فصل ۳ : استفاده از توابع درونی VBA

توابع در VBA از نظر ساختار شبیه توابع دیگر زبانهای برنامه نویسی هستند. اغلب توابع VBA ورودی خاصی دارند (نوع داده ، تعداد ورودی و غیره) و همه آنها حداقل یک خروجی دارند. VBA توابع عدیده ای را دارد که بـرای برنامـه نویسـی شما مناسب می باشند. در این فصل ، من توابع عمومی را شرح خواهم داد که برای نوع خاصی از داده ها طراحی شـده انـد. این فصل همه توابع درونی VBA را در برنامه ندارد اما برای این کار با برخی اشیا طراحی شده است.

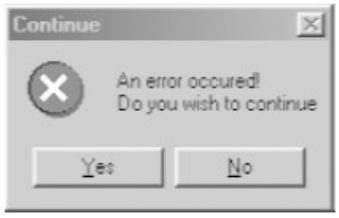
شروع پایه ای I / O با VBA

شما اغلب نیاز خواهید داشت که پیغامی را به کاربران بدهید تا به آنها کمک کنید یا اینکه نحوه کار را به آنها آموزش دهیـد. شاید بخواهید کاربران چیزی را وارد کنند. ساده ترین راه بـرای ایجـاد یـک پیغـام خروجـی اسـتفاده از تـابع MsgBox() می باشد.تابع MsgBox() برای نمایش یک پنجره حاوی اطلاعات به کاربر ، بکار می رود. کاربر قبل از ادامه کـار بایـد بـا کلیک روی دکمه موجود پنجره ، به آن پاسخ دهد. می توانید از تابع InputBox() برای دریفت ورودی کاربر استفاده کنید. این تابع یک پنجره را به همراه یک کادر دریافت داده نشان می دهد تا کاربر توسط کیبورد اطلاعات را وارد نماید.

توجه نام تابع با دو پرانتز خالی پس از آن بیانگر این است که تابع به ورودی نیاز دارد(اغلب یک یا چنـد متغیـر). شما چند مثال را در این کتاب خواهید دید که نیاز به ورودی ندارند. این توابع بدون پرانتز نوشته می شوند.

تابع MsgBox()

شما از تابع MsgBox() استفاده می کنید تا اطلاعات خاصی را به کاربر ارائه نموده و او را مجبور به تصـمیم گیـری کنیـد. مثلا روی OK ، Yes ، Cancel کلیک کند. این تابع دارای یک خروجی بصورت یک پنجره کوچک است هماننـد شـکل ۱,۳ . بر اساس تصمیم کاربر برنامه اجرا و ادامه خواهد یافت.



شکل ۳ . ۱ : یک کادر پیغام

ترکیب و ساختار تابع MsgBox() بصورت زیر است:

```
MsgBox(prompt[, buttons] [, title] [, helpfile, context])
```

پارامترهای تابع (که آرگومان نامیده می شـوند) شـامل Prompt ، Buttons ، title و helpfile و Context مـی باشـد. شما باید یک متن را در قسمت prompt قرار دهید تا هنگام نمایش پنجره به کاربر نشان داده شود. بقیه آرگومانها اختیـاری هستند. البته معمولا قرار دادن مقداری برای آرگومانهای دیگر خالی از لطف نخواهد بود. شما همچنین می توانید با استفاده از

helpfile و context به راهنمایی کاربر بپردازید. مقدار آرگومان button برای VBA شکل و نوع و تعـداد دکمـه هـا را مشخص می کند. دکمه های تابع می تواند یک یا چندینثابت VBA باشد که در جدول ۱,۳ لیست شده اند.

جدول ۱.۳ : ثوابت موجود برای آرگومان button در تابع MsgBox()

ثابت	مقدار	توضیحات
VbOKOnly	۰	فقط دکمه OK نمایش داده می شود.
VbOKCancel	۱	دکمه های OK و Cancel نمایش داده می شود.
VbAbortRetryIgnore	۲	دکمه های Abort , Retry , Ignore نمایش داده می شود
VbYesNoCancel	۳	دکمه های Yes , No , Cancel نمایش داده می شود.
VbYesNo	٤	دکمه های Yes , No نمایش داده می شود.
VbRetryCancel	٥	دکمه های Retry , Cancel نمایش داده می شود.
VbCritical	١٦	آیکون Critical Message را نمایش می دهد.
VbQuestion	٣٢	آیکون Warning Query را نمایش می دهد.
VbExclamation	٤٨	آیکون Warning Message را نمایش می دهد.
VbInformation	٦٤	آیکون Information Message را نمایش می دهد.
vbDefaultButton١	۰	آولین دکمه پیش فرض است.
vbDefaultButton٢	٢٥٦	دومین دکمه پیش فرض است.
vbDefaultButton٣	٥١٢	سومین دکمه پیش فرض است.
vbDefaultButton٤	٧٦٨	چهارمین دکمه پیش فرض است.
VbApplicationModal	۰	کاربر باید یه پیغام کادر قبل از ادامه کارش پاسخ دهد.
VbSystemModal	٤٠٩٦	همه برنامه ها تا زمانی که کاربر به پیغام کادر پاسخ دهد معلق خواهند ماند.
VbMsgBoxHelpButton	١٦٣٨٤	دکمه Help را به کادر پیغام اضاف می کند.
VbMsgBoxSetForeground	٦٥٥٣٦	تعیین اینکه کادر پیغام بر روی دیگر برنامه قرار گیرد.
VbMsgBoxRight	٥٢٤٢٨٨	متن به صورت راست چین مرتب شود.
VbMsgBoxRtlReading	١٠٤٨٥٧٦	تعیین اینکه متن باید به صورت راست به چپ در سیستم های عبری و عربی (و البته فارسی) قرار گیرد.

در مثال زیر ، برنامه به کاربر اعلام می کند که اشکالی در اجرای برنامه رخ داده است و ورودی کاربر تعیین کننده نحوه اجرا خواهد بود:

```
Dim msgString As String
Dim retValue As Integer
msgString = "An error occurred!" & vbCrLf & "Do you wish to continue"
retValue = MsgBox(msgString, vbCritical + vbYesNo, "Continue")
```

پنجره پیغام ، پیغام و دکمه های Yes و No را نشان خواهد داد. مقدار برگشت الزامی است چرا که برنامه باید بداند کاربر کدام را انتخاب می کند (Yes یا No). تابع MsgBox() بر اساس دکمه ای که کاربر انتخاب می کند ، مقدار متفاوتی را باز می گرداند. مقادیر برگشتی به صورت ثوابت VBA در نظر گرفته می شوند. (که در جدول ۲,۳ لیست شده اند.) و تنها در مواقعی که پنجره حاوی بیش از یک دکمه باشد مفید خواهد بود.

جدول ۲.۳ : مقادیر برگشتی MsgBox()

ثابت	مقدار	توضیحات
VbOK	۱	OK
VbCancel	۲	Cancel
VbAbort	۳	Abort
VbRetry	٤	Retry
VbIgnore	٥	Ignore
VbYes	٦	Yes
VbNo	۷	No

اگر کاربر در پنجره پیغام روی Yes کلیک کند ، مقدار ٦ به متغیر retValue اختصاص می یابد. بعد از آن برنامه می تواند دوره ای از کدها را با توجه به عدد ٦ اجرا کند. (فصل ٥ را برای جزئیات بیشتر ببینید.)

اخطار

از بکار گیری بیش از اندازه پنجره های پیغام اجتناب کنید. این برای کاربران آزار دهنده خواهد بود. آنها را برای موارد مناسب نگهدارید. هرگز از آنها برای ورود یا خروج داده هایی که به اجرای برنامه کمکی نمی کنند استفاده نکنید.

تابع InputBox()

از این تابع برای مواقعی استفاده کنید که می خواهید کاربر مقداری را قبل از اجرای برنامه وارد نماید. تابع InputBox() پنجره ای مشابه شکل ۲,۳ ایجاد می کند.

شکل ۳ . ۲ :یک کادر ورودی

همانند تابع MsgBox() در میان آرگومانها ، تنها Prompt ضروری است.

```
InputBox(prompt[, title] [, default] [, xpos] [, ypos] [, helpfile, context])
```

اما درج عنوان و مقدار پیش فرض ، کاربر را در اینکه چه چیزی باید وارد کند، یاری خواهد کرد. همچنین شاید بخواهید مکان نمایش پنجره را در صفحه نمایش با xpos و ypos تعیین کنید و نیز ارجاعی به فایل help داشته باشید.

```
Dim retValue As String
retValue = InputBox("Enter your name.", "Name", "First, Last")
```

در کد اخیر ، من متنی را به prompt وارد کرده ام و نیز به عنوان و مقدار پیش فرض نیز مقادیری اختصاص داده ام که نتیجه در شکل ۲,۳ دیده می شود. من می توانستم از متغیرهای متنی برای درج در prompt ، title و Default استفاده کنم. اغلب اوقات ، مقادیر پیش فرض ، کاربر را در نحوه درج نام و ترتیب درج آنها یاری می کند.

اخطار اگر راهنمایی را آماده کنید، من به طور قوی به کاربردن تابع InputBox() را توصیه می کنم.

تابع InputBox() یک رشته متنی را بر می گرداند که کاربر درون کادر مخصوص متن درج کرده است. اگر کادر متن خالی باشد یا کاربر روی Cancel کلیک کند، تابع مقدار متن صفر کاراکتر را برمی گرداند یعنی (" ").

تغییر نوع متغیرها با تابع Type – Conversion

اگر بخواهید یک متغیر را از یک نوع به نوعی دیگر تبدیل کنید ، می توانید از یکی از چندین تابع تبدیلاتی استفاده نمایید. (جدول ۳,۳ را برای مشاهده لیست این توابع ببینید.)

جدول ۳ . ۳ : توابع تغییر نوع

تابع	نوع برگشت	مثال
CBool()	Boolean	CBool(۱) بر می گرداند True
CByte()	Byte	CByte(۵,۶) بر می گرداند ۶
CCur()	Currency	CCur(۱۰,۳۲٤٥٦٧) بر می گرداند۱۰,۳۲٤٦
CDate()	Date	CDate(۳٦٦,٥) بر می گرداند.۱۲/۳۱/۱۹۰۰ ۱۲:۰۰:۰۰ PM

تابع	نوع برگشت	مثال
CDbl()	Double	CDbl(٩٩,٢ ^ ٢٨) بر می گرداند ٧,٩٨٥٩٥٤٢٩٤٨٩٢١٢E+٥٥
CDec()	Decimal	CDec(٢ ^ ٣٢) بر می گرداند ٤٢٩٤٩٦٧٢٩٦
CInt()	Integer	CInt(٢٥ * ٥,٦٢٣) بر می گرداند ١٤١
CLng()	Long	CLng(٢ ^ ٣٠) بر می گرداند ١٠٧٣٧٤١٨٢٤
CSng()	Single	CSng(٢٥ * ٥,٥) بر می گرداند ١٣٧,٥
CStr()	String	CStr(١٠٠١٩) بر می گرداند "١٠٠١٩"
CVar()	Variant	

شما همچنین می توانید از این توابع برای حصول اطمینان استفاده کنید ، مثلا اگر بخواهید که حاصل ضرب دو عدد یک عدد صحیح باشد از CInt() استفاده کنید. کد زیر باعث می شود که حاصل ضرب دو عدد ، یک عدد صحیح باشد:

```
val١ = ٣,١٤
val٢ = ٦,٦٣
val٣ = CInt(val١ * val٢)
```

تابع CInt() در این مثال نیاز نیست. چرا که می توان متغیر Val٣ را بصورت Integer تعریف کرد. اما استفاده از تابع CInt() سبب می شود که کد به صورت ساده تری در بیاید.

اخطار اگر نتیجه یک عبارت محاسباتی از مرز مجاز یک نوع متغیر عبور کند (مثلا CInt(٣٣٠٠٠)) ، خطای سرریز رخ خواهد داد.

توابع اصلی ریاضی و مثلثاتی در VBA

در برنامه ای چون اکسل ، به عنوان برنامه نویس به تعداد زیادی تابع ریاضی نیاز دارید. اگر نیاز پیدا کردید، VBA چندین تابع مثلثاتی و ریاضی را برای استفاده در برنامه شما آماده کرده است. (جدول ٤,٣ این توابع را لیست کرده است.) کاربرد آنها کاملا واضح است. شما تنها آرگومان تابع را تعیین می کنید و آن نتیجه را برمی گرداند. در مثال زیر ، تابع Atn() (آرک تانژانت) مقدار عدد پی (π) را محاسبه می کند.این نتیج کاملا مفید هستند زیرا VBA مستیقما عدد پی را نمی شناسد. (شما می توانید از توابع کاربرگ برای محاسبه مقدار پی استفاده کنید.)

جدول ٤.٣ : توابع ریاضی VBA

تابع	توضیحات	مثال
Abs()	قدر مطلق یک عدد را ارائه می دهد.	Abs(-١٠) بر می گرداند ١٠
Atn()	آرک تانژانت مقدار (رادیان)	Atn(١) * ٤ بر می گرداند ٣,١٤١٥٩٢٦٥٣٥٨٩٧٩
Cos()	کسینوس	Cos(٠) بر می گرداند ١

تابع	توضیحات	مثال
Exp()	حاصل توان عدد بر پایه e	Exp(۱) بر می گرداند ۲,۷۱۸۲۸۱۸۲۸٤٥۹۰٥
Fix()	قسمت صحیح عدد	Fix(-۹,۱) بر می گرداند -۹
Int()	قسمت صحیح عدد (گرد به پایین)	Int(-۹,۱) بر می گرداند -۱۰
Log()	لگاریتم طبیعی	Log(۲,۷۱۸۲۸۱۸۲۸) بر می گرداند ۱
Rnd()	عدد تصادفی بین ۰ و ۱	
Sgn()	ساین (تابع علامت)	Sgn(-۹) بر می گرداند -۱
Sin()	سینوس	Sin(۳,۱٤۱٥۹۲٦٥/۲) بر می گرداند ۱
Sqr()	مربع عدد	Sqr(۹) بر می گرداند ۳
Tan()	تانژانت	Tan(۳,۱٤۱٥۹۲٦٥/٤) بر می گرداند ۱

```
Dim pi As Double
pi = ٤ * Atn(١)
```

تابع اعداد تصادفی Rnd() تابع متداولی است. تابع Rnd() یک عدد بین ۰ و ۱ را به طور تصادفی بر می گزینـد. شـما مـی توانید از مرزهای اختیاری برای تعیین عدد تصادفی استفاده کنید. اغلب ، هر فراخوانی تـابع Rnd() از عـدد تصـادفی قبلـی استفاده خواهد کرد.این قسمت کار قابل تکرار است. برای تصادفی سازی واقعی از عبارت Randomize استفاده کنید تـا از روی ساعت سیستم به تولید عدد تصادفی بپردازد.

```
Dim val١
Randomize
val١ = Int(١٠٠ * Rnd + ١)
```

شما می توانید از کد اخیر برای ایجاد اعداد صحیح تصادفی بین ۱ و ۱۰۰ استفاده کنید.

تغییر متون با توابع متنی

دستکاری و تغییر متن ها یکی از متداول ترین کارها به شما می آید. بـرای جسـتجوی مـورد خاصـی در یـک رشـته متنـی ، دستیابی به محتویات یک متن یا فقط شمارش تعداد کاراکترهای یک متن ، VBA دارای توابعی است که این کارها را بـرای شما انجام می دهد. کار با اکثر توابع آسان است. کافی است یک یا دو آرگومان آن را که مربوط به متن مورد نظـر مـی شـود تعیین کنید. این بخش تعداد زیادی از این توابع را در بر می گیرد. (جدول ٥,۳ را برای مشاهده تعداد زیـادی از ایـن توابـع ببینید.)

جدول ۵.۳ : توابع متنی VBA

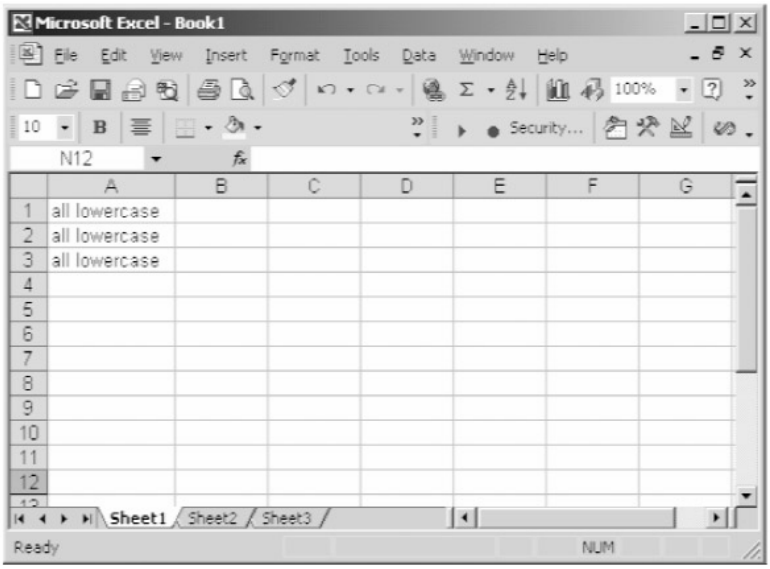
تابع	توضیح	مثال
Str()	تبدیل یک عدد به متن	Str(۱۲) بر می گرداند متن۱۲
Val()	تبدیل یک متن به عدد	Val("۵XJT") بر می گرداند ۵
LTrim(), RTrim(), Trim()	حذف فواصل قبل و بعد از متن	Trim("My String ") بر می گرداند "My String".
Left()	ارائه کاراکترهایی از سمت چپ متن	Left("Test",۲) بر می گرداند "Te".
Right()	ارائه کاراکترهایی از سمت راست متن	Right("Test",۲) بر می گرداند "st".
Mid()	ارائه کاراکترهایی از وسط متن	Mid("Test", ۲،۲) بر می گرداند "es".
InStr()	تعیین محل یک کاراکتر در متن	InStr("Test", "e") بر می گرداند ۲.
Len()	تعداد کاراکتر متن	Len("Test") بر می گرداند ٤.
LCase()	تبدیل حروف متن به حروف کوچک	LCase("ABCD") بر می گرداند "abcd".
UCase()	تبدیل حروف متن به حروف بزرگ	UCase("abcd") بر می گرداند "ABCD".
StrConv()	تبدیل متن بصورت تعیین شده با چندین گزینه انتخابی	Debug.Print StrConv("fred", vbProperCase) بر می گرداند "Fred".
StrComp()	ارائه نتیجه مقایسه دو متن	Debug.Print StrComp(str۱,str۲,vbTextCompare) بر می گرداند ۰ اگر هر دو متن برابر باشند، ۱- اگر متن اول بزرگتر باشد ، ۱ اگر متن کوچکتر از متن دوم باشد.
StrReverse()	ترتیب معکوس کاراکترهای متن	StrReverse("Test") بر می گرداند "tseT".
Format()	شکلبندی متن به صورت تعیین شده	Format("ABCD", "<") بر می گرداند "abcd".
Space()	ارائه متنی که پس از تعداد معینی فاصله واقع شده است	
Asc()	کد اسکی کاراکتر	Asc("A") بر می گرداند ٦٥.
Chr()	کاراکتر متناظر با کد اسکی	Chr(٦٥) بر می گرداند "A".
Dir()	نام فایل و محل قرار گیری آن به صورت متن	Dir("C:\Windows*.ini") بر می گرداند اولین فایلی را که با پسوند تعیین شده می باشد.
CurDir()	متنی که دایرکتوری فعلی را تعیین می کند	CurDir() بر می گرداند دایرکتوری فعلی

توابع شکلبندی متن

تعداد توابع متن بیش از اندازه زیاد است طوری که شما برای اجرای یک کار ، می توانید از چندین تابع مختلف استفاده کنید. این مطلب در مورد توابع شکلبندی نیز صادق است. (Format() , StrConv() , UCase() , LCase()) مثلا هر سه خط کد زیر متن های داده شده را با حروف کوچک شکلبندی می کنند.

```
Cells(١, "A").Value = LCase("ALL LOWERCASE")
Cells(٢, "A").Value = Format("ALL LOWERCASE ", "<")
Cells(٣, "A").Value = StrConv("ALL LOWERCASE ", vbLowerCase)
```

خروجی کد اخیر در شکل ۳,۳ نشان داده شده است. برای شکلبندی متن با حروف کوچک می توانید از هر یک از توابع LCase() , StrConv() , Format() استفاده کنید. تفاوت آنها در تعداد آرگومانها می باشد. تابع Format() دارای انعطاف بیشتری بوده و می تواند برای شکلبندی اعداد و تاریخ نیز بکار رود از اینرو برای این کار مناسب می باشد.



شکل ۳ . ۳ : تبدیل متون به حروف کوچک

توجه به جای بحث در مورد تمام آرگومانهای ضروری و اختیاری یک تابع ، من تنها به شرح موضوعی خواهم پرداخت که برای حل مشکل فعلی مسئله مورد نیاز است.

ترکیب متن : Val() و Str()

در VBA می توانید متون را با عملگر (+) یا امپرسند (&) با هم ترکیب کنید.جهت جلوگیری از اینکه عملگر (+) با جمع اشتباه گرفته نشود از عملگر (&) استفاده کنید. در کد زیر با توجه به عملگر (&) می توان نتیجه گرفت که مقدار متغیر newstr ، "٥٥" خواهد بود.

```
string١ = "٥"
string٢ = "٥"
```

```
newStr = string١ & string٢
```

شما می توانید از توابع Str() و Val() جهت تبدیل بین عدد و متن استفاده کنید. در اغلب موارد ، VBA نوع مناسب را به متغیرها اعمال خواهد کرد. در مثال زیر ، تابع Val() لازم نیست که قبل از اضافه شدن به num١ به عدد تبدیل شود زیرا VBA فرض می کند که عملگر جمع بکار رفته است. در کل به کار بردن Val() در این مورد ، کد را از هر ابهامی رها می کند.

```
num١ = ٥
str١ = "٥"
num٢ = num١ + Val(str١)
```

به طور مشابه ، استفاده از تابع Str() کمک می کند تا کد شما در هنگام ترکیب متن ها درست عمل کند.

```
zip = ٥٥٥٥٥
zipx = ٤٤٤٤
newzip = Str(zip) + Str(zipx)
```

بدون تابع Str() در آخرین خط کد، عمل انجام شده جمع محسوب می شود.استفاده از Str() این اطمینان را حاصل می کند که عمل انجام شده ترکیب به حساب می آید. البته، بهتر خواهد بود اگر از علامت امپرسند برای ترکیب استفاده می شد.

Asc() و Chr()

شما می توانید از توابع Chr() و Asc() برای تبدیل بین عدد و کاراکتر استفاده کنید (جدول ٦,٣ را برای برخی کدها ببینید.) مجموعه کدهای کاراکترها در حقیقت کدهای اسکی استاندارد هستند که ٢٥٦ کاراکتر دارند. البته بیشتر کاراکترها از سی VBA پشتیبانی نمی شوند. اغلب نیلز دارید که در یک کاربرگ اکسل بین سطرها و ستونهای خاصی دور بزنید. در این موقع می توان از تابع Chr() استفاده کرد. کد زیر را ببینید:

```
Range(Chr(١٥ + I) & "١").Value
```

جدول ٣ . ٦ : منتخبی از کدهای کاراکترها در VBA

کد کاراکتر	Character
٨	backspace
٩	tab
١٠	linefeed
١٣	carriage return

کد کاراکتر	Character
٣٢	[space]
٣٣	!
٣٤	"
٣٥	#
٣٦	$
٣٧	%
٣٨	&

کد کاراکتر	Character
٣٩	'
٤٠	(
٤١	)
٤٢	*
٤٣	+
٤٤	,
٤٥	-

کد کاراکتر	Character
٤٦	.
٤٧	/
٤٨	٠
٤٩	١
٥٠	٢
٥١	٣
٥٢	٤
٥٣	٥
٥٤	٦
٥٥	٧
٥٦	٨
٥٧	٩
٥٨	:
٥٩	;
٦٠	<
٦١	=
٦٢	>
٦٣	?
٦٤	@
٦٥	A
٦٦	B
٦٧	C
٦٨	D
٦٩	E
٧٠	F
٧١	G
٧٢	H

کد کاراکتر	Character
٧٣	I
٧٤	J
٧٥	K
٧٦	L
٧٧	M
٧٨	N
٧٩	O
٨٠	P
٨١	Q
٨٢	R
٨٣	S
٨٤	T
٨٥	U
٨٦	V
٨٧	W
٨٨	X
٨٩	Y
٩٠	Z
٩١	[
٩٢	\
٩٣	]
٩٤	^
٩٥	_
٩٦	`
٩٧	a
٩٨	b
٩٩	c

کد کاراکتر	Character
١٠٠	d
١٠١	e
١٠٢	f
١٠٣	g
١٠٤	h
١٠٥	i
١٠٦	j
١٠٧	k
١٠٨	l
١٠٩	m
١١٠	n
١١١	o
١١٢	p
١١٣	q
١١٤	r
١١٥	s
١١٦	t
١١٧	u
١١٨	v
١١٩	w
١٢٠	x
١٢١	y
١٢٢	z
١٢٣	{
١٢٤	|
١٢٥	}
١٢٦	~

در این مثال اگر I=٠ باشد در اینصورت ارجاع به سلول A١ خواهد بود زیرا مقدار ٦٥ کاراکتر A بزرگ را ارائه می دهد. تـا زمانی که I ما بین ٠ و ٢٥ قرار دارد کد قبلی به سلولهای متناظر ارجاع داد ه خواهد شد.(A-Z)

به طور مشابه ، می توانید از تابع Asc() استفاده کرده و کاراکتر ها را به اعداد متناظر شان تبدیل نمایید. متغیر num١ مقـدار صحیح ٦٥ را در مثال زیر خواهد گرفت:

```
num١ = Asc("A")
```

توجه می توانید در فصل ٦ در مورد حلقه ها در VBA جزئیات بیشتری را پیدا کنید.

توابع تودرتو (آشیانه ای)

شما می توانید توابع را تا هر درجه ای بصورت تو در تو بکار ببرید. البته ، خوانایی کد شما وحشتناک خواهد شد. کد زیر را مشاهده کنید و ببینید می توانید بفهمید که منظور آن چیست:

```
cell۲ = Mid(rangeStr, InStr(۱, rangeStr, ":") + ۱, _
     Len(rangeStr) - InStr(۱, rangeStr, ":"))
```

اگر فکر می کنید که خط اول کد متنی را ارئه می دهد و خط دوم کد به یک سلول ارجاع می دهد، شما درست فکر کرده اید. مثلا اگر محدوده کاربرگ توسط متغیر rangeStr عبارت باشد از "A۱:C۱۰" و متغیر Cell۲ مقدار C۱۰ را پس از اجرای کد دارا خواهد بود. در این مثال تابع Len() یکبار ، تابع Instr() دوبار درون تابع Mid() قرار گرفته اند. اولین عمل تابع Instr() برای تعیین تابع Mid() آغاز می شود. ترکیب Len() و Instr() و سپس طول متن توسط Mid() شمارش می شود. به طور واضح این خط کد برای خواندن مشکل می باشد و تا حد امکان باید از نوشتن توابع تودرتو اجتناب کرد.

استفاده از توابع تاریخ و زمان VBA

VBA دارای چندین تابع برای کارکردن و تغییر تاریخ ها و زمان ها می باشد. برخی از این توابع نیازی به آرگومان نداشته و برخی بدون پرانتز در انتهای اسم تابع می باشند. توابع دیگر آرگومانهایی را به صورت عددی یا متنی نیاز دارند. جدول ۷,۳ لیست توابعی را که برای کار با تاریخ طراحی شده اند ، نشان می دهد.

جدول ۳ . ۷: توابع تاریخ و زمان VBA

تابع	توضیحات
Time	زمان فعلی سیستم
Now	زمان و تاریخ فعلی سیستم
Date	تاریخ فعلی سیستم
DateDiff()	تفاضل بین دو تاریخ
DateAdd()	تاریخ اضافه شده به یک فاصله زمانی
Year()	عدد بیانگر سال
Month()	عددی بین ۱ تا ۱۲ بیانگر ماه سال
Weekday()	عددی بین ۱ تا ۷ بیانگر روز هفته
Day()	عددی بین ۱ تا ۳۱ بیانگر روز ماه
Hour()	عددی بین ۰ تا ۲۳ بیانگر ساعت

تابع	توضیحات
Minute()	عددی بین ۰ تا ۵۹ بیانگر دقیقه
Second()	عددی بین ۰ تا ۵۹ بیانگر ثانیه
TimeSerial()	تاریخ متناظر با ساعت ، دقیقه و ثانیه داده شده
DateSerial()	تاریخ متناظر با سال ، ماه و روز
TimeValue()	تاریخ حاوی زمان داده شده را نشان می دهد
DateValue()	تاریخ حاوی تاریخ داده شده را نشان می دهد

سه تابع Time ، Now و Date دارای خروجی زمان ، زمان و تاریخ ، و تاریخ می باشند. (توجه : این سه تابع بدون آرگومان می باشند پس من برای آنها پرانتز درج نکرده ام). اغلب توابع تاریخ به این صورت طراحی شده اند که قسمتی از یک تاریخ را بر گردانند یا زمان مربوطه را محاسبه کنند. توابع Year() ، Month() ، Weekday() ، Day() ، Hour() ، Minute() و Second() همگی اعداد صحیحی بر اساس داده تاریخ ورودی یا زمان به دست می آورد. آرگومان مربوط به این توابع باید یک تاریخ یا یک متن باشد که بصورت تاریخ شکلبندی شده باشد. مثال زیر از تابع Month() استفاده می کند تا ماه جاری را نمایش دهد. تابع Now زمان و تاریخ فعلی سیستم را به تابع Month() اعلام می کند.

```
curMonth = Month(Now)      'returns current month as an integer
prevMonth = Month("Feb ٤, ١٩٠٠")      'returns ٢
```

توابع دیگری که در جدول ۷,۳ لیست شده اند از متن یا عدد ورودی ، تاریخی را به عنوان خروجی ارائه می دهند. توابع TimeSerial() ، DateSerial() ، TimeValue() و DateValue() همگی تاریخ هایی را برمی گردانند که بیانگر زمان یا تاریخ می باشد.

```
myTime = TimeValue("٩:٠٠")      'returns ٩:٠٠:٠٠ AM
myDate = DateValue("Nov, ٢٣ ٠٤")      'returns ١١/٢٣/٢٠٠٤
myTime = TimeSerial(٩, ٠, ٠)       'returns ٩:٠٠:٠٠ AM
myDate = DateSerial(٤, ١١, ٢٣)     'returns ١١/٢٣/٢٠٠٤
```

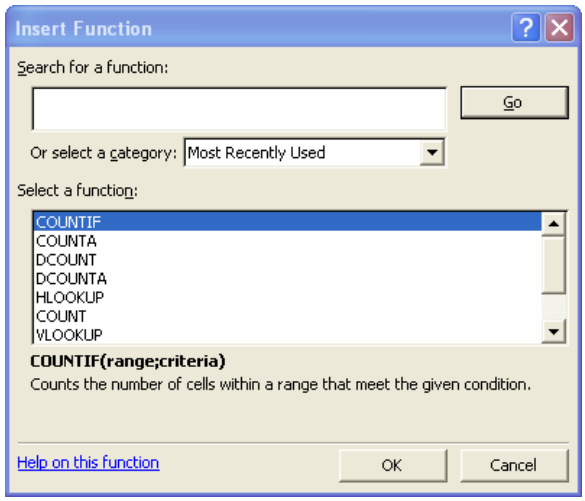
تابع مفید دیگر DateDiff() می باشد. همان طور که از اسم آن می توان حدس زد، تابع مذکور اختلاف بین دو تاریخ را محاسبه می کند. واحد اختلاف بر حسب کاراکتر مخففی است که اعلام می کنید. (s برای ثانیه ، m برای دقیقه و yyyy برای سال و الی آخر.)

```
myInterval = DateDiff("s", Now, "١٠/١٠/٠٣")
myInterval = DateDiff("m", Now, "١٠/١٠/٠٣")
myInterval = DateDiff("yyyy", Now, "١٠/١٠/٠٣")
```

مثال اخیر ، تعداد ثانیه ها ، ماهها و سالهای بین ۱۰ اکتبر ۲۰۰۳ تا حالا را محاسبه می کند. دقت کنید که از متغیر مناسبی استفاده کنید تا جواب باعث سرریز شدن محدوده مجاز متغیر نگردد.

استفاده از توابع کاربرگ اکسل در کد VBA

اگر شما یک کاربر اکسل باشید، حتما از وکود توابعی که می تواند در فرهای صفحه گسترده بکار روند، اطلاع دارید. برنامه اکسل بیش از ۳۰۰ تابع دارد که می توان با آنها در مورد داده های عددی و متنی و تاریخی عملیات مختلف و حتی ریاضی انجام داد. اکثر این توابع یک یا چند آرگومان دریافت می کنند و یک خروجی ایجاد می کنند. (ضمیمه A را ببینید.)
شما می توانید لیستی از توابع موجود کاربرگ را در کادر محاوره ای Insert Function مشاهده کنید.



شکل ۳ . ٤ : کادر محاوره ای Insert Function اکسل

اکثر توابع کاربرگ ، در راستای تکمیل توابع VBA می باشد مثلا تابع PI() که مقدار عدد پی را محاسبه می کند. توابعی چون SORT() نیز کار توابع VBA را تکرار می کند. شما می توانید از توابع کاربرگ بصورت زیر استفاده کنید:

```
myAvg = Application.WorksheetFunction.Average(٥, ٦, ٧)     'returns ٦
mySum = Application.WorksheetFunction.Sum(٥, ٦, ٧)      'returns ١٨
```

شما باید از طریق شی worksheetFunction و Application به این توابع دسترسی پیدا کنید. سپس نام تابع را اضافه کنید و آرگومانهای آن را تعیین نمایید. شی Application در فصل ۷ بیشتر بحث شده است که در آنجا چند مثال نیز هست.

خلاصه

این فصل به طور دقیق ، توابع MsgBox() و InputBox() را که برای ورود و خروج در VBA بکار می رود، مورد بررسی قرار داد. سپس برخی توابع ریاضی ، تاریخ و زمان را بررسی کردم. همچنین از توابع تبدیل نوع متغیر نیز صحبت

کردیم. توابعی که بحث شد ، توابعی متداول هستند اما درصد کمـی از آنهـا را شـما دیدیـد و توابـع بیشـتری در VBA در دسترس است. در فصل های بعدی توابع دیگری را مشاهده خواهیم کرد.

فصل ٤ : درک روشهای VBA

فصل ٥:شاخه بندی برنامه با ساختارهای تصمیم گیری

فصل ٦: ایجاد حلقه ها و آرایه ها در VBA

فصل ٧: اشیا VBA که مخصوص برنامه نویسی هستند

فصل ٨: درک فایل های I / O ، خطایابی و رفع خطا در VBA

فصل ٩: منوه و نوار ابزارهای سفارشی

فصل ١٠ : ایجاد و دستکاری نمودارهای اکسل

فصل ١١: مدولهای کلاس VBA ، ایجاد اشیا سفارشی

فصل ١٢ : ساختن مبدل اکسل به HTML

فصل١٣ : ورود و خروج داده ها بین اکسل و اکسس

فصل١٤ : ایجاد و کاربرد جداول محوری اکسل برای آنالیز سریع داده ها

فصل ١٥ : آنالیز معاملات شرکت Pear Tree Used Cars

فصل ١٦ : پرس و جوهای VB و اشیا مرتبط با وب در VBA

فصل ١٧ : دسترسی به Windows API توسط VBA

فصل ١٨ : ساختن Stock Ticker

فصل ١٩ : کاهش کار با آنالیز خودکار داده ها

فصل ٢٠ : آنالیز پیشرفاه داده ها توسط انطباق منحنی و نمودارهای فعال در VBA

فصل ٢١ : خودکار سازی آنالیز داده ها و برازش منحنی توسط Excel – VBA

ضمیمه A : آموزش اصول بنیادین اکسل

ضمیمه B : آموزش اصول XHTML / HTML

ضمیمه C : آموزش اصول SQL