آموزش کیوبیسیک Qbasic

آموزش : QBASIC جلسه اول

جلسه اول

بیسیک چیست ؟

****

با توجه به گستردگی روزافزون علم کامپیوتر در تمام ابعاد زندگی انسان و نیاز به یادگیری این علم نوین و از آنجایی که زبان QBASIC (نسخه جدید تر BASIC) زبانی ساده و در عین حال قدرتمند برای آموزش مفاهیم برنامه نویسی است ، در این وبلاگ اقدام به آموزش قدم به قدم این زبان کرده ام . امید است مورد استفاده دوستان عزیز قرار گیرد .

****

در این یادداشت ، می توانید با تاریخچه و خانواده BASIC آشنا شوید . در یادداشت های بعدی اصول برنامه نویسی QBASIC را شروع خواهم کرد .
بیسیک را می توان متداولترین زبان برنامه نویسی کامپیوتر در چند دهه اخیر به حساب آورد . عموما این زبان اولین زبانی است که کارآموزان ، دانش آموزان و دانشجویان با آن آشنا می شوند و برنامه نویسان با ذوق ، زبان فوق را به علت سهولت استفاده و قابلیت های مناسب آن دوست دارند . کلمه BASIC مخفف عبارت Beginner's All-purpose Symbolic Instruction Code به معنای "کد حاوی دستورات نمادین همه منظوره مخصوص نوآموزان" می باشد . نسخه ی اصلی آن در کالج دارتموث (Dartmouth) در سال ۱۹۶۴ برای نوآموزان به عنوان یک زبان آموزشی ایجاد گشت . علت اصلی ایجاد این زبان را می توان در پیچیدگی های زبان های متداول از قبیل Fortran، Assembly و غیره جستجو کرد.

برنامه نویسی ساخت یافته :
همزمان با آمدن زبان هایی که دارای ساختارهای کنترلی کاربردی (مانند C و پاسکال) بودند و امکان طراحی زیباتر ، دقیق تر و مناسبتر برنامه ها را به برنامه نویسان دادند نیاز به نسخه اول

BASIC کم شده و نسخه های جدیدی از آن مانند QBASIC برای رفع کمبودهای نسخه اول به بازار عرضه شد که ضمن حفظ سادگی بیسیک ، قابلیت برنامه نویسی ساخت یافته را نیز به آن افزودند . برنامه نویسی ساخت یافته روشی از برنامه نویسی است که در آن برنامه را به قطعات کوچکتر تقسیم بندی می کنند و هر قطعه کار مشخصی را انجام می دهد . همچنین در این روش اجرای برنامه و کنترل ترتیب اجرای دستورالعمل ها با استفاده از کلمات و سمبل های جدید آسان تر شده است . اصلی ترین مزیت برنامه نویسی ساخت یافته را می توان در سادگی امکان توسعه و نگهداری برنامه ها دانست . برنامه هایی که با زبانهای غیر ساخت یافته نوشته می شدند ، به سادگی قابل تغییر نبودند و گاهی فراتر رفتن میزان تغییرات از حد مشخصی ، نیاز به بازنویسی کامل برنامه را ایجاب می کرد . در نتیجه برنامه نویسی ساخت یافته ، راه حل این مشکل بود .

مفسر و مترجم :

پس از آن که برنامه ای در محیط زبان برنامه نویسی نوشته شد ، باید توسط کامپیوتر اجرا شود . در دنیای زبان های برنامه نویسی دو روش مختلف برای اجرای برنامه ها وجود دارد :

روش اول : مفسر (Interpreter) که برنامه ها را در زمان اجرا خط به خط به کدهای ماشین (صفر و یک) تبدیل می کند .

روش دوم : مترجم (Compiler) که کل برنامه را یکجا به کدهای ماشین تبدیل می کنند . مترجم ها معمولا فایل هایی با پسوند EXE یا COM ایجاد می کنند که این فایل ها قابلیت اجرا در محیط سیستم عامل را دارند و نیازی به خود زبان برنامه نویسی در هنگام اجرا ندارند ، در حالی که برای اجرای برنامه ای که با زبان های دارای مفسر نوشته شده است ، نیاز به وجود فایل های زبان برنامه نویسی نیز می باشد .

بیسیک نیز جزو زبان های تفسیری محسوب می شود . زبان های تفسیری علی رغم سادگی در پیاده سازی زبان چند اشکال مهم دارند :

کندی اجرای برنامه ها در مقایسه با زبان های مترجم دار نیاز به تفسیر هر دستور حتی اگر در یک حلقه آن دستور را هزاران بار داشته باشیم .

QBASIC علاوه بر آنکه مفسر است ، با بهینه سازی هایی که در آن صورت گرفته ، کل برنامه را یکجا تفسیر می کند . یعنی دارای کامپایلر نیز می باشد و این امر تا حد زیادی کارایی آن را بالا برده است .

QBASIC مفسر بیسیکی است که به همراه MS-DOS نسخه ۵ و بالاتر ارائه می گردد .

شـــرکت مایکروســـافت تولیـــد کننـــده QBASIC بـــا تکمیـــل ایـــن محصـــول ، نســـخه هـــای دیگـــری از بیسیک را روانه بازار کرده است که عبارتند از :

BASIC Quick و BASIC PDS و VISUAL BASIC .

آموزش : QBASIC جلسه دوم

جلسه دوم

در این یادداشت می خواهم به معرفی انواع داده ها بپردازم. در Qbasic لزومی برای معرفی داده ها وجود ندارد ولی بهتر است در ابتدای برنامه آنها را معرفی کنیم .

انواع داده ها :

داده ها به طور کلی به دو نوع ثابت و متغیر تقسیم می شوند که هرکدام دارای دو دسته عددی و رشته ای هستند .

*داده ثابت :

داده ای که در طول برنامه مقدار آن تغییر نمی کند . برای معرفی این نوع داده از دستور CONST استفاده می کنیم

مثال :

```
pi=3.14 CONST
```

*داده متغیر:
داده ای است که می تواند بر حسب منطق برنامه و در جایگاه های مختلف مقادیر متفاوتی بپذیرد که همانطور که گفتم دو نوع دارد : متغیر عددی و متغیر رشته ای .
*متغیر عددی :
متغیر های عددی دارای ۴ نوع مختلف هستند :

نام	مخفف	حافظه اشغال کننده	علامت	نوع
عدد صحیح معمولی	INT	BYTE ۲	%	INTEGER
عدد صحیح بلند	LNG	BYTE ۴	&	LONG

عدد اعشاری با دقت معمولی	SNG	BYTE ۴	!	SINGLE
عدد اعشاری با دقت مضاعف	DBL	BYTE ۸	#	DOUBLE

اگر بخواهیم محاسبات برنامه با دقت بالایی انجام شود ، باید متغیرها را از نوع مناسب (مثلا اعشاری با دقت مضاعف) در ابتدای برنامه به وسیله دستور DIM تعریف کنیم .

مثال : برای این که متغیر A از نوع اعشاری با دقت معمولی باشد ، در ابتدای برنامه این دستور را می نویسیم :

SINGLE DIM A AS

راه دوم این است که هرجا نیاز به استفاده از A بود ، آن را به صورت A! به کار می بریم .

*متغیرهای رشته ای

همان طور که از نام این متغیر مشخص است،برای نگهداری رشته ای از حروف به کار می رود و دو نوع دارد :

نام	مخفف	علامت	نوع
متغیر رشته ای با طول متغیر	STR	$	STRING
متغیر رشته ای با طول ثابت	STR	$	STRING*N%

فرض کنید می خواهیم متغیر رشته ای به نام name و متغیر رشته ای با طول ۱۰ کاراکتر به نام family تعریف کنیم . مانند متغیرهای عددی از دستورِ DIM استفاده می کنیم :

STRING DIM name AS

family AS STRING*10 DIM

حتما متوجه شده اید که N تعداد حروف متغیر را بیان می کند و طبیعتا از نوع % یعنی عدد صحیح معمولی است .

اگر بدون معرفی متغیر رشته ای را در برنامه استفاده کنیم ، Qbasic به طور خودکار آن را مساوی " " یعنی یک رشته خالی قرار می دهد و اگر متغیر از نوع عددی باشد ، Qbasic به طور پیش فرض آن را مساوی صفر قرار می دهد.

آموزش : QBASIC جلسه سوم

جلسه سوم

حال می خواهیم اولین برنامه خود را در QBasic بنویسیم . برای این کار نیاز به معرفی چند دستور داریم :

۱ -دستور PRINT برای چاپ عدد رشته و یا مقادیر متغیرها به کار می رود . فقط فراموش نکنید اگر می خواهید پیغامی نمایش دهید رشته مورد نظر را حتما داخل علامت کوتیشن (") قرار دهید برای مثال دستور زیر عبارت This is my first programرا چاپ می کند .

```
" PRINT "This is my first pogram
```

برای ترکیب چند رشته یا متغیر باید بین متغیرها / رشته ها از ; یا , استفاده کنید . در ادامه تفاوت این دو را بیان خواهیم کرد .

۲ -دستور INPUT برای دریافت مقادیر (عددی و رشته ای) از کاربر به کار می رود . برای مثال دستور INPUT a !یک عدد اعشاری با دقت مضاعف از کاربر دریافت می کند . وقتی دستور INPUT اجرا می شود به طور خودکار علامت ؟ روی صفحه نمایش چاپ شده و برنامه منتظر ورود داده از طرف کاربر می ماند . اما این دستور را می توان طور دیگری نیز به کار برد . مثلا فرض کنید می خواهیم به کاربر پیغام دهیم :"یک عدد وارد کن" و سپس این عدد را در متغیر a قرار دهیم . برای این کار باید ابتدا یک دستور PRINTبرای چاپ رشته و سپس دستور INPUT را برای دریافت عدد به کار ببریم . مانند زیر :

```
"PRINT "Enter a number
```

INPUT a

اما می توان این کار را در یک خط انجام داد ! دستور INPUT اجازه چاپ پیغام را نیز به ما می دهد . به این نکته توجه داشته باشید : بین پیغام و نام متغیر باید ; یا , قرار دهیم . اگر از ; استفاده کنیم بعد از چاپ پیغام یک علامت سوال چاپ می شود و نشانگر بلافاصله بعد از علامت سوال قرار می گیرد ولی در صورت استفاده از , علامت سوال خودکار چاپ نمی شود و نشانگر بعد از یک tabفاصله روبه روی پیغام قرار می گیرد . به دستورات زیر که ترکیب دو دستور بالاست توجه کنید و تفاوت های آنها را در خروجی مشاهده کنید (فرض کنید کاربر عدد ۲ را وارد می کند)

2 INPUT "Enter a number" , a \\ khorooji=> Enter a number

INPUT "Enter a number" ; a \\ khorooji=> Enter a number?2

حال دو دستور ساده دیگر را هم یاد بگیرید :

۱ : REM -این دستور برای اضافه کردن توضیحات به برنامه به کار می رود و اصولا زمانی که مترجم برنامه به این دستور می رسد بلافاصله به خط بعدی می رود و عبارت جلوی REM را نادیده می گیرد .

۲ : END -استفاده از این دستور نیز مانند REM اختیاری است و نشان دهنده پایان برنامه است .

با توجه به درس جلسه قبل و این جلسه با هم برنامه ای می نویسیم که دو عدد از کاربر دریافت کرده و حاصل جمع آنها را نمایش دهد

```
REM this program can add two numbers
".PRINT "I can add two numbers . please enter them
INPUT "Enter first number:",a
INPUT "Enter second number:",b
c=a+b
PRINT "Sum is :";c
END
```

مثالی از خروجی برنامه :

```
. I can add two numbers . please enter them
Enter first number: 31
Enter second number: 24
Sum is : 55
```

تمرین : برنامه ای بنویسید که ۳ عدد را دریافت کند و الف) حاصل جمع آنها را چاپ کند . ب) میانگین آنها را چاپ کند .

نکته : اگر برنامه بالا را بنویسید و اجرا نمایید متوجه می شوید که اگر برای متغیرها علامت خاصی قرار ندهید QBasic آنها را به عنوان عدد صحیح (integer)می شناسد و در نتیجه میانگین نیز مقدار صحیحی نمایش داده می شود . برای دقیق تر شدن خروجی برنامه و اینکه میانگین عددی اعشاری باشد باید متغیرهای مربوط به سه عدد دریافتی , حاصل جمع و میانگین از نوع اعشاری باشد . یعنی باید متغیرها را مانند a! یا a# به کار ببرید تا خروجی

برنامه دقیق و برابر مقدار واقعی میانگین باشد .

آموزش : QBASIC جلسه چهارم

جلسه چهارم

ابتدا حل تمرین جلسه قبل . راه اول این بود که متغیر ها را به صورت a! به کار ببریم (مانند برنامه زیر) و راه دوم این بود که با توجه به درس جلسه دوم در ابتدای برنامه تمام متغیرها را از نوع اعشاری تعریف نماییم برای مثال عبارت DIM a AS SINGLE را در ابتدای برنامه بنویسیم تا هرجا از a استفاده کردیم نوع آن اعشاری باشد .در این روش نیازی نیست در هر بار استفاده از متغیر علامت نوع را در انتهای آن بنویسیم.

```
***REM ***program number 1
"PRINT" enter three number
!INPUT a! , b! , c
!sum! = a! + b! + c
!PRINT" Sum is : ";sum
PRINT "Average is;" : sum!/3
END
```

حال بررسی ساختارهای کنترلی که یکی از پرکاربردترین دستورها هستند را شروع می کنم .

۱- ساختار IF

این دستور برای انجام کاری در صورت درست بودن شرطی به کار می رود . شکل کلی آن به صورت زیر است :

IF شرط ۱ THEN

کارهایی که در صورت برقرار بودن شرط ۱ باید انجام شود .

ELSE IF شرط ۲ THEN

کارهایی که در صورت برقرار بودن شرط ۲ باید انجام شود .

...

ELSE

کارهایی که در صورت برقرار نبودن هیچ یک از شرط های بالا باید انجام شود .

IF END

مثال ۱: فرض کنید عددی از کاربر دریافت کرده و می خواهیم مثبت یا منفی بودن آن را مشابه تابع(x)sgnدر ریاضیات اعلام کنیم . (در صورت مثبت بودن ، عدد ۱ / منفی بودن ، عدد ۱- / و اگر صفر وارد شد ، عدد ضفر چاپ می شود) . دستور زیر این امکان را فراهم می سازد : (فرض کنید عدد در n ذخیره شده است)

```
IF n ۰<THEN
"PRINT "1
ELSE IF n ۰>THEN
"PRINT ۱-"
Else
"PRINT "0
END IF
```

نکته : اگر می خواهید فقط یک شرط را بررسی کنید (مثلا فقط در صورت مثبت بودن n عدد ۱ چاپ شود و در غیر این صورت هیچ کاری انجام نشود) می توانید دستورIF را در یک خط بدون نیاز به END IF بنویسید . مانند زیر :

```
"IF n<۰ 1" PRINT THEN
```

۲- DO...WHILE

این ساختار کارهایی را تا زمانی که شرطی برقرار باشد انجام می دهد . شکل کلی آن به صورت زیر است :

```
DO WHILE شرط
کارهایی که باید انجام شوند
LOOP
```

عبارت LOOP نشان دهنده پایان حلقه است . این ساختار دارای انواع دیگری نیز هست . مثلا در نوع معرفی شده کارها فقط زمانی انجام می شوند که شرط برقرار باشد و اگر شرط برقرار نباشد برنامه به خط بعد از LOOP می رود . اما نوع دیگری از این ساختار وجود دارد که ابتدا یک بار کار را انجام می دهد و سپس شرط را چک می کند . شکل کلی آن به صورت زیر است :

```
DO
کارهایی که باید انجام شوند
LOOP WHILE شرط
```

می توان به جای عبارت WHILE از UNTIL استفاده کرد اما در این صورت باید نقیض شرط را بنویسیم .

مثال ۲: می خواهیم تا زمانی که کلمه وارد شده از طرف کاربر مخالف رمز عبور (مثلا kamyar) است ، رمز دوباره دریافت شود . به روش های مختلف نوشتن این برنامه توجه کنید :

```
1) $INPUT pass
   "DO WHILE pass"<>$kamyar
   $INPUT pass
```

```
LOOP
 ---------------------------------------
$ (۲INPUT pass
"DO UNTIL pass$="kamyar
$INPUT pass
LOOP
 ---------------------------------------
 (۳DO
$INPUT pass
"LOOP WHILE pass"<>kamyar
 ----------------------------------------
 (۴DO
$INPUT pass
"LOOP UNTIL pass$="kamyar
```

همانطور که مشاهده می کنید هر ۴ بخش یک کار را انجام می دهند اما بخش های ۳ و ۴ از ۱ و ۲ بهترند زیرا متن برنامه آنها کوتاه تر است . بسته به منطق برنامه باید تشخیص دهید کدام روش بهتر است .

مثال ۳: برنامه ای بنویسید که مجموع اعداد از ۱ تا ۱۰۰۰ را حساب کند .

```
counter = 1
sum ۰ =
DO WHILE counter ۱۰۰۰=>
sum=sum+counter
counter=counter+1
LOOP
PRINT "Sum of numbers between 1 ۱۰۰۰ & is : ";sum
END
```

مثال ۴: برنامه ای بنویسید که میانگین نمرات را تا زمانی که عدد ۱- وارد نشده ، حساب کند .

```
counter=1
sum=0
DO
INPUT "Enter grade",grade
sum=sum+grade
```

```
counter=counter+1
LOOP WHILE grade ۱-<>
IF sum ۱-<>THEN
PRINT "average is";(sum+1)/counter
ELSE
!"PRINT "there isn't any grade
END IF
END
```

توضیح : در قسمت چاپ میانگین مقدار میانگین را برابر sum+1)/counter قرار دادیم زیرا وقتی عدد ۱- وارد می شود ابتدا با sum جمع شده و سپس شرط چک می شود و از حلقه بیرون می آید پس برای محاسبه میانگین اعدادی که قبل از ۱- وارد شده اند باید ۱- را از sum حذف کرد یعنی به sum یک مقدار اضافه کنیم.

FOR-۳

این ساختار شبیه ساختار WHILE است با این تفاوت که برای ایجاد حلقه هایی که دارای شمارنده هستند به کار می رود . می توانید شمارنده حلقه را درون این دستور مقدار دهی اولیه کنید . همچنین مقدار تغییر شمارنده (یکی یکی یا دوتا دوتا یا ...) نیز با دستور STEP قابل تنظیم است . شکل کلی این دستور به شکل زیر است :

FOR شمارنده = مقدار اولیه TO مقدار نهایی STEP مقدار تغییر شمارنده

کارهایی که باید در طول حلقه انجام شود

NEXT متغیر شمارنده

مثال ۴ : مثال ۳ را با این دستور می نویسیم :

```
FOR i=1 TO ۱۰۰۰ STEP 1
sum=sum+i
NEXT i
PRINT "Sum of numbers between 1 ۱۰۰۰ & is;" : sum
END
```

نکته : اگر مقدار تغییر شمارنده برابر یک است ، می توانید STEP 1 را ننویسید .

تمرین :

۱- برنامه ای بنویسید که مجموع اعداد زوج سه رقمی را چاپ کند .

۲- برنامه ای بنویسید که ابتدا رمز عبور را از کاربر بخواهد . اگر رمزabc بود آنگاه ۱۰ عدد را بخواند و max آنها را چاپ کند . اگر رمز اشتباه بود ، پیغام مناسبی نمایش داده شده و رمز عبور دوباره خواسته شود .

۳- برنامه ای بنویسید که رمز عبوری را از کاربر بخواهد . اگر رمز درست بود پیغام "your password is correct" را چاپ کند . برنامه باید فقط ۳ بار از کاربر رمز را بپرسد . اگر کاربر نتوانست رمز را در این ۳ بار وارد کند پیغام "Sorry! you can't continue" چاپ شود .

آموزش : QBASIC جلسه پنجم

جلسه پنجم

حل تمرین جلسه قبل :

۱- برنامه بسیار ساده و به صورت زیر است :

```
sum=0
FOR i=100 TO 998 STEP 2
sum=sum+i
NEXT i
PRINT "Sum is,": sum
```

```
DO
$INPUT "Enter pass";pass
IF pass"<>$abc "THEN
"PRINT "Your password is not correct ! try again
END IF
"LOOP WHILE pass>$abc<max THEN max=a
NEXT i
END
```

*نکات : تا زمانی که کاربر رمز عبور را درست وارد نکرده ، حلقه تکرار می شود و تنها زمانی که رمز به درستی وارد شود ، برنامه به خط بعد از LOOP می رود و دستورات بعدی انجام می شود . برای تعیین max دسته ای از اعداد ، ابتدا اولین عدد ورودی را در max قرار می دهیم سپس بقیه اعداد را در یک حلقه از کاربر دریافت کرده ، بعد از دریافت هرکدام ، آنها با max مقایسه می شوند تا اگر بیشتر از max بودند ، در آن قرار گیرند . دقت کنید چون یک عدد بیرون حلقه دریافت کرده ایم ۹ عدد دیگر باقیمانده است (شرط حلقه) برای به دست آوردن min نیز از همین روش استفاده می کنیم .

-۳

```
c=1
DO
$INPUT"Enter password",pass
c=c+1
LOOP WHILE pass "۱۲۳"<>$AND c ۳=>
IF pass$="123" THEN
!"PRINT "your password is correct
ELSE
"PRINT "Sorry ! you can't continue
END
```

نکات : در شرط حلقه از عملگر منطقی AND استفاده شده زیرا استفاده از AND سبب می شود اگر یکی از شرطها هم غلط باشد ، کل عبارت نادرست شود . (عینا مانند استفاده از "و" در منطق ریاضی) یعنی در هریک از دو صورتی که pass$="abc" (رمز درست وارد شود) یا c (شمارنده) بزرگتر از ۳ شود ، برنامه از حلقه خارج می شود و با توجه این که بر اثر درست وارد شدن رمز از حلقه خارج شده ایم یا بر اثر تمام شدن تعداد حدس ها ، پیغام مناسب چاپ می شود .

و اما درس جدید :

در درس امروز ابتدا به معرفی آرایه می پردازیم .

فرض کنید می خواهیم n عدد را به صورت صعودی مرتب کنیم . برای ذخیره سازی این اعداد باید n متغیر به نام های a1,a2,a3,...,an تعریف کنیم . نوشتن برنامه ای که بتواند این n عدد را مرتب کند مخصوصا زمانی که مقدار n بزرگ باشد ، بسیار مشکل و تقریبا غیر ممکن است . در چنین مواردی از آرایه ها یا ماتریس ها استفاده می کنیم . به کمک آرایه ها می توان تعداد مشخصی متغیر همنوع را فقط با یک نام و مشخص کردن اندیس آنها به کار برد . برای مثال می توان ده عدد را در متغیری به نام N از نوع آرایه ای به طول ۱۰ ذخیره کرد :

۳۲۵

۱۴۵

۲۳

۹۶۵

۳

۴۷

۱۰

۰

۷۱

۴۰۰

N

مثلا N(1)=325 و N(9)=71

برای تعریف آرایه از دستور DIM به شکل کلی زیر استفاده می شود :

[نوع داده AS] (دامنه یا تعداد عناصر) نام متغیر آرایه DIM

(عبارت داخل [ ] اختیاری است)

به مثال های زیر توجه کنید :

ایجاد آرایه ای به نام B و دارای ۲۰ عنصر : DIM B(1 TO 20) یا DIM B(20)

ایجاد آرایه ای به نام M از نوع عدد صحیح و دارای ۵۰ عنصر : DIM M%(50)

ایجاد آرایه ای به نام Grade از نوع اعشاری ساده دارای ۳۵ عنصر : DIM Grade(35) AS SINGLE

مثال ۱ : برنامه ای بنویسید که ۱۴ نمره یک دانش آموز را در آرایه ذخیره کرده و مجموع و میانگین نمرات را چاپ کند .

```
DIM Grade(1 TO 14 ) AS INTEGER
sum!=0
FOR i=1 TO 14
%)INPUT "Grade : "Grade(i
%)sum = !sum! + Grade(i
NEXT i
average!=sum!/14
!PRINT "sum = ";sum
!PRINT "average = ";average
```

این برنامه را می توانستید با دریافت نمرات در حلقه یا دریافت تک تک نمرات (۱۴ بار) نیز بنویسید . اما فرض کنید کاربر بخواهد نمره سوم یا ششم خود را مشاهده کند . در این صورت باید از آرایه ها استفاده کنید زیرا در روش قبلی امکان دسترسی به تک تک اطلاعات وجود نداشت ولی حالا با افزودن قطعه کد زیر به برنامه بالا می توان برنامه ای ساخت که نمره دلخواه کاربر را نمایش دهد :

```
INPUT "which grade do you want to see?",a
)PRINT "grade ";a;" is;" = Grade(a
```

آرایه های دوبعدی و چند بعدی :

در مواقعی نیاز به آرایه هایی داریم که هر عنصر آن دارای دو بعد باشد . مثلا برای ذخیره مختصات چند نقطه نیاز به یک آرایه دو بعدی داریم تا مختصات x و y نقطه را ذخیره کنیم . تعریف آرایه های n بعدی به طور کلی به شکل زیر است : (A1,A2,A3,...,An و B1,B2 و ... همگی عدد هستند)

DIM (A1 TO A2,B1 TO B2, ... , Z1 TO Z2) یا DIM A1,A2,A3,...,An)

مثال ۲ : برنامه ای بنویسید که جدول ضرب اعداد از یک تا ده را در یک آرایه دوبعدی ذخیره نماید .

```
DIM N(10,10)
FOR I%=1 TO 10
FOR J%=1 TO 10
N(I%,J%)=I%*J%
NEXT J
NEXT I
```

مثلا N(3,5)=15 .

تمرین :

این جلسه تنها یک تمرین نسبتا مشکل را عنوان می کنم تا در تعطیلات روی آن فکر کنید ! : برنامه ای بنویسید که ۱۰ عدد را دریافت کند و آنها را به ترتیب به صورت صعودی مرتب کند . برنامه را برای n عدد تعمیم دهید . (راهنمایی : دستور SWAP a,b مقدار دو متغیر را عوض می کند . یعنی اگر a=1,b=2 باشد بعد از اجرای این دستور a=2,b=1 خواهد بود . از این دستور می توانید برای جابه جایی عناصر آرایه به منظور مرتب سازی استفاده کنید .)

دستورات : QBASIC جلسه اول

CLS : این دستور زمانی بکار می رود که نیاز باشد صفحه نمایش پاک گردد . معمولا در ابتدای هر برنامه نوشته می شود .

LET : برای انجام کارهای جایگزینی و محاسباتی

MOD : باقیمانده تقسیم بصورت ۱۰ mod 2 جواب صفر است

\ (Back Slash) علامت کسر وارونه : تقسیم جزء صحیح بصورت ۱۰ \ ۳ که جواب ۳ می شود

INPUT : برای ورودی (متغییر عددی مانند m ,n , b , a , x, Sum, Min , Max, و ... . متغییر نشانه ای(جلوی متغییر عددی علامت $ اضافه می شود مانند City , Name$, b$, a)

GOTO n:برو به خط n در اول خطی که قرار است دستور از آنجا اجرا شود عدد دلخواه n را مینویسیم

PRINT : دستور خروجی برای به نمایش در آوردن نتیجه برنامه (همانند متغییر عددی با این تفاوت که در جلوی متغییر علامت $ باشد).

برنامه مساحت مستطیل

```
Cls
Input a , b
Let S= a * b
Print S
End
```

برنامه تبدیل Ngr (وزن یک جسم) به Kg و Gr

```
Cls
Input Ngr
Let Kg = Ngr \ 1000
Let Gr = Ngr Mod 1000
Print Kg , Gr
End
```

برنامه مجموع ارقام عدد طبیعی دو رقمی N

```
Cls
Input N
```

```
Let D2 = N \ 10
Let D1 = N Mod١٠
Let SD = D1 + D2
Print SD
End
```

برنامه چاپ نام ورودی

```
Cls
Input Name$
Print Name$
End
```

دستورات : QBASIC جلسه دوم

● IF در بیسیک چند نوع دستور شرطی از نوع IF داریم :

شرط : یک عبارت منطقی است که ارزش آن یا درست یا نادرست باشد . اگر شرط برقرار باشد (یعنی ارزش عبارت منطقی بعد از IF درست باشد) ؛ دستور(ات) بعد از Then اجرا می شود والا دستور بعد از خط Then ... Then ... اجرا می شود .

عبارات منطقی : در دستور شرطی بعد از کلمه IF یک عبارت منطقی است .که دارای دو ارزش درست یا نادرست است .

عبارات منطقی دو گونه است ۱- رابط های منطقی (2 AND - OR - NOT- عملگرهای رابطه ای (= , < , > , <> , =< , => , < >)

IF ... Then ... (در یک خط): صورت کلی آن دستور(ات) Then شرط IF

برنامه Max بین دو عدد

```
Cls
Input a , b
Let Max = a
If b < Max Then Let max = b
Print Max
```

```
End
```

IF بلوکی (بیش از یک خط) : صورت کلی آن Then شرط IF

```
دستور (۱)
دستور (۲)
... .... ...
دستور (...)
END IF
```

برنامه چاپ زوج (EVEN) یا فرد (ODD) ؛ عدد ورودی N

```
Cls
Input N
If N / 2 = Int ( N ( ۲ / Then
"Print N ; " Is EVEN

End
End IF
"Print N ; " Is ODD
End
```

IF ... Then ... ELSE (یک خط و بیش از یک خط) در صورتی که شرط برقرار باشد دستور(ات) بعد از Then اجرا می شود . والا دستور(ات) بعد از Else اجرا خواهد شد . اگر نتوان دستور فوق را دریک خط نوشت ؛ باید از IF بلوکی استفاده کرد.

IF شرط Then دستور(ات) Else دستور(ات)

IF شرط Then

```
دستور (۱)
دستور (۲)
... .... ...
دستور (...)
ELSE
دستور (۱)
دستور (۲)
```

... .

دستور (...)

END IF

در یک موسسه ؛ مالیات حقوق کارمندان را به قرار زیر محاسبه می شود .

الف : تا ۵۰۰۰۰ ریال معاف از مالیات

ب : از ۵۰۰۰۱ تا ۷۰۰۰۰ ریال چهار درصد مازاد ۵۰۰۰۰

ج : از ۷۰۰۰۱ به بالا ده درصد

```
Cls
Input W
If W ۵۰۰۰۰ = > Then
"Print " Tax = 0 Rial

End
End If
If W = 70000 Then
Let Tax = 4 * ( W – 50000 ) / 100
Else
Let Tax = 800 + 10 * ( W - 70000)
End If
Print " Tax = " ; Tax ; " Rials "

End
```