebruary |
| March | 3 | mvwMarch |
| April | 4 | mvwApril |
| May | 5 | mvwMay |
| June | 6 | mvwJune |
| July | 7 | mvwJuly |
| August | 8 | mvwAugust |
| September | 9 | mvwSeptember |
| October | 10 | mvwOctober |
| November | 11 | mvwNovember |
| December | 12 | mvwDecember |

به‌عنوان مثال با توجه به شکل ۱۲-۳۳ فرمان Print MonthView1. Month مقدار عددی ۸ را نمایش می‌دهد.

### ۱۲-۷-۱-۶ خاصیت MonthBackColor

به‌وسیله این خاصیت می‌توانید رنگ زمینه را در تقویم تنظیم کنید. نحوه استفاده از این

خاصیت به‌صورت زیر است :

MonthView نام کنترل .MonthBackColor [ = color ]

color یک مقدار ثابت است که بیانگر رنگ مورد نظر می‌باشد.

**نکته**

برای مشاهده مقادیر ثابت رنگ‌ها به فصل گرافیک مراجعه کنید.

به‌عنوان مثال دستور MonthView1. MonthBackColor = vbBlue رنگ زمینه را در تقویم به آبی تغییر می‌دهد.

### ۷-۱-۷-۱۲ خواص MonthRows و MonthColumns

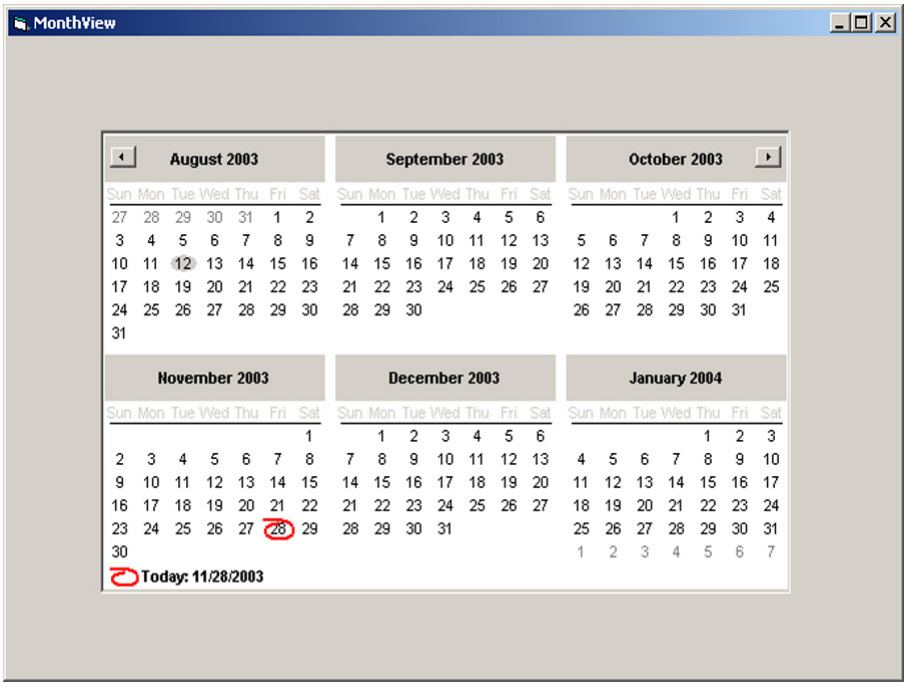
این دو خاصیت تعداد ماه‌های نمایشی در تقویم را در تعداد سطر و ستون تعیین شده، نمایش می‌دهد. نحوه استفاده از این دو خاصیت به‌صورت زیر است :

MonthView نام کنترل .MonthRows [ = number ]

MonthView نام کنترل .MonthColumns [ = number ]

number یک عبارت عددی است که تعداد سطرها و ستون‌ها را معین می‌کند. به‌عنوان مثال دستورات زیر سبب می‌شود تقویم در دو سطر و در هر سطر سه ماه از سال را نمایش دهد (مجموعاً ۶ ماه به‌طور هم‌زمان) :

```
MonthView1. MonthRows = 2
MonthView1. MonthColumns = 3
```




**شکل ۳۴-۱۲**

### ۱۲-۷-۱-۸ خاصیت ScrollRate

این خاصیت تعداد ماه‌هایی را که کاربر می‌تواند به‌وسیله دکمه‌های پیمایش موجود در تقویم به‌طور هم‌زمان به جلو یا عقب حرکت کند، معین می‌نماید. نحوه استفاده از این خاصیت به‌صورت زیر است:

نام کنترل MonthView. ScrollRate [ = number ]

number یک مقدار عددی از نوع صحیح است. به‌عنوان مثال دستور زیر سبب می‌شود تا کاربر تقویم را به‌صورت سه ماه، سه ماه پیمایش کند.

```
MonthView. ScrollRate = 3
```

### ۱۲-۷-۱-۹ خاصیت ShowToday

این‌خاصیت از نوع منطقی است و در صورتی‌که مقدار آن برابر با True باشد، تاریخ روز جاری در پایین تقویم مشاهده می‌شود و در صورتی که روی False تنظیم شده باشد نمایش داده نمی‌شود. نحوه استفاده از این خاصیت به این صورت است:

نام کنترل MonthView. ShowToday [ = boolean ]

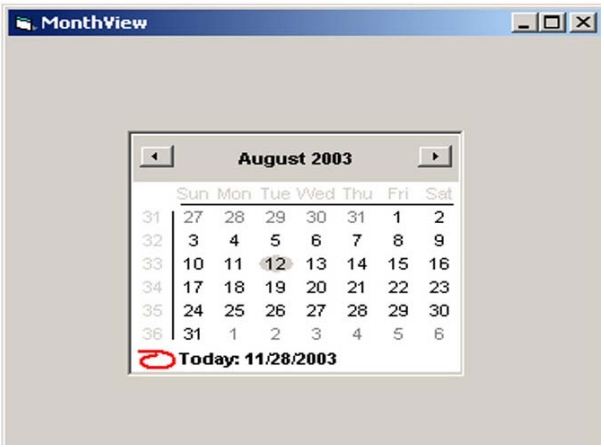
boolean یک مقدار منطقی است که می‌تواند True یا False باشد.

### ۱۲-۷-۱-۱۰ خاصیت ShowWeekNumbers

این خاصیت نیز از نوع منطقی بوده و در صورتی که مقدار آن روی True تنظیم شده باشد شماره هر هفته از سال را در سمت چپ تقویم نمایش می‌دهد. (شکل ۳۵-۱۲). نحوه استفاده از این خاصیت به‌صورت زیر است:

نام کنترل MonthView. ShowWeekNumbers [ = boolean ]

boolean می‌تواند یکی از مقادیر True یا False را کسب کند.




**شکل ۳۵-۱۲**

### ۱۲-۷-۱-۱۱ خواص TitleForeColor و TitleBackColor

این دو خاصیت می‌توانند به ترتیب رنگ قلم و رنگ زمینه را در نوار عنوان کنترل MonthView تنظیم کنند. نحوه استفاده از این دو خاصیت به‌صورت زیر است:

نام کنترل MonthView. TitleForeColor [ = color ]

نام کنترل MonthView. TitleBackColor [ = color ]

color یک ثابت است که بیانگر رنگ مورد نظر خواهد بود.

### ۱۲-۷-۱-۱۲ خاصیت Value

به‌وسیله این خاصیت می‌توان تاریخ انتخاب شده در تقویم را به‌دست آورده یا آن‌را تنظیم کرد. نحوه استفاده از این خاصیت به‌صورت زیر است:

نام کنترل MonthView. Value [ = date ]

date یک عبارت از نوع تاریخ است.

#### ۱۲-۷-۱-۱۳ خاصیت Week

به‌وسیله این خاصیت می‌توان هفته را روی یک عدد صحیح یک تا ۵۲ تنظیم کرد یا مقدار هفته را از تقویم به‌دست آورد. نحوه استفاده از این خاصیت به‌صورت زیر است:

نام کنترل MonthView. Week [ = number ]

number می‌تواند عبارت عددی که بیانگر شماره ترتیب هفته در تقویم است، باشد. به‌عنوان مثال با توجه به شکل ۳۳-۱۲ فرمان Print MonthView1. Week مقدار عددی ۳۳ را نمایش دهد.

#### ۱۲-۷-۱-۱۴ خاصیت Year

به‌وسیله این خاصیت می‌توان سال را روی یک عدد صحیح از ۱۶۰۱ تا ۹۹۹۹ تنظیم کرد یا مقدار سال را از تقویم به‌دست آورد. نحوه استفاده از این خاصیت به‌صورت زیر است:

نام کنترل MonthView. Year [ = number ]

Number می‌تواند عبارت عددی که بیانگر مقدار سال در تاریخ انتخاب شده در تقویم است، باشد. به‌عنوان مثال با توجه به شکل ۳۳-۱۲ فرمان Print MonthView1.Year مقدار عددی ۲۰۰۳ را نمایش دهد.

### ۱۲-۷-۲ متدهای کنترل MonthView

مهم‌ترین متد این کنترل SetFocus است که پس از اجرا فوکوس را به کنترل مزبور انتقال می‌دهد. نحوه استفاده از این متد به‌صورت زیر است:

نام کنترل MonthView. SetFocus

### ۱۲-۷-۳ رویدادهای کنترل MonthView

این کنترل علاوه بر رویدادهای مشترک با سایر کنترل‌ها دارای رویدادهای ویژه‌ای است که به بررسی آن‌ها می‌پردازیم.

#### ۱۲-۷-۳-۱ رویداد DateClick

این رویداد زمانی اجرا می‌شود که کاربر روی یک تاریخ در کنترل کلیک کند. این رویداد مقدار تاریخ انتخاب شده را به‌وسیله آرگومان DateClicked باز می‌گرداند. به‌عنوان مثال به‌وسیله این رویداد

می‌توان تاریخ انتخاب شده کاربر را در یک کنترل برچسب نمایش داد:

```
Private Sub MonthView1_DateClick(ByVal DateClicked As Date)
     Label1.Caption = DateClicked
End Sub
```

### ۱۲-۷-۳-۲ رویداد DateDblClick

این رویداد زمانی اجرا می‌شود که کاربر روی یک تاریخ در کنترل دابل کلیک کند. این رویداد تاریخ انتخاب شده را به‌وسیله آرگومان DateDblClick باز می‌گرداند. به‌عنوان مثال به‌وسیله رویداد زیر می‌توان تاریخ انتخاب شده کاربر را در یک کنترل برچسب نمایش داد.

```
Private Sub MonthView1_DateDblClick(ByVal DateDblClicked As _
Date)
    Label1.Caption = DateClicked
End Sub
```

### ۱۲-۷-۳-۳ رویداد SelChange

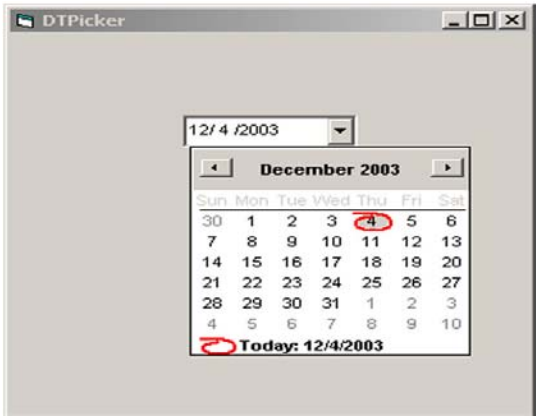
این رویداد زمانی اجرا می‌شود که کاربر تاریخ جدیدی را انتخاب کرده یا محدوده‌ای از تاریخ را در کنترل انتخاب می‌کند. این رویداد سه آرگومان دارد، آرگومان StartDate که اولین تاریخ انتخاب شده را در کنترل باز می‌گرداند، آرگومان EndDate که آخرین تاریخ انتخاب شده را در کنترل باز می‌گرداند و آرگومان Cancel که یک مقدار منطقی True یا False را باز می‌گرداند. در صورتی که مقدار این آرگومان True باشد به این معنی است که کاربر تاریخ یا محدوده تاریخی را که برگزیده لغو کرده است. به‌عنوان مثال به‌وسیله رویداد زیر می‌توان محدوده تاریخ انتخاب شده به‌وسیله کاربر را در دو کنترل برچسب نمایش داد.

```
Private Sub MonthView1_SelChange(ByVal StartDate As Date, _
   ByVal EndDate As Date, Cancel As Boolean)
   Label1.Caption = StartDate
   Label1.Caption = EndDate
End Sub
```

## ۱۲-۸ کنترل DTPicker

کنترل DTPicker (Date Time Picker) یکی از کنترل‌های دیگر در رابطه با داده‌های نوع تاریخ و ساعت می‌باشد. این کنترل علاوه بر امکانات کنترل MonthView، قابلیت تایپ تاریخ و ساعت را برای کاربر فراهم می‌کند. این کنترل کمی‌شبیه به کنترل‌های ComboBox است با این تفاوت که در صورت کلیک روی‌دکمه در کنترل، تقویم مشابه کنترل MonthView با خواص مشابه آن در اختیار

کاربر قرار می‌گیرد. نمونه‌ای از این کنترل را در شکل ۳۶-۱۲ مشاهده می‌کنید.




**شکل ۳۶-۱۲ کنترل DTPicker**

## ۱-۸-۱۲ خواص کنترل DTPicker

این کنترل خواص مشترک زیادی با کنترل MonthView دارد که در این رابطه می‌توان خواصی مانند Day ، MaxDate، MinDate ، Month، Value و Year را نام برد. به‌علاوه دارای خواص دیگری نیز می‌باشد که به توضیح آن‌ها می‌پردازیم.

### ۱-۱-۸-۱۲ خواص CalendarBackColor و CalendarForeColor

به‌وسیله این دو خاصیت می‌توانید رنگ قلم و زمینه تقویم را در کنترل DTPicker تعیین کنید. نحوه استفاده از این دو خاصیت به‌صورت زیر است:

نام کنترل DTPicker. CalendarBackColor [ = color ]

نام کنترل DTPicker. CalendarForeColor [ = color ]

color یک مقدار ثابت است که بیانگر رنگ مورد نظر می‌باشد. به‌عنوان مثال دستورات زیر رنگ قلم و زمینه تقویم را به ترتیب روی آبی و قرمز تنظیم می‌کنند.

```
DTPicker1. CalendarForeColor = vbBlue
DTPicker1. CalendarBackColor = vbRed
```

**نکته**

برای مشاهده مقادیر ثابت رنگ‌ها به فصل گرافیک مراجعه کنید.

### ۱۲-۸-۱-۲ خواص CalendarTitleForeColor و CalendarTitleBackColor

به‌وسیله این دو خاصیت می‌توان رنگ قلم و زمینه نوار عنوان تقویم را تعیین کرد. نحوه استفاده از این دو خاصیت به‌صورت زیر است:

نام کنترل DTPicker. CalendarTitleBackColor [ = color ]

نام کنترل DTPicker. CalendarTitleForecolor [ = color ]

Color یک مقدار ثابت است که بیانگر رنگ مورد نظر است. به‌عنوان مثال دستورات زیر رنگ قلم و زمینه نوار عنوان را در تقویم به ترتیب روی آبی و قرمز تنظیم می‌کند.

```
DTPicker1. CalendarTitleForeColor = vbBlue
DTPicker1. CalendarTitleBackColor = vbRed
```

### ۱۲-۸-۱-۳ خاصیت Format

به‌وسیله این خاصیت می‌توان قالب‌بندی و شکل نمایش تاریخ و ساعت را در کنترل DTPicker تعیین کرد. نحوه استفاده از این خاصیت به‌صورت زیر است:

نام کنترل DT Picker. Format [ = integer ]

integer یک ثابت عددی یا رشته‌ای است که شکل نمایش تاریخ و ساعت را در کنترل معین می‌کند. مقادیر مربوط به این خاصیت در جدول ۱۰-۱۲ ارایه شده است.

**جدول ۱۰-۱۲**

| ثابت رشته‌ای | ثابت عددی | مثال |
|---|---|---|
| DtpLongDate | 0 | Friday, Nov 14, 1972 |
| DtpShortDate | 1 | 11/14/72 |
| DtpTime | 2 | 5:31:47 PM |

### ۱۲-۸-۱-۴ خاصیت Hour

به‌وسیله این خاصیت می‌توان ساعت را به‌صورت یک عدد صحیح بین صفر تا ۲۳ تنظیم کرد یا مقدار ساعت را از کنترل به‌دست آورد. نحوه استفاده از این خاصیت به‌صورت زیر است:

نام کنترل DTPicker. Hour [ = value ]

value یک عبارت عددی از نوع صحیح است که بیانگر مقدار ساعت است.

#### ۱۲-۸-۱-۵ خاصیت Minute

به‌وسیله این خاصیت می‌توان مقدار دقیقه را به‌صورت یک عدد صحیح بین صفر تا ۵۹ تنظیم کرد یا مقدار دقیقه را از کنترل به‌دست آورد. نحوه استفاده از این خاصیت به‌صورت زیر است:

نام کنترل DTPicker. Minute [ = value ]

value یک عبارت عددی از نوع صحیح است که بیانگر مقدار دقیقه است.

#### ۱۲-۸-۱-۶ خاصیت Second

به‌وسیله این خاصیت می‌توان مقدار ثانیه را به‌صورت یک عدد صحیح بین صفر تا ۵۹ تنظیم کرد یا مقدار ثانیه را از کنترل به‌دست آورد. نحوه استفاده از این خاصیت به‌صورت زیر است:

نام کنترل DTPicker. Second [ = value ]

value یک عبارت عددی از نوع صحیح است که بیانگر مقدار ثانیه است.

### ۱۲-۸-۲ متدهای کنترل DTPicker

متدهای این کنترل شبیه به کنترل MonthView است.

### ۱۲-۸-۳ رویه‌های کنترل DTPicker

این کنترل علاوه بر رویدادهای مشترک با سایر کنترل‌ها دارای یک رویداد دیگر به نام Change است. این رویداد زمانی رخ می‌دهد که مقدار تاریخ یا زمان در کنترل DTPicker تغییر کند.

## ۱۲-۹ کنترل FlatScrollBar

ممکن است که تاکنون با نوارهای پیمایش یا همان ScrollBar در برنامه‌ها و پنجره‌های ویندوز برخورد کرده باشید و از نوارهای پیمایش جهت بزرگ و کوچک کردن تصاویر و سایر اجزای موجود در برنامه، کم و زیاد کردن مقادیر مورد نظر بین دو اندازه مشخص و نظایر آن استفاده کرده باشید. همان‌طور که در شکل ۱۲-۳۷ مشاهده می‌کنید این کنترل از دو دکمه مثلثی شکل و یک دکمه مستطیل شکل متحرک در بین دکمه‌های مثلثی تشکیل شده است. کاربر می‌تواند با کلیک روی دکمه‌های مثلثی، کشیدن دکمه مستطیل یا کلیک روی نقطه‌ای خالی از کنترل، مقدار یا اندازه مربوطه را تغییر دهد. برای اضافه کردن این کنترل به جعبه ابزار گزینه Microsoft Windows Common Controls-2 6.0 را در کادر محاوره Components انتخاب کنید.


**شکل ۳۷-۱۲ کنترل FlatScrollBar**

## ۱-۹-۱۲ خواص کنترل‌های FlatScrollBar

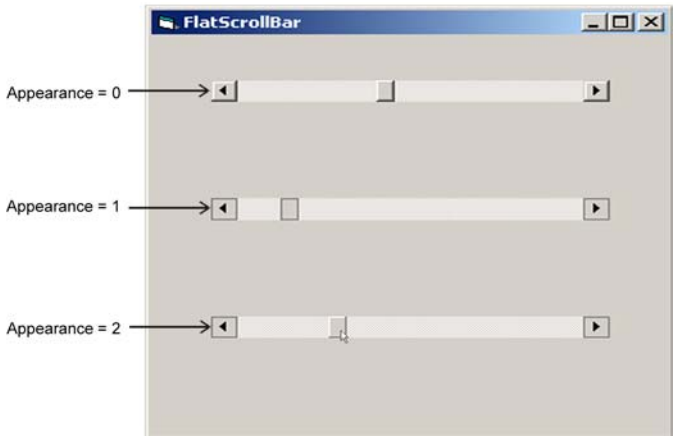
این کنترل علاوه بر خواص مشترکی که با سایر کنترل‌ها دارد، دارای خواص دیگری است که به توضیح آن‌ها خواهیم پرداخت:

### ۱-۱-۹-۱۲خاصیت Appearance

به‌وسیله این خاصیت می‌توان شکل ظاهری کنترل را تعیین کرد، اگر مقدار این خاصیت برابر با صفر (fsb3D) باشد، کنترل به‌صورت سه بعدی و اگر مقدار آن برابر با ۱ (fsbFlat) باشد، کنترل به‌صورت دو بعدی یا مسطح نمایش داده می‌شود و اگر مقدار آن برابر با ۲ (fsbTrack3D) باشد، کنترل در حالت عادی به‌صورت دو بعدی و وقتی اشاره‌گر ماوس روی دکمه‌های کنترل قرار گیرد، دکمه مربوطه به‌صورت سه بعدی نمایش داده می‌شود. (شکل ۳۸-۱۲) نحوه استفاده از این خاصیت به‌صورت زیر است:

نام کنترل FlatScrollBar. Appearance [ = integer ]

integer یک عبارت عددی از نوع صحیح یا یک ثابت رشته‌ای است که در بالا به‌صورت مجزا توضیح داده شده‌اند.




**شکل ۳۸-۱۲**

### ۱۲-۹-۱-۲ خاصیت Arrow

به‌وسیله این خاصیت می‌توان دکمه‌های مثلثی شکل ابتدا و انتهای کنترل را فعال یا غیرفعال کرد. اگر مقدار این خاصیت برابر صفر (cc2Both) باشد هر دو دکمه قابل استفاده می‌باشند و اگر مقدار این خاصیت برابر با یک (cc2LeftUp) باشد فقط دکمه سمت چپ و اگر برابر با ۲ (CC2RightDown) باشد فقط دکمه سمت راست در کنترل فعال و قابل استفاده است. نحوه استفاده از این خاصیت به صورت زیر است:

نام کنترل Flat ScrollBar .Arrows [= integer ]

integer یک عدد صحیح یا یک ثابت رشته‌ای از سه مقدار ارایه شده در رابطه با این خاصیت می‌باشد.

### ۱۲-۹-۱-۳ خواص LargeChange و SmallChange

به‌وسیله این دو خاصیت می‌توانید مقدار تغییرات را در هنگام استفاده از کنترل معین کنید. خاصیت LargeChange مقدار تغییرات را هنگامی‌که کاربر در روی مکانی از نوار پیمایش (به جز دکمه‌ها) کلیک می‌کند، معین می‌کند و خاصیت SmallChange مقدار تغییرات را هنگامی‌که کاربر روی دکمه‌های مثلثی کلیک می‌کند، تعیین می‌نماید. نحوه استفاده از این دو خاصیت به صورت زیر است:

نام کنترل Flat ScrollBar . LargeChange [= number ]

نام کنترل Flat ScrollBar . SmallChange [= number ]

number یک عبارت عددی از نوع صحیح است که میزان تغییرات را معین می‌کند. به عنوان مثال دستورات زیر باعث می‌شوند که در صورت کلیک روی نوار پیمایش مقدار مربوطه به اندازه ۱۰

واحد و اگر از دکمه‌های مثلثی استفاده شود مقدار تغییرات هر بار به اندازه سه واحد باشد.

```
FlatScrollbar. LargeChange=1
FlatScrollbar. SmallChange=3
```

#### ۱۲-۹-۱-۴ خواص Max و Min

این دو خاصیت مقدار حداکثر (رسیدن دکمه متحرک به انتهای نوار پیمایش) و مقدار حداقل (رسیدن دکمه متحرک به ابتدای نوار پیمایش) را در نوار پیمایش معین می‌کنند. نحوه استفاده از این دو خاصیت به صورت زیر است:

نام کنترل FlatScrollBar. Max [ = value ]

نام کنترل FlatScrollBar. Min [ = value ]

value یک عبارت عددی است که مقدار حداکثر و حداقل را معین می‌کند.

#### ۱۲-۹-۱-۵ خاصیت Orientation

به‌وسیله این خاصیت می‌توان نوار پیمایش را به صورت افقی یا عمودی نمایش داد. نحوه استفاده از این خاصیت به صورت زیر است:

نام کنترل FlatScrollBar. Orientation [ = integer ]

integer یک عبارت عددی یا ثابت رشته‌ای است که اگر مقدار آن صفر(CC2OrientationHorizantal) باشد کنترل به‌صورت افقی و در صورتی که مقدار آن برابر ۱ (CC2OrientationVertical) باشد، کنترل به‌صورت عمودی نمایش داده می‌شود.

#### ۱۲-۹-۱-۶ خاصیت Value

این خاصیت مقدار فعلی را در نوار پیمایش معین می‌کند. نحوه استفاده از این خاصیت به صورت زیر است:

نام کنترل FlatScrollBar. Value [ = integer ]

integer یک عبارت عددی از نوع صحیح است که مقدار حرکت در نوار پیمایش را معین می‌کند.

### ۱۲-۹-۲ متدهای کنترل FlatScrollBar

این کنترل متد ویژه‌ای ندارد و مهم‌ترین متد آن، متد SetFocus است که قبلاً در سایر کنترل‌ها به مقدار کافی در این رابطه توضیح داده شده است.

## ۱۲-۹-۳ رویدادهای کنترل FlatScrollBar

این کنترل دارای دو رویداد مهم با نام‌های Change و Scroll می‌باشد.

### ۱۲-۹-۳-۱ رویداد Change

این رویداد وقتی رخ می‌دهد که مقدار خاصیت Value در کنترل تغییر کند. فرض کنید می‌خواهیم یک فرم مطابق شکل ۱۲-۳۷ ایجاد کنیم که با استفاده از نوار پیمایش اندازه کاراکترهای عبارت FlatScrollBar کم و زیاد شود. برای این کار رویداد Load مربوط به فرم و رویداد Change کنترل نوار پیمایش را مطابق کد ارایه شده، تنظیم کنید.

```
Private Sub Form_Load()
    Fsbsize.Max = 50
    Fsbsize.Min = 10
    Fsbsize.LargeChange = 8
    Fsbsize.SmallChange = 1
    Fsbsize.Value = 20
End Sub

Private Sub Fsbsize_Change()
    lbltext.FontSize = Fsbsize.Value
End Sub
```

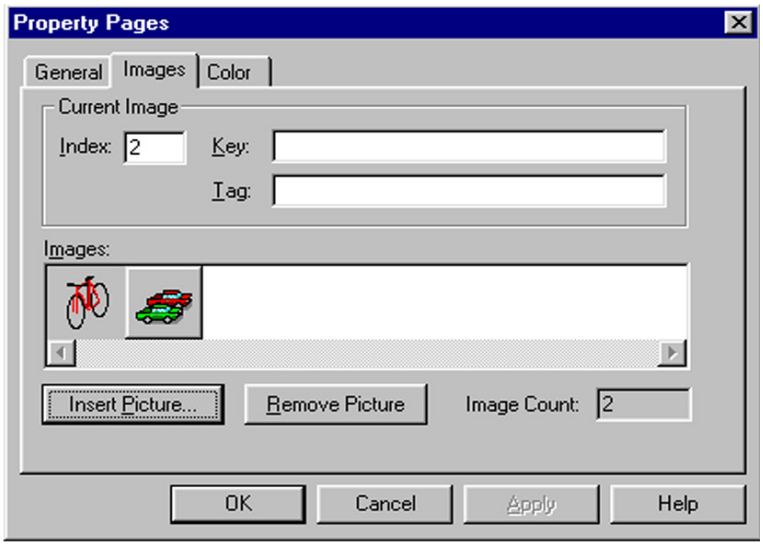
همان‌طور که مشاهده کردید ابتدا درروید‌اد Load مربوط به فرم، خواص کنترل FlatScrollBar تنظیم می‌شود. کمترین مقدار در این کنترل ۱۰ و بیشترین مقدار ۵۰ خواهد بود و میزان تغییرات بزرگ روی ۸ و میزان تغییرات کوچک روی ۱ تنظیم شده‌اند و به‌وسیله خاصیت Value اندازه اولیه کنترل، روی مقدار ۲۰ تنظیم شده است. علاوه بر این، رویداد Change کنترل طوری کد نویسی شده است تا با هر تغییر در مقدار خاصیت Value در کنترل، اندازه قلم عبارت نمایشی با مقدار خاصیت Value تنظیم شود، بدین صورت با هر تغییر در مقدار نوار پیمایش اندازه قلم نیز در عبارت تنظیم می‌شود.

### ۲-۳-۹-۱۲ رویداد Scroll

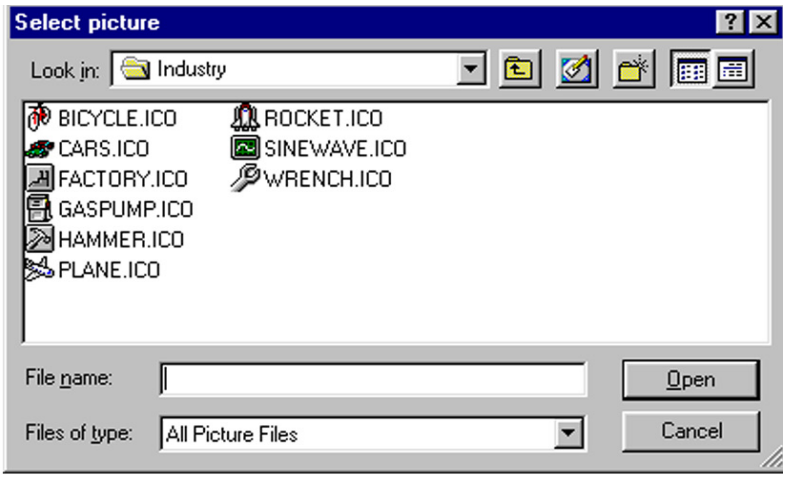
عملکرد این رویداد مشابه رویداد Change است، اما با این تفاوت که این رویداد فقط زمانی اجرا می‌شود که کاربر دکمه متحرک را در نوار پیمایش، به‌وسیله عمل Drag جابه‌جا کند. در این صورت با حرکت دکمه متحرک به‌وسیله عمل Drag بلافاصله و هم‌زمان با عمل Drag دستورات موجود در رویداد اجرا می‌شوند. می‌توانید تمرین قبل را به‌وسیله این رویداد انجام دهید و نتیجه را مقایسه کنید.

## ۱۰-۱۲ کنترل ImageList

این کنترل به تنهایی کاربردی ندارد و معمولاً برای ایجاد و طراحی کنترل‌های دیگری نظیر کنترل‌های ToolBar، ImageCombo و CoolBar مورد استفاده قرار می‌گیرند. این کنترل می‌تواند مجموعه‌ای از تصاویر را در خود نگهداری کند تا در صورت لزوم این تصاویر در سایر کنترل‌ها مورد استفاده قرار گیرند. این کنترل در هنگام اجرای برنامه دیده نمی‌شود و شامل هیچ رویدادی نیست و مهم‌ترین خاصیت آن خاصیت Custom است که به‌وسیله آن می‌توانید تصاویر خود را انتخاب و به تصاویر موجود در کنترل اضافه کنید. برای اضافه کردن این کنترل به جعبه ابزار ویژوال بیسیک در نوار منوی ویژوال بیسیک روی منوی Project کلیک کنید وگزینه Component را انتخاب نمایید و از کادر محاوره‌ای که ظاهر می‌شود زبانه Control و سپس گزینه Microsoft Windows Common Controls 6.0 را انتخاب کنید و روی دکمه OK کلیک نمایید. برای قرار دادن تصاویر در کنترل ابتدا در پنجره خواص روی دکمه  در روبه‌روی خاصیت Custom کنترل ImageList کلیک کنید، در این صورت کادر محاوره‌ای مانند شکل ۱۲-۳۹ ظاهر خواهد شد. اگر روی دکمه فرمان ...Insert Picture در زبانه Image کلیک کنید کادر محاوره دیگری مانند شکل ۱۲-۴۰ ظاهر می‌شود که به‌وسیله آن می‌توانید تصاویر مورد نیاز خود را پیدا کرده و با کلیک روی دکمه فرمان Open آن را به لیست تصاویر در کنترل اضافه کنید. هم‌چنین می‌توانید با انتخاب تصویر مورد نظر در کادر محاوره Property Pages و کلیک روی دکمه فرمان Remove Picture، تصویر مربوطه را از لیست تصاویر حذف کنید، هر یک از تصاویر به ترتیب ورود به لیست، شماره‌ای دریافت می‌کنند که از عدد یک آغاز شده و در جعبه متن Index در کادر محاوره Property Pages نمایش داده می‌شوند این عدد در زمان انتخاب تصاویر از کنترل ImageList، تصویر مورد نظر را معین می‌کند.




شکل ۱۲-۳۹




شکل ۱۲-۴۰

## ۱-۱۰-۱۲ خواص کنترل ImageList

مهم‌ترین خاصیت این کنترل خاصیت ListImage است؛ به‌وسیله این خاصیت و با استفاده از متد Add می‌توان به‌وسیله کد نویسی، تصاویر مورد نظر خود را با تابع LoadPicture به کنترل اضافه کرد. علاوه بر موارد ذکر شده برای انجام این کار باید یک متغیر از نوع ListImage نیز تعریف کرد.

به عنوان مثال به رویه رویداد ارایه شده در صفحه بعد توجه کنید. همان‌طور که مشاهده

می‌کنید با استفاده از موارد گفته شده سه تصویر از نوع ico در کنترل ImageList1 قرار می‌گیرد.

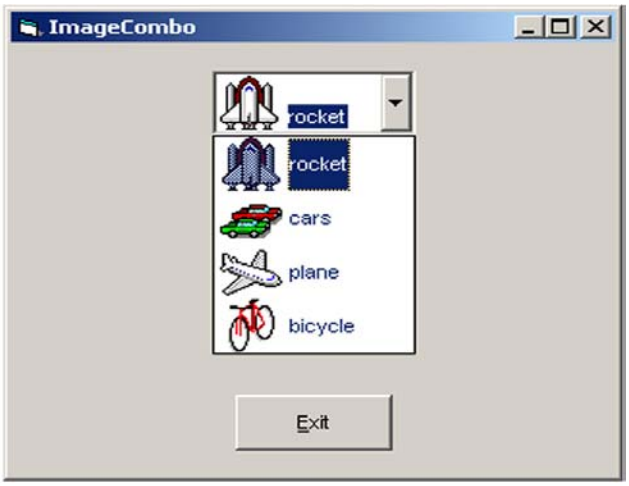
```
Private Sub Form_Load()

     Dim imgX As ListImage

     Set imgX = ImageList1.ListImages.Add(, ,
LoadPicture("D:\Program Files\Microsoft Visual
Studio\Common\Graphics\Icons\Industry\rocket.ico"))

     Set imgX = ImageList1.ListImages.Add(, ,
LoadPicture("D:\Program Files\Microsoft Visual
Studio\Common\Graphics\Icons\Industry\cars.ico"))

     Set imgX = ImageList1.ListImages.Add(, ,
LoadPicture("D:\Program Files\Microsoft Visual
Studio\Common\Graphics\Icons\Industry\plane.ico"))

     Set imgX = ImageList1.ListImages.Add(, ,
LoadPicture("D:\Program Files\Microsoft Visual
Studio\Common\Graphics\Icons\Industry\bicycle.ico"))

End Sub
```

### ۱۲-۱۰-۲ متدهای کنترل ImageList

مهم‌ترین متد این کنترل متد Add است که در بخش قبل به همراه خاصیت ListImage توضیح داده شده است. از این متد جهت اضافه کردن تصاویر به کنترل مربوطه استفاده می‌شود.

## ۱۲-۱۱ کنترل ImageCombo

تاکنون با چگونگی و نحوه ایجاد انواع لیست‌ها آشنا شده‌اید، اما گاهی اوقات لازم است که اسامی‌موجود درلیست‌ها را همراه با یک تصویرنمایش دهیم، کنترل‌هایی نظیر ListBox و ComboBox چنین امکانی را فراهم نمی‌کنند. به‌وسیله کنترل ImageCombo می‌توانید چنین لیست‌هایی را ایجاد کنید، البته برای استفاده از این کنترل باید از یک کنترل ImageList استفاده کرد. برای آن که این دو کنترل را به جعبه ابزار اضافه کنید ابتدا روی منوی Project در پنجره ویژوال بیسیک و سپس روی گزینه Components کلیک کنید در کادر محاوره‌ای که ظاهر می‌شود روی زبانه Controls و بعد از آن روی کادر علامت Microsoft Windows Common Controls 6.0 کلیک کنید، در پایان روی دکمه فرمان OK کلیک کنید. کنترل‌های فوق را به همراه چند کنترل دیگر در جعبه ابزار مشاهده خواهید کرد. نمونه‌ای از این کنترل را در شکل ۱۲-۴۱ مشاهده می‌کنید.




**شکل ۴۱-۱۲**

## ۱-۱۱-۱۲ خواص کنترل ImageCombo

این کنترل دارای چهار خاصیت ویژه است که عبارتند از: ComboItems، ImageList، Locked و SelectedItem، در این‌جا به توضیح آن‌ها خواهیم پرداخت.

### ۱-۱-۱۱-۱۲ خاصیت ComboItems

به‌وسیله این خاصیت و متد Add می‌توان تصاویر مورد نظر خود را که قبلاً در یک کنترل ImageList ذخیره کرده‌اید به گزینه‌های موجود در کنترل نسبت دهید. نحوه استفاده از این خاصیت را در مثالی که در پایان این بخش ارایه شده است، مشاهده خواهید کرد.

### ۲-۱-۱۱-۱۲ خاصیت ImageList

این‌خاصیت کنترل ImageList مربوط به‌کنترل ImageCombo را تعیین می‌کند. نحوه استفاده از این خاصیت به صورت زیر است:

نام کنترل ImageList = ImageList. نام کنترل ImageCombo

### ۳-۱-۱۱-۱۲ خاصیت Locked

این خاصیت از انواع منطقی است و وقتی مقدار آن برابر با True باشد، کاربر توانایی تایپ مقادیر خود را در لیست نخواهد داشت و اگر مقدار آن روی False تنظیم شده باشد، کاربر می‌تواند مقادیر مورد نظر خود را در کنترل تایپ کند. نحوه استفاده از این خاصیت به صورت زیر است:

ImageCombo نام کنترل . Locked [ = boolean ]

boolean یک عبارت منطقی است که می‌تواند True یا False باشد.

#### ۱۲-۱۱-۱-۴ خاصیت SelectedItem

به‌وسیله این خاصیت می‌توان نام گزینه‌ای را که کاربر انتخاب کرده است، به دست آورد. نحوه استفاده از این خاصیت به صورت زیر است:

ImageCombo نام کنترل .SelectedItem

در این‌جا ذکر این نکته ضروری است که همراه با این خاصیت می‌توان از خاصیت Index نیز استفاده کرد. خاصیت Index شماره ترتیب گزینه‌ای را که کاربر انتخاب کرده است، باز می‌گرداند شماره‌ها از یک (برای اولین گزینه) آغاز شده و به ترتیب گزینه‌ها افزایش می‌یابد. نحوه استفاده از این خاصیت به صورت زیر است:

ImageCombo نام کنترل .SelectedItem. Index

#### ۱۲-۱۱-۱-۵ خاصیت Text

این خاصیت نام گزینه‌ای را که کاربر از لیست انتخاب کرده یا مقداری را که در آن تایپ کرده است، باز می‌گرداند. علاوه بر این به‌وسیله این خاصیت می‌توان مقدار اولیه‌ای را برای نمایش در لیست تعیین کرد. نحوه استفاده از این خاصیت به صورت زیر است:

ImageCombo.Text [ = string ]

string یک عبارت از نوع رشته‌ای است.

### ۱۲-۱۱-۲ متدهای کنترل ImageCombo

مهم‌ترین متد این کنترل متد Add می‌باشد که در بخش قبل به همراه خاصیت ComboItems توضیح داده شده است. از این متدجهت اضافه کردن تصاویر به کنترل ImageCombo استفاده می‌شود.

به عنوان مثال به رویداد زیر توجه کنید:

```
Private Sub Form_Load()

     Dim imgX As ListImage

     Set imgX = ImageList1.ListImages.Add(, ,
LoadPicture("D:\Program Files\Microsoft Visual
Studio\Common\Graphics\Icons\Industry\rocket.ico"))

     Set imgX = ImageList1.ListImages.Add(, ,
LoadPicture("D:\Program Files\Microsoft Visual
```

```
Studio\Common\Graphics\Icons\Industry\cars.ico"))

     Set imgX = ImageList1.ListImages.Add(, ,
LoadPicture("D:\Program Files\Microsoft Visual
Studio\Common\Graphics\Icons\Industry\plane.ico"))

     Set imgX = ImageList1.ListImages.Add(, ,
LoadPicture("D:\Program Files\Microsoft Visual
Studio\Common\Graphics\Icons\Industry\bicycle.ico"))

     ImageCombo1.ImageList = ImageList1

     ImageCombo1.ComboItems.Add , , "rocket", 1

     ImageCombo1.ComboItems.Add , , "cars", 2

     ImageCombo1.ComboItems.Add , , "plane", 3

     ImageCombo1.ComboItems.Add , , "bicycle", 4

     ImageCombo1.ComboItems(1).Selected = True

End Sub
```

همان‌طور که مشاهده می‌کنید در این فرم از دو کنترل ImageList و ImageCombo استفاده شده است. کنترل ImageList تصاویر مورد نیاز را برای کنترل ImageCombo فراهم می‌کند برای قرار دادن تصاویر در کنترل ImageList می‌توانید از متد Add یا از خاصیت Custom در پنجره خواص استفاده کنید.

اما برای تنظیم گزینه‌های مورد نظر در ImageCombo ابتدا باید نام کنترل ImageList را در خاصیت ImageList کنترل ImageCombo ذخیره کنید، سپس به‌وسیله خاصیت ComboItems و متد Add گزینه‌ها را در لیست قرار دهید، برای این کار از دو آرگومان استفاده کرده‌ایم. توجه داشته باشید که استفاده از دو آرگومان اول در متد Add اختیاری است و در این‌جا فقط از دو آرگومان سوم و چهارم استفاده کرده‌ایم. آرگومان سوم نام گزینه در لیست و آرگومان چهارم شماره تصویر در کنترل ImageList1 است. به علاوه در آخرین فرمانی که در این رویه مشاهده می‌کنید به‌وسیله خاصیت ComboItem و خاصیت Selected که روی مقدار True تنظیم شده است، اولین گزینه (rocket) را به عنوان گزینه پیش فرض در لیست انتخاب کرده‌ایم. با اجرای این رویداد فرمی مشابه شکل ۱۲-۴۱ به‌دست می‌آید.

### ۳-۱۱-۱۲ رویدادهای کنترل ImageCombo

از رویدادهای مهم این کنترل می‌توان به رویدادهای Change، Click، LostFocus وGotFocus اشاره کرد. با این رویدادها در سایر کنترل‌ها آشنا شده‌اید و در این‌جا از ذکر مجدد توضیحات خودداری می‌کنیم. تنها در رابطه با رویداد Change ذکر این نکته ضروری است که این رویداد زمانی اجرا می‌شود که کاربر اطلاعات خود را در داخل کنترل ImageCombo تایپ کند.

به عنوان مثال به رویداد زیر که برای کنترل ImageCombo در فرم مربوط به شکل ۱۲-۴۱ تهیه شده است، توجه کنید.

```
Private Sub ImageCombo1_LostFocus()

    Print ImageCombo1.SelectedItem.Index

    Print ImageCombo1.SelectedItem

    Print ImageCombo1.Text

End Sub
```

همان‌طور که مشاهده می‌کنید در رویداد LostFocus کنترل ImageCombo از سه فرمان و خاصیت استفاده شده است. اولین خط از این رویداد با استفاده از خواص Index و SelectedItem شماره ترتیب مربوط به گزینه‌ای که کاربر از لیست کنترل انتخاب کرده است، توسط دستور Print نمایش داده می‌شود در خط دوم و سوم از این رویداد، خواص SelectedItem و Text نام گزینه انتخاب شده توسط کاربر را در روی فرم نمایش می‌دهند.

## ۱۲-۱۲ کنترل MaskedEdit

یکی دیگر از کنترل‌های ActiveX در ویژوال بیسیک کنترل MaskedEdit (یا MaskEdBox) است. این کنترل عملکردی شبیه به کنترل کادر متن یا Text Box دارد، اما با این تفاوت که توانایی دریافت اطلاعات را با یک قالب‌بندی مشخص دارد. برای اضافه کردن این کنترل به جعبه ابزار ابتدا روی منوی Project و سپس روی گزینه Components کلیک کنید و از کادر محاوره‌ای که ظاهر می‌شود، روی زبانه Controls و بعد روی کادر علامت Microsoft Masked Edit Control 6.0 کلیک کنید و سپس روی دکمه OK کلیک کنید. شکل ظاهری این کنترل در روی فرم مانند کنترل کادر متن است.

## ۱-۱۲-۱۲ خواص کنترل MaskedEdit

این کنترل علاوه بر خواص مشترکی که با سایر کنترل‌ها دارد دارای خواص دیگری نیز می‌باشد که از اهمیت به‌سزایی برخوردار است. که در این‌جا به توضیح آن‌ها می‌پردازیم.

### ۱-۱-۱۲-۱۲ خاصیت AutoTab

این خاصیت از نوع منطقی می‌باشد. اگر مقدار این خاصیت روی True تنظیم شود وقتی کاربر مقادیر خود را در داخل کنترل تایپ می‌کند در صورت پر شدن کنترل از داده، فوکوس به طور خودکار به کنترل بعدی انتقال می‌یابد، اما در صورت تنظیم این خاصیت روی مقدار False با پر شدن کنترل از داده‌ها، فوکوس روی کنترل MaskedEdit باقی خواهد ماند. نحوه استفاده از این خاصیت به صورت زیر است:

نام کنترل MaskedEdit .AutoTab [ = boolean ]

boolean یک عبارت منطقی است که می‌تواند True یا False باشد.

### ۲-۱-۱۲-۱۲ خاصیت Format

این خاصیت شکل نمایشی داده‌های ورودی را پس از آن که کنترل فوکوس را از دست می‌دهد، معین می‌کند. نحوه استفاده از این خاصیت به صورت زیر است :

نام کنترل Masked Edit .Format =value

value می‌تواند یکی از مقادیر موجود در جدول ۱۱-۱۲ باشد.

**جدول ۱۱-۱۲**

| توضیح | مقدار خاصیت | نوع داده |
|---|---|---|
| نمایش مقادیر به صورتی که وارد شده‌اند | بدون مقدار (پیش فرض) | عددی |
| نمایش مقادیر به صورت نوع پولی، استفاده از علامت کاما برای جدا کردن ارقام و نمایش اعداد منفی در داخل پرانتز | $# , ## 0.00 ; ( $#,## 00.0 ) | عددی |
| اعداد را با توجه به قسمت اعشاری گرد می‌کند. | 0 | عددی |
| از کاراکتر کاما برای جداکردن ارقام استفاده می‌شود. | # , # 0 | عددی |
| نمایش اعداد به صورت مقادیر درصدی | 0 % | عددی |
| نمایش اعداد به صورت نماد علمی | 0 .00E + 00 | عددی |

| توضیح | مقدار خاصیت | نوع داده |
|---|---|---|
| نمایش تاریخ و زمان به‌صورت معمولی | c | تاریخ و زمان |
| نمایش تاریخ و زمان با شکل طولانی (مثل Tuesday, May 26, 1992) | dddddd | تاریخ و زمان |
| نمایش تاریخ وزمان با ذکر نام ماه (مثل 26-May-92) | dd-mmm-yy | تاریخ و زمان |
| نمایش تاریخ و زمان با شکل کوتاه (مثل 2/10/80) | dddd | تاریخ و زمان |
| نمایش ساعت با شکل طولانی (مثل 06:21:42 A.M) | tttt | تاریخ و زمان |
| نمایش ساعت بدون نمایش ثانیه (مثل 06:21 A.M) | hh:mm AM/PM | تاریخ و زمان |
| نمایش ساعت با شکل کوتاه (مثل 06:21) | hh:mm | تاریخ و زمان |

### ۳-۱-۱۲-۱۲ خاصیت Mask

این خاصیت نوع کاراکتر ورودی را معین می‌کند در نتیجه می‌توان نوع داده‌های ورودی توسط کاربر را کنترل کرد. برای تعیین نوع کاراکترهای ورودی می‌توانید از مقادیر موجود در جدول ۱۲-۱۲ استفاده کنید. نحوه استفاده از این خاصیت به این صورت است:

MaskedEdit نام کنترل .Mask =value

**جدول ۱۲-۱۲**

| توضیح | کاراکتر |
|---|---|
| ورود یک کاراکتر رقمی بین صفر تا ۹ | # |
| علامت نقطه اعشار | . |
| علامت جدا کننده ارقام | , |
| علامت جدا کننده ساعت، دقیقه و ثانیه | : |
| علامت جدا کننده روز، ماه و سال | / |
| با کاراکتر بعدی مانند یک حرف رفتار می‌شود. از این کاراکتر همراه # , & , A و ؟ استفاده می‌شود. | \ |
| ورود یک مقدار کاراکتری با کد اسکی در محدوده ۳۲ تا ۱۲۶ و ۱۲۸ تا ۲۵۵ | & |
| تبدیل کاراکترهای ورودی به کاراکترهای بزرگ | > |
| تبدیل کاراکترهای ورودی به کاراکترهای کوچک | < |
| ورود یک کاراکتر حرفی A تا Z، a تا z و ارقام صفر تا ۹ | A |

| کاراکتر | توضیح |
|---|---|
| a | ورود یک کاراکتر حرفی یا رقمی |
| 9 | ورود یک کاراکتر رقمی بین صفر تا ۹ |
| c | عملکرد مشابه & دارد. |
| ? | ورود یک کاراکتر حرفی A تا Z و یا a تا z |

برای مثال اگر خاصیت Mask به صورت ###.### تنظیم شود، کاربر می‌تواند یک عدد سه رقمی با دو رقم بعد از اعشار تایپ کند و قادر به ورود سایر مقادیر نیست.

#### ۱۲-۱۲-۱-۴ خاصیت MaskLength

به‌وسیله این خاصیت می‌توان حداکثر تعداد کاراکترهای ورودی را تعیین کرد. نحوه استفاده از این خاصیت به صورت زیر است:

نام کنترل MaskedEdit . Masklength=value

value یک مقدار عددی است که می‌تواند حداکثر ۶۴ باشد.

#### ۱۲-۱۲-۱-۵ خاصیت Text

خاصیت Text مقدار وارد شده در کنترل را نگهداری می‌کند، این خاصیت شبیه به خاصیت Text در کنترل TextBox می‌باشد. نحوه استفاده از این خاصیت به این صورت است:

نام کنترل MaskedEdit .Text =[string]

string یک مقدار رشته‌ای است.

### ۱۲-۱۲-۲ متدهای کنترل MaskedEdit

این کنترل متد ویژه‌ای ندارد و مهم‌ترین متد آن ، متد SetFocus است که قبلاً در سایر کنترل‌ها در این رابطه توضیح داده شده است.

### ۱۲-۱۲-۳ رویدادهای کنترل MaskedEdit

این کنترل رویداد ویژه‌ای ندارد و مهم‌ترین رویداد آن GotFocus است که قبلاً در سایر کنترل‌ها در این رابطه توضیح داده شده است.

# ۱۲-۱۳ کنترل RichTextBox

کنترل RichTextBox یکی دیگر از کنترل‌های ویژوال بیسیک است که اجازه کار بر روی داده‌های متنی را فراهم می‌کند. این کنترل اجازه ورود و ویرایش داده‌های متنی را با امکانات بیشتری نسبت به کنترل TextBox در اختیار کاربر قرار می‌دهد. به عنوان مثال می‌توان انواع قالب‌بندی‌های متن مثل حالت Bold ، Italic ، Font ، اندازه، رنگ قلم و غیره را به بخشی از متن موجود در کنترل اعمال کرد. علاوه بر این به‌وسیله این کنترل می‌توان فایل‌هایی با قالب‌بندی RTF و متنی را باز و ذخیره کرد.

**نکته**

برای اضافه کردن کنترل RichTextBox به جعبه ابزار، کادر محاوره Components را باز کنید و گزینه Microsoft RichTextBox Control 6.0 را انتخاب کنید.

## ۱-۱۳-۱۲ خواص کنترل RichTextBox

این کنترل علاوه بر خواص مشترکی که با سایر کنترل‌ها دارد، دارای خواص دیگری نیز می‌باشد که به توضیح آن‌ها می‌پردازیم:

### ۱-۱-۱۳-۱۲ خاصیت FileName

به‌وسیله این خاصیت می‌توانید نام و مسیر فایل RTF مورد نظر را که می‌خواهید محتویات آن را در کنترل نمایش دهید، تعیین کنید. نحوه استفاده از این خاصیت به این صورت است :

نام کنترل RichTextBox.Filename=path

path یک عبارت رشته‌ای است که بیانگر مسیر و نام فایل مورد نظر می‌باشد.

### ۲-۱-۱۳-۱۲ خاصیت Locked

این خاصیت از نوع منطقی می‌باشد. اگر مقدار آن روی True تنظیم شود محتویات کنترل قابل ویرایش نیست و در صورتی که مقدار آن روی False تنظیم شود متن موجود در کنترل قابل ویرایش است. نحوه استفاده از این خاصیت به‌صورت زیر است:

نام کنترلRichTextBox.Locked=boolean

boolean یک مقدار منطقی است که می‌تواند True یا False باشد.

### ۱۲-۱۳-۱-۳ خاصیت MaxLength

این خاصیت حداکثر تعداد کاراکترها را در کنترل معین می‌کند. مقدار پیش فرض صفر است که با توجه به میزان حافظه سیستم، حداکثر تعداد کاراکترها معین می‌شود. نحوه استفاده از این خاصیت به‌صورت زیر است:

نام کنترلRichTextBox.MaxLengh=long

long یک مقدار عددی از نوع اعداد صحیح بلند (Long Integer) می‌باشد.

### ۱۲-۱۳-۱-۴ خاصیت MultiLine

این خاصیت از نوع منطقی است. اگر مقدار این خاصیت روی True تنظیم شده باشد کنترل توانایی دریافت و نمایش متن را در چندین خط دارد، ولی اگر مقدار این خاصیت روی False تنظیم شود از یک خط جهت نمایش متن استفاده می‌شود. نحوه استفاده از این خاصیت به‌صورت زیر است:

نام کنترلRichTextBox.MultiLine=boolean

boolean یک مقدار منطقی است که می‌تواند True یا False باشد.

### ۱۲-۱۳-۱-۵ خاصیت ScrollBars

به‌وسیله خاصیت ScrollBars می‌توانید نوارهای پیمایش افقی و عمودی را در کنترل RichTextBox فعال کنید. نحوه استفاده از این خاصیت به صورت زیر است:

نام کنترلRichTextBox.ScrollBars=value

مقدار value می‌تواند یکی از مقادیر ثابت رشته‌ای یا ثابت عددی موجود در جدول ۱۳-۱۲ باشد.

**جدول ۱۳-۱۲**

| مثال | ثابت عددی | ثابت رشته‌ای |
|---|---|---|
| کنترل بدون نوار پیمایش | 0 (Default) | rtfNone |
| کنترل با نوار پیمایش افقی | 1 | rtfHorizontal |
| کنترل با نوار پیمایش عمودی | 2 | rtfVertical |
| کنترل با هر دو نوار پیمایش | 3 | rtfBoth |

**نکته**

قبل از استفاده این خاصیت جهت نمایش نوارهای پیمایش، خاصیت MultiLine را روی مقدار True تنظیم کنید.

#### ۱۲-۱۳-۱-۶ خاصیت TextRTF

این خاصیت شبیه به خاصیت Text در کنترل TextBox و MaskedEdit است ومتن موجود در کنترل را تنظیم و نگهداری می‌کند. نحوه استفاده از این خاصیت به صورت زیر است:

نام کنترلRichTextBox . TextRTF=string

string یک عبارت رشته‌ای است که به متن موجود در کنترل اشاره می‌کند.

## ۱۲-۱۳-۲ متدهای کنترل RichTextBox

این کنترل دارای دو متد ویژه است که امکان نمایش محتویات یک فایل به‌وسیله کنترل یا ذخیره سازی محتویات کنترل در یک فایل RTF و یا متنی را فراهم می‌آورد.

#### ۱۲-۱۳-۲-۱ متد LoadFile

به‌وسیله این متد می‌توان محتویات یک فایل متنی یا RTF را در کنترل نمایش داد. نحوه استفاده از این متد به‌صورت زیر است:

نام کنترلRichTextBox.LoadFile(pathname, filetype)

pathname یک عبارت رشته‌ای است که به نام و مسیر فایل اشاره می‌کند و filetype می‌تواند یکی از مقادیر موجود در جدول ۱۲-۱۴ باشد.

**جدول ۱۲-۱۴**

| توضیح | ثابت عددی | ثابت رشته‌ای |
|---|---|---|
| فایل RTF (پیش فرض) | ۰ | rtfRTF |
| فایل متنی | ۱ | rtfText |

به عنوان مثال فرمان زیر محتویات فایل Test.rtf را در کنترل بارگذاری می‌کند.

```
rtftext.LoadFile    ("D:\Test.rtf", rtfRTF )
```

#### ۱۲-۱۳-۲-۲ متد SaveFile

به‌وسیله این متد می‌توان محتویات موجود در کنترل را به‌صورت یک فایل RTF یا متنی در روی دیسک ذخیره کرد. نحوه استفاده از این خاصیت به‌صورت زیر است:

نام کنترلRichTextBox.SaveFile(pathname, filetype)

آرگومان pathname یک عبارت رشته‌ای است که به نام و مسیر فایل اشاره می‌کند و filetype می‌تواند یکی از مقادیر موجود در جدول ۱۴-۱۲ باشد.
به عنوان مثال فرمان بعد محتویات خاصیت Text را در فایل Text.rtf در ریشه درایو D:\ ذخیره می‌کند:

```
rtftext.SaveFile ("D:\Test.rtf" ،rtfRTF)
```

### ۱۲-۱۳-۳ رویدادهای کنترل RichTextBox

این کنترل رویداد ویژه‌ای ندارد و مهم‌ترین رویدادهای آن قبلاً در سایر کنترل‌ها توضیح داده شده‌اند.

## ۱۲-۱٤ کنترل‌های HScrollBar و VScrollBar

قبلاً با نوار پیمایش FlatScrollBar و کاربرد آن آشنا شده‌اید، در این‌جا دو نوع دیگر از این نوع کنترل را تشریح می‌کنیم. نوار پیمایش افقی HScrollBar و نوار پیمایش عمودی VScrollBar. این دو کنترل از نظر بعضی از خواص و رویدادها شبیه به کنترل FlatScrollBar هستند با این حال به ذکر مهم‌ترین خواص این دو کنترل می‌پردازیم.

### ۱۲-۱۴-۱ خواص کنترل‌های نوار پیمایش

این دو کنترل علاوه بر خواص مشترکی که با سایر کنترل‌ها دارند دارای خواص دیگری نیز هستند. که به توضیح آن‌ها می‌پردازیم.

#### ۱۲-۱۴-۱-۱ خاصیت LargeChange

این خاصیت میزان تغییرات را هنگامی که کاربر روی مکانی به جز دکمه‌های ابتدا و انتهای کنترل در نوار پیمایش کلیک کند، تعیین می‌نماید. نحوه استفاده از این خاصیت به صورت زیر است :

[=number] .LargeChange نام کنترل نوار پیمایش

number یک عبارت عددی از نوع صحیح است که میزان تغییرات را معین می‌کند.

#### ۱۲-۱۴-۱-۲ خاصیت Max

این خاصیت بیانگر حداکثر مقداری است که با رسیدن دکمه متحرک کنترل به انتهای نوار پیمایش به‌دست می‌آید. نحوه استفاده از این خاصیت به صورت زیر است:

Max [=value]. نام کنترل نوار پیمایش

value یک عبارت عددی است که مقدار حداکثر را در نوار پیمایش تعیین می‌کند.

#### ۱۲-۱۴-۱-۳ خاصیت Min

این خاصیت بیانگر حداقل مقداری است که با رسیدن دکمه متحرک کنترل به ابتدای نوار پیمایش به‌دست می‌آید. نحوه استفاده از این خاصیت به صورت زیر است:

Min [=value] .نام کنترل نوار پیمایش

value یک عبارت عددی است که مقدار حداقل را در نوار پیمایش تعیین می‌کند.

#### ۱۲-۱۴-۱-۴ خاصیت SmallChange

این خاصیت میزان تغییرات را هنگامی که کاربر روی دکمه‌های مثلثی در ابتدا و انتهای کنترل‌ها کلیک کند، تعیین می‌نماید. نحوه استفاده از این خاصیت به صورت زیر است :

SmallChange [=number]. نام کنترل نوار پیمایش

number یک عبارت عددی از نوع صحیح است که میزان تغییرات را معین می‌کند.

#### ۱۲-۱۴-۱-۵ خاصیت Value

این خاصیت مقدار کنونی را در نوار پیمایش معین می‌کند. نحوه استفاده از این خاصیت به صورت زیر است :

Value [=integer]. نام کنترل نوار پیمایش

integer یک عبارت عددی از نوع صحیح است که مقدار حرکت در نوار پیمایش را معین می‌کند.

### ۱۲-۱۴-۲ متدهای کنترل‌های نوار پیمایش

این کنترل‌ها فاقد متد ویژه‌ای هستند و سایر متدهای آن‌ها نیز در سایر کنترل‌ها توضیح داده شده‌اند.

### ۱۲-۱۴-۳ رویدادهای کنترل‌های نوار پیمایش

این کنترل‌ها دارای دو رویداد Change و Scroll هستند که به توضیح هر یک از آن‌ها می‌پردازیم.

#### ۱-۳-۱۴-۱۲ رویداد Change

این رویداد وقتی اجرا می‌شود که مقدار خاصیت Value کنترل نوار پیمایش تغییر کند.

#### ۲-۳-۱۴-۱۲ رویداد Scroll

عملکرد این رویداد مشابه رویداد Change است، تنها تفاوتی که بین این دو رویداد وجود دارد در این است که این رویداد زمانی اجرا می‌شود که کاربر با انجام عمل Drag دکمه متحرک نوار پیمایش را جابه‌جا کند.

## ۱۵-۱۲ کنترل Slider

یکی دیگر از کنترل‌های ActiveX در ویژوال بیسیک، کنترل Slider است. به‌وسیله این کنترل می‌توانید مقادیر مورد نظر خود را برای بخش‌های مختلف برنامه، قابل تنظیم کنید. عملکرد این کنترل تا حدودی شبیه به کنترل های نوار پیمایش است. برای تغییر مقدار در این کنترل می‌توانید روی کنترل و درجات تقسیم‌بندی کلیک کنید یا به‌وسیله فشردن کلیدهای PAGEUP یا PAGEDOWN مقدار آن را تغییر دهید، به‌علاوه می‌توانید این کار را به‌وسیله کلیدهای جهت دار چپ، راست یا انجام عمل Drag روی دکمه تنظیم مقدار در روی کنترل انجام دهید.

| نکته |
| --- |
| برای اضافه کردن این کنترل به جعبه ابزار در کادر محاوره Components گزینه Microsoft Windows Common Controls 6.0 را انتخاب کنید. |

### ۱-۱۵-۱۲ خواص کنترل Slider

این کنترل دارای خواص مشابهی مانند کنترل‌های نوار پیمایش است، به‌علاوه دارای خواص ویژه‌ای نیز می‌باشد که به توضیح آن‌ها می‌پردازیم.

#### ۱-۱-۱۵-۱۲ خاصیت BorderStyle

به‌وسیله این خاصیت می‌توان یک کادر در اطراف کنترل نمایش داد. اگر مقدار این خاصیت 0-CCNone باشد کنترل بدون کادر و اگر مقدار آن روی 1- CCFixedSingle تنظیم شده باشد کنترل به همراه یک کادر نمایش داده می‌شود.

### ۱۲-۱۵-۱-۲ خاصیت LargeChange

این خاصیت میزان تغییرات را هنگامی که کاربر در مکانی روی کنترل، عمل کلیک انجام می‌دهد یا کلیدهای PAGEUP یا PAGEDOWN را می‌فشارد، تعیین می‌کند. نحوه استفاده از این خاصیت به صورت زیر است:

نام کنترل Slider.LargChange[=number]

number یک عبارت عددی از نوع صحیح بلند است که میزان تغییرات را در کنترل تعیین می‌کند.

### ۱۲-۱۵-۱-۳ خاصیت SmallChange

این خاصیت میزان تغییرات را هنگامی که کاربر دکمه تنظیم مقدار در روی کنترل را Drag کرده یا کلیدهای جهت دار چپ یا راست را می‌فشارد، تعیین می‌کند. نحوه استفاده از این خاصیت به صورت زیر است:

نام کنترل Slider.SmallChange[=number]

number یک عبارت عددی از نوع صحیح است که میزان تغییرات را معین می‌کند.

### ۱۲-۱۵-۱-۴ خاصیت Max

این خاصیت بیانگر حداکثر مقداری است که با رسیدن دکمه متحرک کنترل به انتهای کنترل به‌دست می‌آید، نحوه استفاده از این خاصیت به این صورت است:

نام کنترل Slider.Max [=value]

value یک عبارت عددی از نوع صحیح است که می‌تواند بین ۳۲۷۶۷- تا ۳۲۷۶۷ باشد.

### ۱۲-۱۵-۱-۵ خاصیت Min

این خاصیت بیانگر حداقل مقداری است که با رسیدن دکمه متحرک کنترل به ابتدای کنترل به‌دست می‌آید. نحوه استفاده از این خاصیت به صورت زیر است:

نام کنترل Slider.Min[=value]

value یک عبارت عددی از نوع صحیح است که می‌تواند بین ۳۲۷۶۷- تا ۳۲۷۶۷ باشد.

### ۱۲-۱۵-۱-۶ خاصیت TickFrequency

این خاصیت فاصله بین درجه‌بندی‌ها را در کنترل معین می‌کند. به‌عنوان مثال اگر دامنه تغییرات بین صفر تا ۱۰۰ باشد و مقدار این خاصیت روی ۲ تنظیم شده باشد با حرکت از یک

درجه‌بندی به درجه‌بندی بعدی، دو واحد به مقدار قبلی اضافه خواهد شد. نحوه استفاده از این خاصیت به صورت زیر است:

نام کنترل Slider.TickFrequency[=number]

number یک عبارت عددی است.

#### ۱۲-۱۵-۱-۷ خاصیت TickStyle

این خاصیت نحوه قرار گرفتن و نمایش درجه‌بندی‌ها را روی کنترل تعیین می‌کند. نحوه استفاده از این خاصیت به صورت زیر است:

نام کنترل Slider.TickStyle [=number]

number یک عدد صحیح یا ثابت رشته‌ای است. مقادیر مربوط به این خاصیت در جدول ۱۵-۱۲ ارایه شده‌اند.

**جدول ۱۵-۱۲**

| ثابت رشته‌ای | ثابت عددی | توضیح |
|---|---|---|
| sldBottomRight | 0 | نمایش درجه‌بندی در پایین کنترل |
| sldTopLeft | 1 | نمایش درجه‌بندی در بالای کنترل |
| sldBoth | 2 | نمایش درجه در بالا و پایین کنترل |
| sldNoTicks | | کنترل بدون نمایش درجه‌بندی |

#### ۱۲-۱۵-۱-۸ خاصیت Value

این خاصیت مقدار فعلی را در کنترل Slider تعیین می‌کند. نحوه استفاده از این خاصیت به صورت زیر است:

نام کنترل Slider.Value [=integer]

integer یک عبارت عددی از نوع صحیح است که مقدار حرکت را در کنترل معین می‌کند.

### ۱۲-۱۵-۲ متدهای کنترل Slider

این کنترل دارای یک متد ویژه با نام GetNumTicks است.

#### ۱-۲-۱۵-۱۲ متد GetNumTicks

این متد تعداد درجه‌بندی‌های بین مقدار حداقل و حداکثر را در کنترل باز می‌گرداند، نحوه استفاده از این متد به صورت زیر است:

GetNumTicks. نام کنترل Slider

## ۳-۱۵-۱۲ رویدادهای کنترل Slider

این کنترل مانند کنترل‌های نوار پیمایش دارای دو رویداد مهم Change و Scroll است که به توضیح آن‌ها می‌پردازیم.

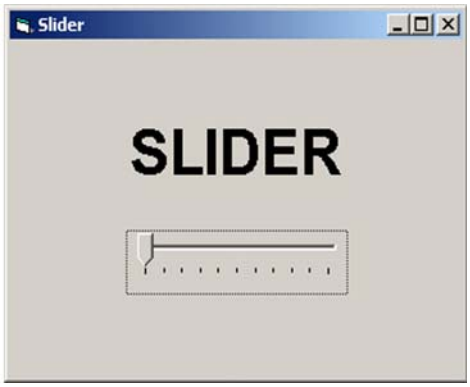
#### ۱-۳-۱۵-۱۲ رویداد Change

این رویداد وقتی اجرا می‌شود که مقدار خاصیت Value کنترل Slider تغییر کند.

#### ۲-۳-۱۵-۱۲ رویداد Scroll

عملکرد این رویداد مشابه رویداد Change است تنها تفاوتی که بین این دو رویداد وجود دارد در این است که این رویداد زمانی اجرا می‌شود که کاربر به‌وسیله کلیدهای جهت دار چپ و راست یا با انجام عمل Drag روی دکمه متحرک کنترل مقدار مورد نظر خود را انتخاب کند.

**مثال :** می‌خواهیم در یک فرم اندازه یک عبارت را به‌وسیله کنترل Slider تنظیم کنیم، بنابراین مطابق شکل ۴۲-۱۲ یک فرم ایجاد کنید و رویدادهای Change و Scroll کنترل Slider و رویداد Load فرم را به این صورت تنظیم نمایید.




**شکل ۴۲-۱۲**

```
Private Sub Form_Load()
     sldsize.Max = 50
     sldsize.Min = 10
     sldsize.TickFrequency = 5
     sldsize.Value = 20
     lblslider.FontSize = 20
     lblslider.Alignment = vbCenter
End Sub

Private Sub sldsize_Change()
     lblslider.FontSize = sldsize.Value
End Sub

Private Sub sldsize_Scroll()
     lblslider.FontSize = sldsize.Value
End Sub
```

## ۱۶-۱۲ کنترل UpDown

این کنترل یکی دیگر از کنترل‌های ActiveX ویژوال بیسیک است که به‌وسیله آن می‌توان مقادیر مربوطه به کنترل‌های دیگر را افزایش یا کاهش داد. به عنوان مثال تنظیم اندازه قلم یا تعیین مدت زمان لازم برای فعال شدن یک محافظ صفحه نمایش و نظایر آن. این کنترل معمولاً به کنترل دیگری مربوط می‌شود تا تغییرات مقادیر در کنترل UpDown روی کنترل دوم اعمال شود، به کنترل دوم BuddyControl می‌گویند.

**نکته**

برای اضافه کردن این کنترل به جعبه ابزار در کادر محاوره Components گزینه Microsoft Windows Common Controls -2 5.0 را انتخاب کنید.

## ۱-۱۶-۱۲ خواص کنترل UpDown

این کنترل علاوه بر خواص مشترکی که با سایر کنترل‌ها دارد، دارای خواص منحصر به فرد دیگری نیز می‌باشد که به توضیح آن‌ها می‌پردازیم.

### ۱-۱-۱۶-۱۲ خاصیت Alignment

این خاصیت نحوه قرار گرفتن کنترل UpDown را نسبت به کنترل BuddyControl تعیین می‌کند. در صورتی که بخواهید این کنترل در سمت چپ کنترل BuddyControl قرار بگیرد، مقدار این خاصیت را روی 1-CC2AlighmentRight و در صورتی که بخواهید این کنترل در سمت راست کنترل BuddyControl قرار بگیرد، مقدار این خاصیت را روی 1-CC2AlighmentRight تنظیم کنید. نحوه استفاده از این خاصیت به صورت زیر است:

نام کنترل UpDown.Alignment [=value]

value می‌تواند یکی از مقادیر ارایه شده باشد.

### ۲-۱-۱۶-۱۲ خاصیت BuddyControl

به‌وسیله این خاصیت کنترل UpDown به کنترلی که نام آن به این خاصیت نسبت داده شده است، متصل می‌شود. نحوه استفاده از این خاصیت به صورت زیر است:

نام کنترل UpDown.BuddyControl [=value]

value نام کنترلی است که کنترل UpDown به آن متصل می‌شود.

### ۳-۱-۱۶-۱۲ خاصیت BuddyProperty

این خاصیت، تعیین می‌کند که چه خاصیتی از کنترل BuddyControl با کنترل UpDown، منطبق و همگام شود. نحوه استفاده از این خاصیت به صورت زیر است:

نام کنترل UpDown.BuddyProperty [=value]

value نام خاصیتی از کنترل BuddyControl است که با کنترل UpDown متصل شده است.

### ۴-۱-۱۶-۱۲ خاصیت Increment

این خاصیت میزان افزایش مقدار خاصیت Value را در کنترل UpDown تعیین می‌کند. از این خاصیت زمانی استفاده می‌شود که کاربر روی دکمه‌های کنترل UpDown کلیک کند. نحوه استفاده از این خاصیت به این صورت است:

نام کنترل UpDown.Increment [=value]

value یک مقدار عددی از نوع صحیح است که میزان افزایش مقدار خاصیت value را در کنترل معین می‌کند.

### ۱۲-۱۶-۱-۵ خاصیت Max

این خاصیت بیانگر حداکثر مقداری است که کنترل UpDown می‌تواند به آن اشاره کند. نحوه استفاده از این خاصیت به صورت زیر است:

نام کنترل UpDown.Max [=value]

value یک مقدار عددی از نوع صحیح است که حداکثر مقدار در کنترل UpDown را تعیین می‌کند.

### ۱۲-۱۶-۱-۶ خاصیت Min

این خاصیت حداقل مقداری را که کنترل UpDown می‌تواند کسب کند، تعیین می‌کند. نحوه استفاده از این خاصیت به صورت زیر است:

نام کنترل UpDown.Min [=value]

value یک مقدار عددی از نوع صحیح است که حداقل مقدار در کنترل UpDown را تعیین می‌کند.

### ۱۲-۱۶-۱-۷ خاصیت Orientation

این خاصیت نحوه قرار گرفتن دکمه‌های کنترل را تعیین می‌کند. اگر مقدار این خاصیت روی 0-cc2OrientationVertical تنظیم شود، دکمه‌ها به‌صورت عمودی و اگر روی مقدار 0-cc2OrientationHorizontal تنظیم شود، دکمه‌ها به‌صورت افقی قرار می‌گیرند نحوه استفاده از این کنترل به صورت زیر است:

نام کنترل UpDown.Orientation. [=value]

value می‌تواند یکی از مقادیر ارایه شده باشد.

### ۱۲-۱۶-۱-۸ خاصیت SyncBuddy

به‌وسیله این‌خاصیت می‌توان‌خاصیت Value کنترل UpDown را با خاصیتی از BuddyControl که در خاصیت BuddyProperty انتخاب شده است، منطبق و هماهنگ کرد. این خاصیت از نوع منطقی است و در صورت تنظیم آن روی مقدار True با افزایش یا کاهش مقدار خاصیت Value، خاصیت انتخاب شده در BuddyControl نیز تغییر می‌کند. نحوه استفاده از این خاصیت به صورت زیر است:

نام کنترل UpDown . SyncBuddy [=value]

value یک مقدار منطقی است‌که می‌تواند True یا False باشد. در صورت انتخاب مقدار True خاصیت انتخاب شده از BuddyControl با مقدار خاصیت Value در کنترل UpDown هماهنگ می‌شود.

#### ۹-۱-۱۶-۱۲ خاصیت Value

این خاصیت مقدار کنونی را در کنترل UpDown تعیین می‌کند. نحوه استفاده از این خاصیت به‌صورت زیر است:

نام کنترل UpDown=Value[integer]

integer یک عبارت عددی از نوع صحیح است که مقدار تعیین شده توسط کنترل را معین می‌کند.

#### ۱۰-۱-۱۶-۱۲ خاصیت Wrap

این خاصیت از نوع منطقی است. اگر مقدار این خاصیت روی True تنظیم شود، پس از رسیدن مقدار خاصیت Value کنترل UpDown به مقدار تعیین شده در خاصیت Max و پس از عبور از آن، مقدار خاصیت Value روی مقدار تعیین شده در خاصیت Min تنظیم می‌شود. همین‌طور با رسیدن به مقدار Min و عبور این محدوده، مقدار خاصیت Value روی مقدار تعیین شده در خاصیت Max تنظیم می‌شود، به عبارت دیگر کاربر قادر به حرکت از محدوده حداکثر به حداقل و بالعکس خواهد بود.

در صورتی که این خاصیت روی مقدار False تنظیم شود با رسیدن به مقدار Max و Min افزایش یا کاهش مقدار خاصیت Value متوقف می‌شود. نحوه استفاده از این خاصیت به صورت زیر است:

نام کنترل UpDown . Warp [=boolean]

boolean یک مقدار منطقی است که می‌تواند True یا False باشد.

### ۲-۱۶-۱۲ متدهای کنترل UpDown

این کنترل متد خاصی ندارد.

### ۳-۱۶-۱۲ رویداد های کنترل UpDown

این کنترل دارای سه رویداد Change ، Downclick ، Upclick است.

#### ۱-۳-۱۶-۱۲ رویداد Change

این رویداد وقتی اجرا می‌شود که مقدار خاصیت Value کنترل UpDown تغییر کند.

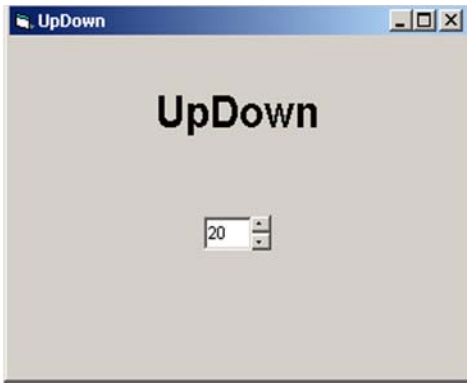
### ۱۲-۱۶-۳-۲ رویداد DownClick

این رویداد وقتی اجرا می‌شود که کاربر روی دکمه سمت چپ یا دکمه پایینی کنترل UpDown کلیک کند.

### ۱۲-۱۶-۳-۳ رویداد UpClick

این رویداد وقتی اجرا می‌شود که کاربر روی دکمه سمت راست یا دکمه بالایی کنترل UpDown کلیک کند.

**مثال** : می‌خواهیم مطابق شکل ۱۲-۴۳ اندازه قلم را در یک کنترل برچسب به‌وسیله یک کنترل UpDown که با یک کنترل TextBox مرتبط شده است، تنظیم کنیم.




**شکل ۱۲-۴۳**

پس از طراحی فرم مطابق شکل ۱۲-۴۳، رویدادهای مورد نظر را مطابق کدهای زیر تنظیم کنید. همان‌طور که مشاهده می‌کنید در رویداد Load فرم، خواص اولیه کنترل updsize و سایر کنترل‌ها تنظیم می‌شود. به علاوه با استفاده از رویدادهای Change و کنترل‌های updsize و txtsize سعی شده است تا کاربر با استفاده از هر دو کنترل بتواند اندازه قلم را در کنترل برچسب تغییر دهد.

```
Private Sub Form_Load()
    updsize.Alignment = cc2AlignmentRight
    updsize.Increment = 2
    updsize.SyncBuddy = True
    updsize.Wrap = True
```

```
    updsize.Max = 50
    updsize.Min = 10
    updsize.Value = 20
    txtsize.Text = "20"
    lblupdown.FontSize = 20
    lblupdown.Alignment = vbCenter
End Sub

Private Sub txtsize_Change()
    If Val(txtsize.Text) < 10 Then
        lblupdown.FontSize = 10
    Else
        lblupdown.FontSize = txtsize.Text
    End If
End Sub

Private Sub updsize_Change()
    lblupdown.FontSize = updsize.Value
End Sub
```

**نکته**

خواص BuddyControl ، BuddyProperty و Orientation را در زمان طراحی فرم و از طریق پنجره خواص به ترتیب روی مقادیر txtsize و Text و صفر تنظیم کنید.

## ۱۲-۱۷ رابط‌های گرافیکی چند سندی MDI

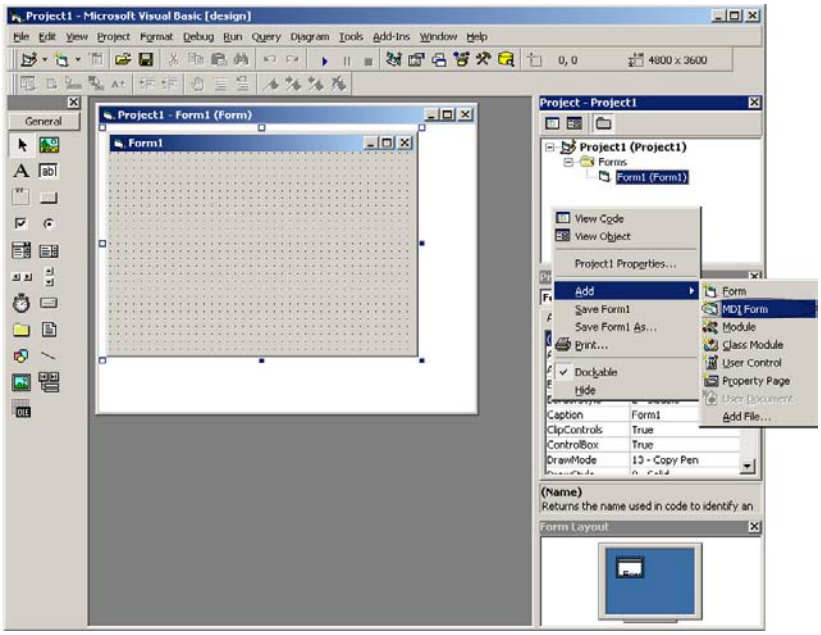
تاکنون معمولاً پروژه‌هایی را طراحی کرده‌اید که از یک فرم تشکیل شده‌اند، اما در برنامه نویسی واقعی معمولاً بیش از یک فرم مورد استفاده قرار می‌گیرد. در جلد اول آموختید که چگونه از چند فرم در یک پروژه استفاده کنید، به چنین برنامه‌هایی، برنامه‌های با رابط‌های گرافیکی یک سندی یا SDI (Single Document Interface) گفته می‌شود. در چنین برنامه‌هایی فرم‌ها به‌صورت مستقل از یکدیگر عمل می‌کنند و با بستن یا تغییر مکان یک فرم، فرم‌های دیگر بدون هیچ‌گونه عکس العملی باقی می‌مانند.

تفاوتی که بین یک رابط گرافیکی از نوع MDI با SDI وجود دارد، این است که در رابط گرافیکی MDI یک فرم به عنوان فرم اصلی و مادر وجود دارد که سایر فرم‌ها از آن تبعیت می‌کنند، یعنی با بسته شدن فرم مادر، سایر فرم‌ها (فرم‌های فرزند) نیز بسته می‌شوند؛ اما عکس این حالت صدق نمی‌کند. فرم‌های فرزند فقط می‌توانند در محدوده فرم مادر جابه‌جا شوند. علاوه بر این نمی‌توانید از کنترل‌ها در روی فرم مادر استفاده کنید.

نمونه بارزی از این‌گونه نرم افزارها، مجموعه نرم افزارهای Office و برنامه ویژوال بیسیک است. شما در پنجره برنامه ویژوال بیسیک می‌توانید پنجره چندین پروژه و فرم را باز کرده و مورد استفاده قرار دهید یا برنامه Word شرکت مایکروسافت که می‌تواند چندین سند را به‌طور هم‌زمان در پنجره‌های جداگانه باز کند تا کاربر با هر یک از آن‌ها که لازم می‌داند کار کند.

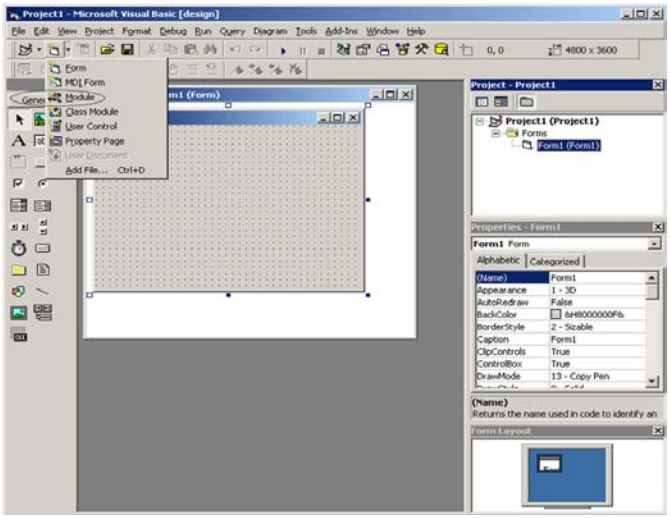
برای ایجاد یک فرم MDI می‌توانید یکی از روش های زیر را مورد استفاده قرار دهید:

۱- در پنجره پروژه کلیک راست کنید و گزینه Add را برگزینید، سپس گزینه MDI Form را انتخاب کنید (شکل ۴۴-۱۲).



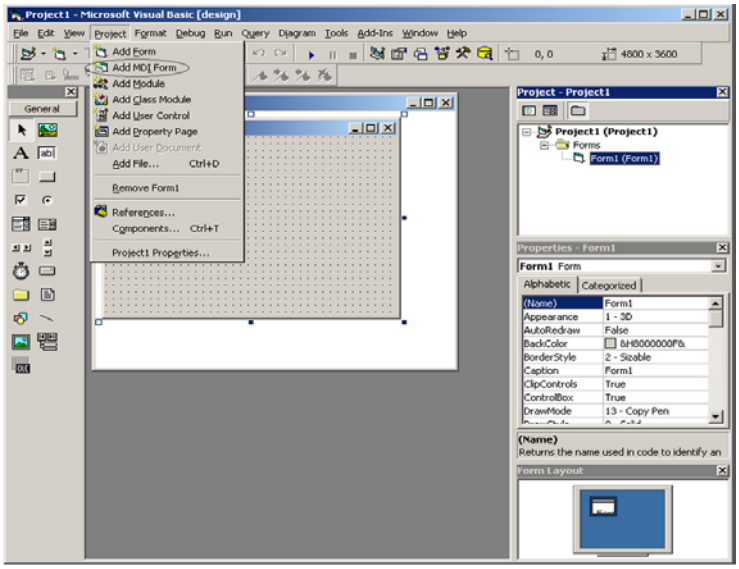

**شکل ۴۴-۱۲**

۲- در پنجره ویژوال بیسیک و در نوار ابزار روی علامت مثلث کنار دکمه Add در بخش ProjectIcons کلیک کنید، سپس گزینه MDI Form را برگزینید.



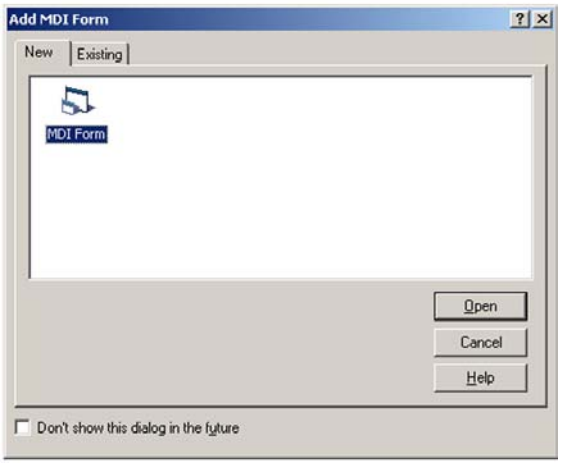

**شکل ۴۵-۱۲**

۳- در پنجره ویژوال بیسیک، روی منوی Project کلیک کنید و گزینه Add MDI Form را برگزینید. (شکل ۴۶-۱۲).



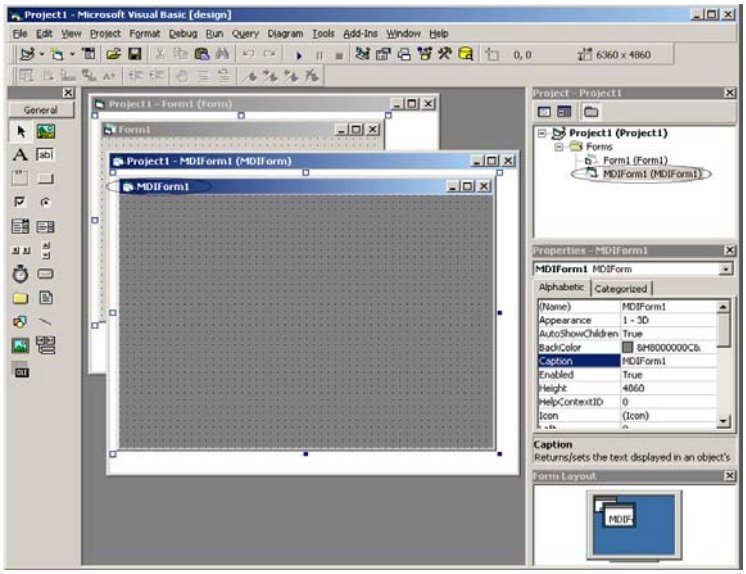

**شکل ۴۶-۱۲**

پس از اجرای یکی از روش‌های فوق، کادرمحاوره‌ای مطابق شکل۴۷-۱۲ نمایش داده می‌شود. در این مرحله آیکن MDI Form را انتخاب کرده و روی دکمه فرمان Open کلیک کنید.



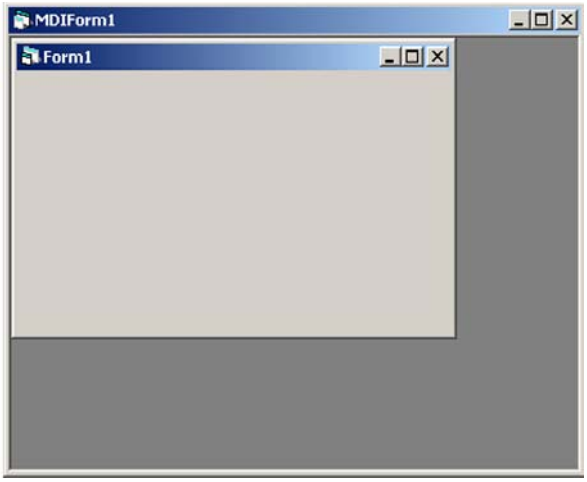

**شکل ۴۷-۱۲**

پس از انجام عملیات فوق، پنجره برنامه ویژوال بیسیک مطابق شکل ۴۸-۱۲ مشاهده می‌شود که یک فرم از نوع MDI و یک فرم SDI که از قبل در پروژه وجود داشته است به چشم می‌خورند، رنگ زمینه فرم MDI تیره‌تر از فرم‌های SDI است.




**شکل ۴۸-۱۲**

برای آن که بین یک فرم SDI و MDI ارتباط برقرار کرده و فرم SDI را به MDI وابسته کنید ابتدا فرم SDI را برگزینید، سپس خاصیت MDIChild را در پنجره خواص پیدا کرده و مقدار آن را روی True تنظیم کنید، در ادامه در منوی Project گزینه Project Properties را برگزینید و در کادر محاوره خواص پروژه در بخش Startup Object روی دکمه کلیک کنید و گزینه MDIForm را برگزینید. در پایان روی دکمه فرمان OK کلیک کنید. اکنون ارتباط برقرار شده است و فرم SDI به فرم MDI وابسته است. برای درک بهتر تفاوت این دو نوع فرم در رویداد Load فرم MDI فرمان Form1.Show را بنویسید و سپس پروژه را اجرا کنید. برنامه اجرا شده و رابط گرافیکی را مطابق شکل ۴۹-۱۲ مشاهده خواهید کرد. فرم SDI را جابه‌جا کنید و سعی کنید آن را از داخل فرم MDI خارج کنید. ابتدا پنجره فرم SDI و سپس پنجره فرم MDI را ببندید. برنامه را مجدداً اجرا کنید و این بار فرم MDI را جابه‌جا کنید و سپس آن را ببندید. چه تفاوتی بین این حالت و حالت قبل مشاهده می‌کنید؟




**شکل ۴۹-۱۲**

**نکته**

قبل از اجرای پروژه، فرم MDI را به عنوان فرم اول پروژه در کادرمحاوره خواص پروژه انتخاب کنید.

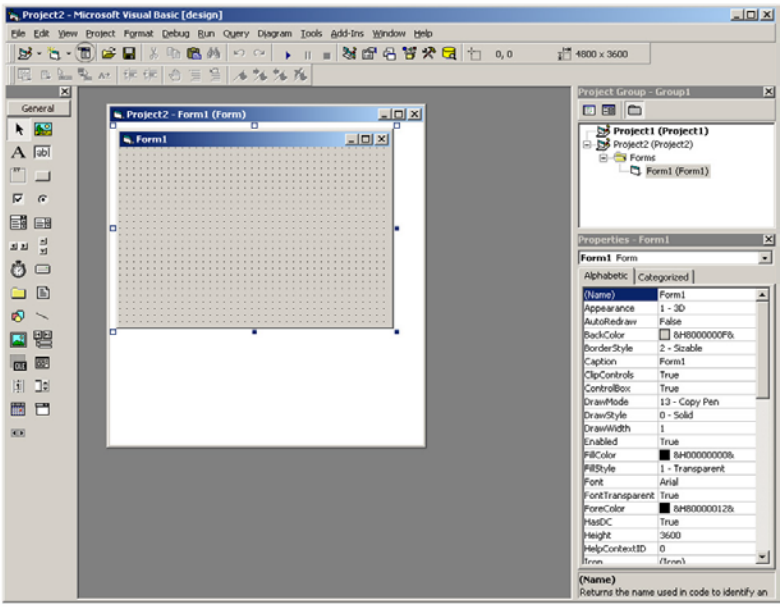
## ۱۸-۱۲ نحوه ایجاد انواع منو در ویژوال بیسیک

یکی از عناصر اصلی در برنامه‌ها نوارهای منو و گزینه‌های موجود در آن است، نوارهای منو و گزینه‌های موجود در آن دسترسی کاربر به امکانات برنامه را آسان‌تر کرده و شکل ظاهری مناسبی را برای رابط گرافیکی ایجاد می‌کنند. در ویژوال بیسیک نیز مانند سایر زبان‌های برنامه‌نویسی امکانات لازم جهت ایجاد انواع منوها فراهم شده است. در این بخش شما را همراه با انجام یک مثال با نحوه ایجاد انواع منو آشنا می‌کنیم.

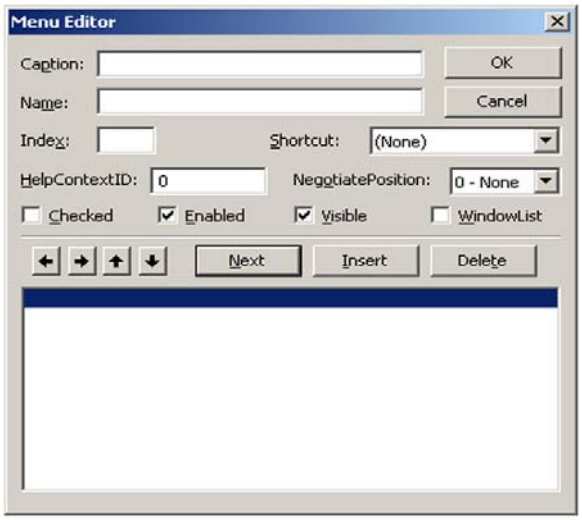
**مثال :** می‌خواهیم یک برنامه با یک فرم به همراه نوار منو طراحی کنیم تا کاربر بتواند هر تصویر دلخواهی را روی فرم نمایش دهد و به‌علاوه بتواند اندازه تصویر را با میل خود تنظیم کند. برای انجام این کار عملیات زیر را به ترتیب انجام دهید:

۱- برنامه ویژوال بیسیک را اجرا کنید و یک پروژه از نوع Standard EXE به همراه یک فرم ایجاد کنید و فرم Form1 را برگزینید.

۲- برای ایجاد منو در روی فرم، در نوار ابزار پنجره ویژوال بیسیک کلیک کنید (شکل ۵۰-۱۲)، تا کادر محاوره Menu Editor مطابق شکل ۵۱-۱۲ نمایش داده شود.




**شکل ۵۰-۱۲**



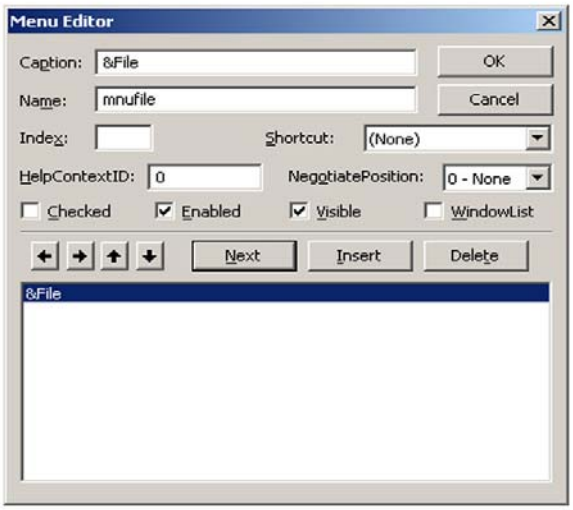

**شکل ۵۱-۱۲**

در این پنجره بخش‌های مختلفی را مشاهده می‌کنید که به توضیح آن‌ها می‌پردازیم.

عنوان منو را در کادر متن Caption تایپ کنید و نام منو را در کادر متن Name بنویسید نام منو شبیه به نام کنترل‌ها و فرم‌هاست و به‌وسیله آن می‌توان به گزینه‌های یک منو با استفاده از

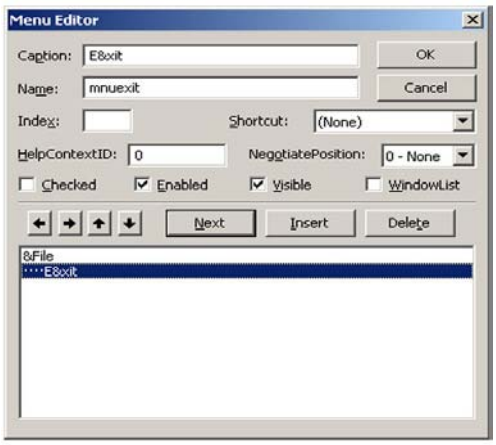
کدنویسی دسترسی پیدا کرد. اگر کادر علامت Checked را برای یک منو فعال کنید در کنار عنوان گزینه‌های منو علامت «✓» را مشاهده خواهید کرد. کادر علامت Enabled، منو و گزینه‌های آن را فعال و غیرفعال می‌کند و کادر علامت Visible، سبب نمایش یا عدم نمایش منو و گزینه‌های آن خواهد شد. به‌علاوه در بخش کادر لیست Shortcut می‌توانید برای هر یک از گزینه‌های منو یک کلید ترکیبی تعیین کنید.

۳- در کادر متن Caption، عبارت &File و در کادر متن Name عبارت mnufile را تایپ کنید، کاراکتر & سبب ایجاد یک کلید دسترسی سریع برای منوی File می‌شود (شکل ۵۲-۱۲).



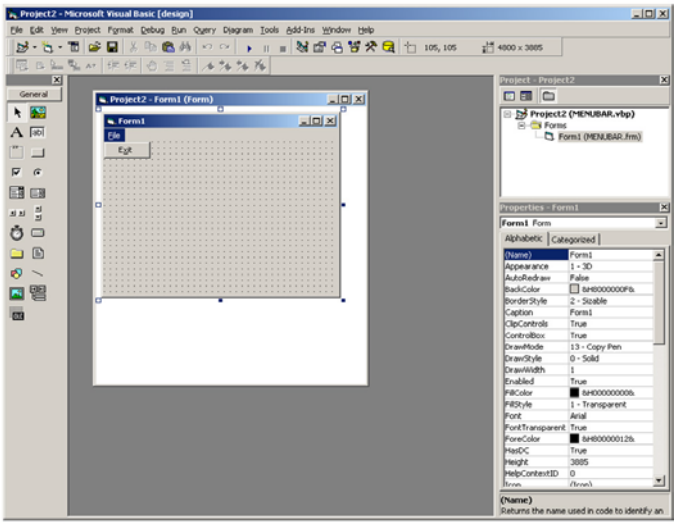

**شکل ۵۲-۱۲**

۴- برای آن که یک گزینه جدید با عنوان Exit در منوی File ایجاد کنید ابتدا روی دکمه Next کلیک کنید و سپس روی دکمه ➔ کلیک کنید. سپس در بخش عنوان منو عبارت E&xit و در کادر متن نام عبارت mnuexit را تایپ کنید (شکل ۵۳-۱۲).




**شکل ۵۳-۱۲**

۵- در کادر محاوره Menu Editor روی دکمه OK کلیک کنید، همان‌طور که مشاهده می‌کنید منوی File در روی فرم به چشم می‌خورد و اگر روی آن کلیک کنید، گزینه Exit نیزمشاهده می‌شود (شکل ۵۴-۱۲).



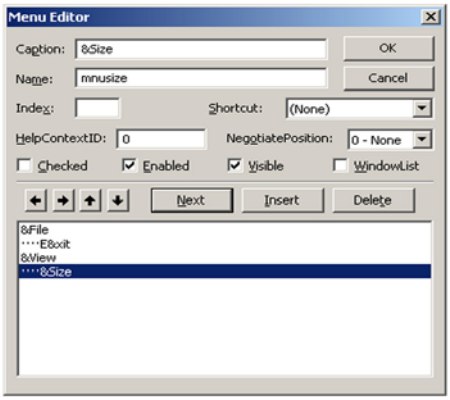

**شکل ۵۴-۱۲**

۶- مجدداً به پنجره Menu Editor باز گردید و در کادر لیست اسامی منوها گزینه E&xit... را انتخاب کنید، سپس روی دکمه Next کلیک کنید.

۷- می‌خواهیم یک منوی دیگر در کنار منوی فایل با عنوان View ایجاد کنیم، بنابراین روی دکمه

کلیک کنید تا نقاط مربوطه حذف شوند، سپس عبارت‌های &View و mnuview را به‌ترتیب برای عنوان و نام منو تایپ کنید.

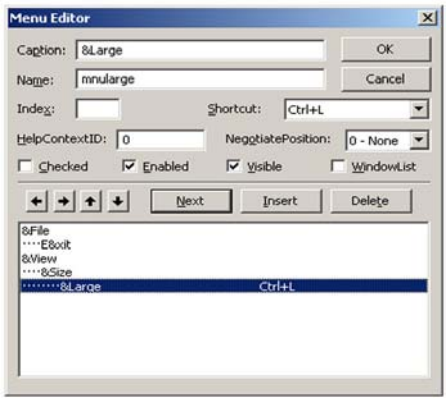
۸- مجدداً روی دکمه Next و سپس روی دکمه کلیک کنید و یک گزینه در منوی View با عنوان &Size و نام mnusize ایجاد کنید (شکل ۵۵-۱۲).




**شکل ۵۵-۱۲**

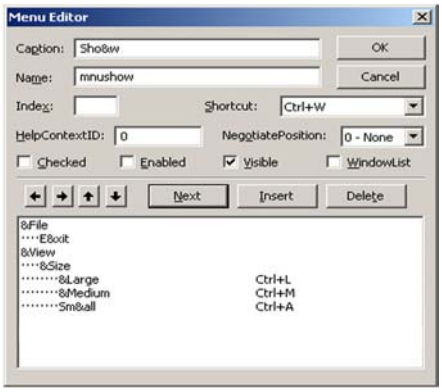
۹- اکنون می‌خواهیم منوی Size را به‌صورت یک زیرمنو در آورده و گزینه‌هایی را در داخل آن قرار دهیم که از آن‌ها برای تعیین اندازه تصویر نمایشی استفاده کنیم.

بنابراین یکبار دیگر روی دکمه Next و سپس روی دکمه کلیک کنید. گزینه‌ای با عنوان &Large و نام mnularge ایجاد کرده (شکل ۵۶-۱۲) و از کادر لیست Shortcut، کلید ترکیبی Ctrl+L را برای آن انتخاب کنید.




**شکل ۵۶-۱۲**

**۱۰**- به همین صورت دو گزینه دیگر با عناوین &Medium و Sm&all و نام‌های mnumedium و mnusmall با کلیدهای ترکیبی Ctrl+M و Ctrl+A ایجاد کنید (شکل ۵۷-۱۲). سپس روی گزینه Medium کلیک کنید و کادر علامت Checked را برای آن فعال کنید.




**شکل ۵۷-۱۲**

**نکته**

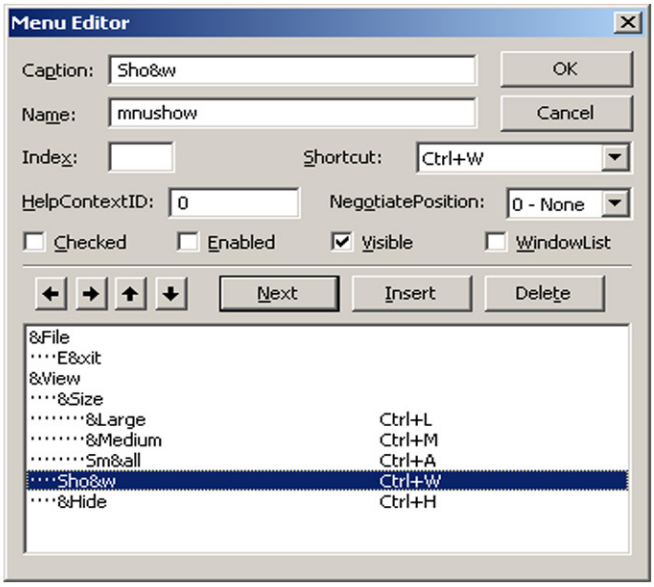
در صورت نیاز می‌توانید با استفاده از دکمه‌های ↓ و ↑ در پنجره Menu Editor گزینه‌ها را در منوهای ایجاد شده جابه‌جا کنید.

**نکته**

برای حذف یک منو از دکمه Delete در پنجره Menu Editor استفاده کنید.

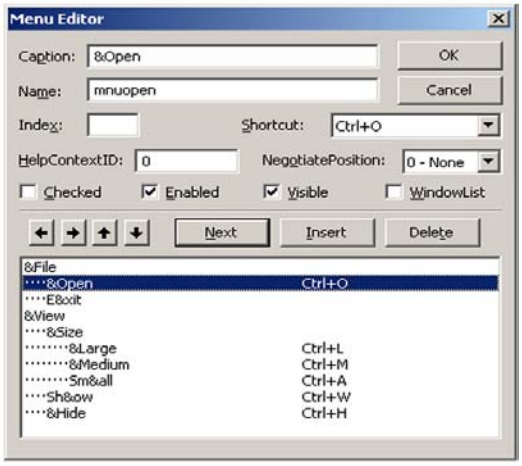
**۱۱**- روی دکمه OK کلیک کنید و در پنجره طراحی فرم روی منوی View کلیک کنید، همان‌طور که مشاهده خواهید کرد در این منو، یک زیرمنو به همراه سه گزینه ایجاد شده است و در روبه‌روی گزینه‌های آن کلیدهای ترکیبی هر یک را مشاهده می‌کنید، به‌علاوه علامت «✓» در کنار گزینه Medium نیز نمایش داده شده است.

**۱۲**- به همین صورت دو گزینه دیگر در منوی View با عناوین Sho&w و &Hide با نام‌های mnushow و mnuhide ایجاد کنید که دارای کلیدهای ترکیبی Ctrl+W و Ctrl+H باشند و کادر علامت Enabled را برای گزینه Show غیرفعال کنید (شکل ۵۸-۱۲).




**شکل ۵۸-۱۲**

**۱۳**-اکنون یک گزینه با عنوان Open در منوی فایل و در بالای گزینه Exit ایجاد کنید، بنابراین به پنجره Menu Editor بروید و در کادر لیست اسامی منوها، گزینه Exit را انتخاب کنید. در ادامه روی دکمه Insert کلیک کنید تا یک گزینه خالی ایجاد شود، سپس در کادر متن عنوان و نام منو عبارات &Open و mnuopen را تایپ کنید (شکل ۵۹-۱۲).




**شکل ۵۹-۱۲**

**۱۴**- در این مرحله یک کنترل CommonDialog با نام opendialog و یک کنترل Image با نام showpicture ایجاد کنید.

**۱۵**-اکنون می‌خواهیم کدهایی بنویسیم که در صورت استفاده کاربر از گزینه‌های موجود در منوهای File و View، رویدادهای مورد نظر رخ دهد. بنابراین پنجره کدنویسی را فعال کنید و رویداد کلیک گزینه Open یعنی mnuopen_Click را برگزینید. برای این که کاربر بتواند هر فایل دلخواه خود را جهت نمایش انتخاب کند از کنترل CommonDialog استفاده کنید و دستورات زیر را در رویداد mnuopen_Click تایپ کنید:

```
opendialog. ShowOpen
showpicture. Picture = LoadPicture (opendialog. FileName)
```

این دستورات سبب می‌شود تا با انتخاب گزینه Open از منوی File کادر محاوره Open باز شود و با انتخاب یک فایل گرافیکی از این کادر محاوره تصویر مربوطه در کنترل Image نمایش داده شود.

**۱۶**- در این مرحله برای آن که تصویر با توجه به انتخاب کاربر با یکی از اندازه‌های Large ، Medium یا Small نمایش داده شود، رویدادهای این سه گزینه را به شکل زیر تنظیم کنید.

```
Private Sub mnularge_Click()
    mnularge.Checked = True
    mnumedium.Checked = False
    mnusmall.Checked = False
    showpicture.Height = 5500
    showpicture.Width = 7500
End Sub


Private Sub mnumedium_Click()
    mnularge.Checked = False
    mnumedium.Checked = True
    mnusmall.Checked = False
    showpicture.Height = 5000
    showpicture.Width = 6000
End Sub


Private Sub mnusmall_Click()
    mnularge.Checked = False
    mnumedium.Checked = False
    mnusmall.Checked = True
    showpicture.Height = 4000
    showpicture.Width = 5000
End Sub
```

در این رویدادها با استفاده از خاصیت Checked گزینه‌ها و در صورت انتخاب یک گزینه توسط کاربر علامت «✓» از گزینه‌ای که قبلاً انتخاب شده است، برداشته شده و در کنار گزینه انتخاب شده نمایش داده می‌شود. به‌علاوه اندازه کنترل Image و در واقع اندازه تصویر تنظیم می‌شود.

**۱۷**-اکنون رویدادهای کلیک گزینه‌های Show و Hide را به شکل زیر تنظیم کنید:

```
Private  Sub  mnushow_Click( )
        mnushow.Enabled  =  False
        mnuhide.Enabled  =  True
        showpicture.Visible  =  True
End  Sub

Private  Sub  mnuhide_Click( )
        mnuhide.Enabled  =  False
        mnushow.Enabled  =  True
        showpicture.Visible  =  False
End  Sub
```

همان‌طور که در دستورات فوق مشاهده کردید با انتخاب گزینه Show تصویر نمایش داده می‌شود و گزینه Hide فعال می‌گردد، اما خود گزینه Show غیرفعال می‌شود و به همین صورت با انتخاب گزینه Hide تصویر مخفی شده و گزینه Show فعال می‌گردد و خود گزینه Hide غیرفعال می‌شود.

**۱۸**-در این مرحله رویدادهای مربوط به فرم و گزینه Exit را به این صورت تنظیم کنید:

```
Private  Sub  Form_Load(  )
        frmshowpicture.Height  =  9000
        frmshowpicture.Width  =  9500
        frmshowpicture.Top  =  100
        frmshowpicture.Left  =  100
```

```
        showpicture.Stretch  =  True
        showpicture.Height  =  5000
        showpicture.Width  =  6000
End  Sub

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

**۱۹**- برنامه را اجرا کنید و با استفاده از گزینه Open در منوی فایل یک فایل BMP یا JPG را انتخاب کنید، سپس گزینه‌های منوی Size و گزینه‌های Hide و Show را به ترتیب برگزینید و نتیجه را بررسی کنید.

**۲۰**- در این مرحله شما را با نحوه ایجاد آخرین نوع از انواع منوها آشنا خواهیم کرد. در واقع می‌خواهیم در صورتی که کاربر در روی فرم کلیک راست انجام دهد، گزینه‌های منوی View در اختیار آن قرار گیرد. به این منظور رویدادهای Form_MouseDown و showpicture_MouseDown را انتخاب کرده و به شکل زیر تنظیم کنید.

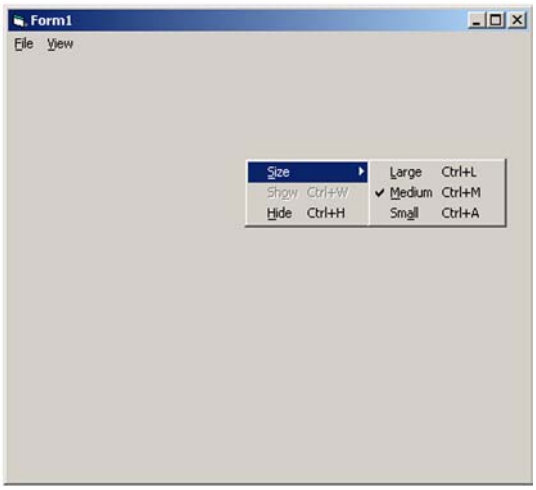
```
Private Sub Form_MouseDown(Button As Integer,Shift As _
Integer,  X  As  Single,  Y  As  Single)
        If  Button  =  2  Then  PopupMenu  mnuview
End  Sub

Private  Sub  showpicture_MouseDown(Button As Integer, _
Shift  As  Integer,  X  As  Single,  Y  As  Single)
        If  Button  =  2  Then  PopupMenu  mnuview
End  Sub
```

در واقع با انجام کلیک راست توسط کاربر، دستور PopupMenu، گزینه‌های موجود در منوی View را در اختیار کاربر قرار می‌دهد که عملکرد آن‌ها دقیقاً مانند انتخاب آن‌ها از نوار منو است. با استفاده از دستور PopupMenu می‌توانید هر منویی را با استفاده از کلیک راست فعال کنید.

**۲۱**- برنامه را اجرا کنید و پس از انتخاب یک تصویر و نمایش آن روی فرم یا تصویر کلیک راست کرده

و عملکرد گزینه‌ها را بررسی کنید، همان‌طور که مشاهده خواهید کرد منوی View مطابق شکل ۶۰-۱۲ نمایش داده می‌شود.

**۲۲**-از برنامه خارج شوید و فرم و پروژه خود را با اسامی menubar.frm و menubar.vbp ذخیره کنید.

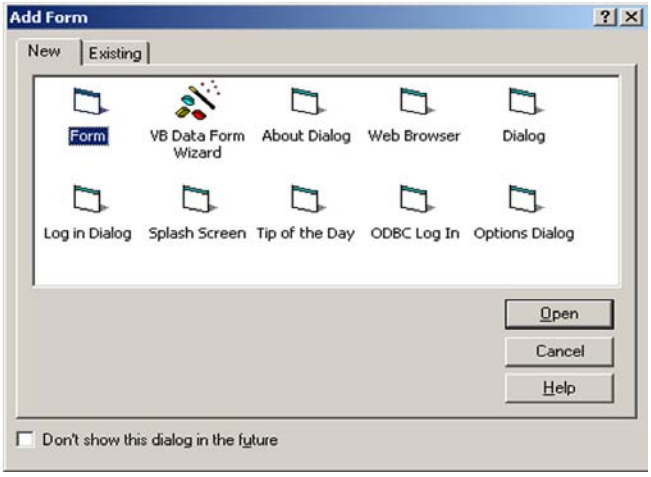



**شکل ۶۰-۱۲**

## ۱۹-۱۲ فرم‌های آماده (Template Forms)

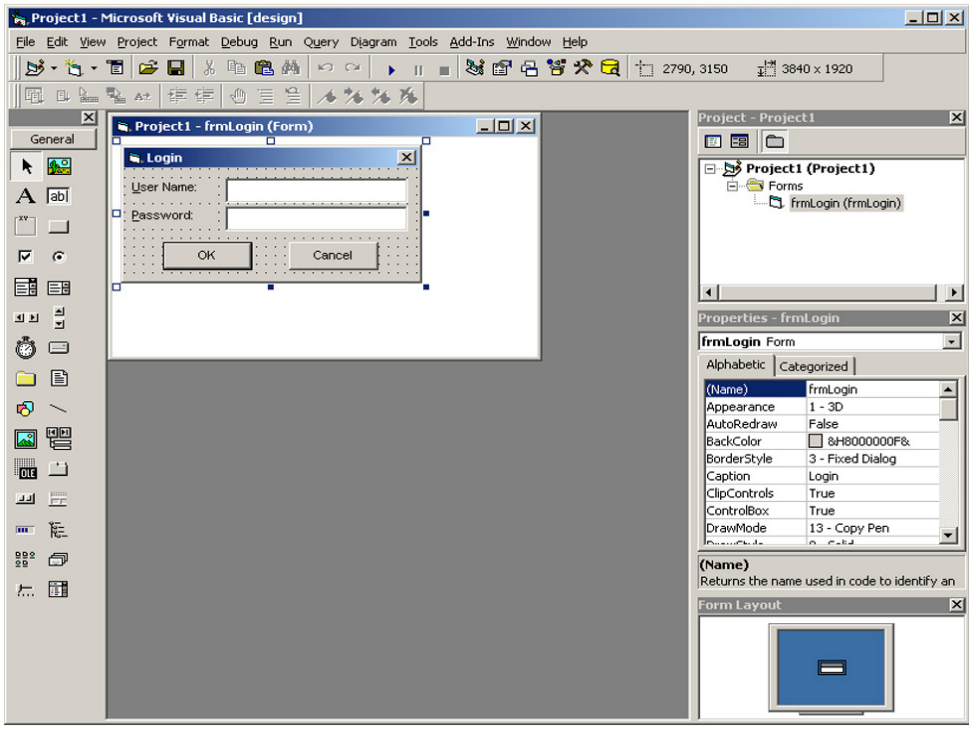
در ویژوال بیسیک فرم‌های آماده متعددی که از قبل طراحی و ساخته شده‌اند، وجود دارد. این فرم‌ها در زمان نصب بسته نرم‌افزاری ویژوال بیسیک در روی سیستم شما قرار داده می‌شوند و شما می‌توانید از آن‌ها در برنامه‌های خود استفاده‌کنید یک نمونه از این‌گونه فرم‌ها را تا کنون به‌صورت مکرر از طریق پنجره Add Form مورد استفاده قرار داده‌اید. انواع دیگری نیز از فرم‌های آماده وجود دارند مانند انواع کادر محاوره و خوش‌آمدگویی و ورود به سیستم و نظایر آن‌ها.

در صورتی که بخواهید از این فرم‌ها به‌صورت آماده در پروژه‌های خود استفاده کنید (مطابق شکل ۶۱-۱۲) در کادر محاوره Add Form، فرم مورد نظر خود را برگزینید و روی دکمه فرمان Open کلیک کنید.




**شکل ۶۱-۱۲**

به عنوان مثال در شکل ۶۲-۱۲، یک فرم از پیش آماده با عنوان Log in Dialog را به پروژه خود به همین صورت اضافه کرده‌ایم.




**شکل ۶۲-۱۲**

## خلاصه مطالب

- برای ایجاد لیستی از گزینه‌های مختلف از کنترل ListBox , ComboBox استفاده می‌شود.
- برای ایجاد کادرهای محاوره بازکردن فایل‌ها، ذخیره‌سازی فایل‌ها، رنگ‌ها، فونت، چاپگر و راهنما از کنترل CommonDialog استفاده می‌شود.
- از کنترل DriveListBox برای نمایش و استفاده از درایوهای موجود در یک سیستم استفاده می‌شود.
- از کنترل DirListBox برای نمایش و انتخاب پوشه‌ها استفاده می‌شود.
- از کنترل FileListBox برای نمایش و انتخاب فایل استفاده می‌شود.
- جهت کار با تقویم و انتخاب تاریخ مورد نظر از کنترل MonthView استفاده می‌شود.
- از کنترل DTPicker جهت کار روی داده‌های تاریخ و ساعت استفاده می‌شود.
- برای استفاده از قابلیت‌های نوار پیمایش از کنترل‌های FlatScrollBar ، HScrollBar و VscrollBar استفاده می‌شود.
- برای ایجاد لیستی از تصاویر جهت استفاده در کنترل‌هایی نظیر ToolBox، ImageCombo و CoolBar از کنترل ImagList استفاده می‌شود.
- برای ایجاد لیستی از گزینه‌های مختلف همراه با تصاویر مربوطه از کنترل ImageCombo استفاده می‌شود.
- از کنترل MaskedEdit برای دریافت داده‌ها با قالب بندی معینی استفاده می‌شود.
- برای دریافت و ویرایش داده‌های متنی از کنترل دیگری به‌نام RichTextBox استفاده می‌شود.
- از کنترل Slider برای تنظیم یک مقدار معین استفاده می‌شود.
- به‌وسیله کنترل UpDown می‌توان مقادیر مربوط به خواص کنترل‌های دیگر را تنظیم کرد یا از آن برای تنظیم یک مقدار استفاده کرد.
- در رابط‌های گرافیکی از نوع SDI هر فرم می‌تواند به‌صورت مستقل عمل کند.
- در رابط‌های گرافیکی از نوع MDI، فرم‌های فرزند به فرم مادر وابسته هستند.
- در ویژوال بیسیک تعداد متعددی از فرم‌های از پیش آماده در اختیار برنامه‌نویس قرار می‌گیرد.

# آزمون پایانی

**۱- کدام کنترل جهت تهیه لیستی از گزینه‌های دلخواه با تصاویر همراه مناسب است؟**

۱- ImageList ۲- ComboList

۳- ListBox ۴- ImageCombo

**۲- کدام کنترل برای ایجاد انواع کادرهای محاوره مناسب است؟**

۱- ListBox ۲- ComboBox

۳- CommonDialog ۴- DTPicker

**۳- کدام کنترل جهت دریافت داده‌ها با قالب‌بندی معین مناسب است؟**

۱- Text Box ۲- List Box

۳- Masked Edit ۴- Slider

**۴- در کدام نوع از رابط‌های گرافیکی یک فرم به فرم دیگر وابستگی دارد؟**

۱- MDI ۲- SDI

۳- Template ۴- گزینه‌های ۱و۲ صحیح هستند.

**۵- کدام رویداد در کنترل‌های نوار پیمایش برای ایجاد تغییرات با اندازه کوچک مفید است؟**

۱- Change ۲- Scroll ۳- LostFocus ۴- GotFocus

## دستور کار آزمایشگاه

**۱**- یک پروژه طراحی کنید که کاربر بتواند با استفاده از چند کنترل لیست، رنگ قلم، رنگ زمینه و اندازه قلم را در یک عبارت متنی تغییر دهد.

**۲**- پروژه‌ای طراحی کنید که بتوان با استفاده از آن تصاویر مورد نظر را از یک کنترل ImageCombo انتخاب کرد و تصویر مورد نظر را در یک کنترل PictureBox نمایش داد.

**۳**- پروژه‌ای طراحی کنید که به‌وسیله آن بتوان یک فایل متنی را در هر مسیر دلخواه باز، ویرایش و ذخیره کرد.

**۴**- پروژه‌ای طراحی کنید که به‌وسیله کنترل UpDown بتوان مدت زمان را برای فعال شدن یک کنترل Timer را تعیین کرد.

## پاسخ پیش آزمون

| | | | |
|---|---|---|---|
| **۱-** ۲ | **۲-** ۳ | **۳-** ۳ | **۴-** ۱ |
| **۵-** ۱ | **۶-** ۴ | **۷-** ۲ | **۸-** ۲ |
| **۹-** ۳ | **۱۰-** ۴ | | |

## پاسخ آزمون پایانی

| | | | |
|---|---|---|---|
| **۱-** ۴ | **۲-** ۳ | **۳-** ۳ | **۴-** ۱ |
| **۵-** ۲ | | | |