Quick Basic

*****

هذا الكتاب من اعداد احمد ابراهيم ، وهناك كتب اخرى كتبها الكاتب لتعلم لغات برمجة مختلفة منها (HTML CSS, JavaScript, jQuery, PHP, Java) يمكنكم تحميلها وقراءتها من هذا الرابط :
https://drive.google.com/open?id=0B2aI_a6mphOUQi1jdzlYSFhITWs
كل هذه الكتب مجانية اي يمكن قراءتها واعادة نشرها .

لاي سؤال او استفسار يمكنكم التواصل مع الكاتب عبر الحساب الشخصي في فيسبوك :
https://www.facebook.com/ah.ib.93

الفهرس

الفصل الاول

البداية مع لغة Quick Basic

الامر CLS

يستخدم هذا الامر لمسح الشاشة، حيث انه يحذف كل الاكواد البرمجية الموجودة قبله، وعادتا ما يكتب في اول البرنامج لمسح شاشة العرض من البرنامج السابق المكتوب.

الامر PRINT

يستخدم هذا الامر لاظهار النصوص على الشاشة، ويجب وضع النص بين علامتي تنصيص مزدوجة، ويمكن الاستغناء عن كتابة علامة التنصيص اذا كتبنا متغير او ارقام، مثال :

```
PRINT “Hello World”
PRINT 3*2
PRINT A$, D$; R$
PRINT
```

كتابة كلمة PRINT لوحدها ستطبع سطر فارغ

الامر END

يستخدم هذا الامر لإنهاء البرنامج وهو يكتب لوحده بدون أي اضافات، ويمكن عدم كتابته .

الفصل الثاني

المتغيرات والعمليات الرياضية

تستخدم المتغيرات لتخزين قيم والرجوع اليها او تغييرها في وقت لاحق، يمكن تسمية المتغير باي اسم لكن هناك بعض الشروط التي يجب الالتزام بها، 1- يجب ان يبدأ اسم المتغير بحرف وليس رقم 2- لا يمكن استخدام الرموز ما عدا الفاصلة السفلية _

هناك نوعين من المتغيرات

1- متغير نصي

وهو يحتوي على قيم نصية أي بين علامة تنصيص حتى وإن كانت ارقام ويجب ان ينتهي اسم المتغير النصي بعلامة الدولار $

2- متغير رقمي

وهو يحتوي على قيم عبارة عن ارقام ولا توضع بين علامتي تنصيص، وهناك عدة انواع من المتغيرات الرقمية :

النوع	القيمة	الاختصار
Integer	From -32764 To 32767	%
Long Integer	From -2147483648 To 2147483647	&
Float	From -3.37x10^38 To 3.37x10^38	!
Double	From -1.67x10^308 To 1.67x10^308	#
64-bit Integer	From -9223372036857775808 To Sam	&&
64-bit Float	From ±1.18E-4932 To ±1.18E+4932	##

- هذه الاختصارات توضع في نهاية اسم المتغير لتشير الى نوعه, وإذا لم نضع اي اختصار لتعريف نوع المتغير فإن البرنامج يفترض ان نوع المتغير هي Integer .

مثال :

```
x& = 55
```

مثال :

```
x$ = “Hello”
```

```
y$ = “ World”
PRINT x$ + y$ + “ !”
```

حيث ستكون النتيجة : Hello World ! ، فكما تلاحظ اضفنا متغيرات نصية ونصوص في جملة طباعة واحدة من خلال علامة + ، ويمكن كتابة الفارزة المنقوطة بدل علامة + وستكون نفس النتيجة .

مثال :

```
x = 5
y = 3
PRINT x * y
```

حيث سيطبع البرنامج النتيجة 15

لاجراء العمليات الرياضية على الارقام والمتغيرات الرقمية نستخدم الرموز التالية :

العملية الرياضية	الرمز
الجمع	+
الطرح	-
القسمة	/
الضرب	*
الجذر التربيعي لـ x	SQR(x)
الاس لـ x	Exp(x)
القيمة المطلقة لـ x	ABS(x)
X مرفوعة للاس 8	X ^ 8

هناك اولويات في العمليات الرياضية، مثلا القسمة والضرب تنفذ قبل الجمع والطرح، وهذا الجدول في الاسفل يوضح الاولوية في التنفيذ وهو مرتب بالتسلسل:

1	( )
2	^ , SQR , Exp , ABS
3	/ , *
4	+ , -

👆 الامر MOD

يعيد هذا الامر باقي القسمة، مثال :

```
X = 10 MOD 3
PRINT X
```

ستطبع النتيجة 1

الامر LET

يستخدم هذا الامر لتعريف المتغيرات حيث ان الاصدارات القديمة من لغة Qbasic كانت تتطلب اضافة هذا الامر قبل تعريف اي متغير لكن في الاصدارات الحديثة لم يعد يشترط كتابة هذا الامر، مثال :

```
LET x% = 7
```

الامر COMMON

يستخدم هذا الامر لجعل المتغير عام في كامل البرنامج، وتكتب المتغيرات العامة في اول سطر من البرنامج قبل كتابة اي كود برمجي

```
COMMON SHARED variablename
```

الامر const

يستخدم هذا الامر لتعريف المتغيرات وسيكون هذا المتغير ثابت ولا يتغير

```
CONST x = 34
```

دوال التحويل

تستخدم هذه الدوال لتحويل المتغيرات من نوع الى نوع اخر وكما في الجدول

FIX(X)	تعطي الجزء الصحيح من X
INT(X)	تعطي اكبر عدد صحيح اقل من او يساوي X
CINT(X)	يقرب X الى اقرب رقم صحيح INTEGER
CDBL(X)	تحول X الى النوع Double
CSING(X)	تحول X الى النوع
CLING(X)	تحول X الى النوع

مثال :

```
PRINT CINT(5.6)   'It will be 6
```

مثال :

```
x = 5.6
y = INT( x )
PRINT y
```

ستكون النتيجة 5

👆 الامر DIM

يستخدم هذا الامر لتعريف انواع المتغيرات بشكل ضمني، مثال :

```
DIM a AS STRING
DIM b AS INTEGER
DIM c AS LONG
DIM d AS SINGLE
DIM e AS DOUBLE
DIM f AS _INTEGER64 'QB64 only
DIM g AS _FLOAT 'QB64 only
```

ولاحظ انه لا نضع علامة الـ $ او غيرها بعد اسم المتغير، مثال :

```
X$ = "Hello"
DIM Y AS STRING
Y = "Goodbye"
PRINT X$
PRINT Y
```

الفصل الثالث

المصفوفات ARRAY

المصفوفة هي عبارة عن مجموعة من المتغيرات ويمكن كتابتها بهذا الشكل :

```
N$(0) = "Ted"
N$(1) = "Jack"
N$(2) = "Jill"
N$(3) = "Fred"
FOR I = 0 TO 3
PRINT N$(I)
NEXT I
```

لكن لاحظ ان هذا الشكل من المصفوفات لا يمكن ان يتجاوز عدد عناصرها 11 عنصر ولكتابة مصفوفات بعدد عناصر كبير نستخدم هذا الصيغة في تعريف المصفوفة :

```
DIM myArray(10) as TYPE
```

حيث ان myArray تمثل اسم المصفوفة والرقم 10 يمثل عدد عناصر المصفوفة و TYPE تمثل نوع المتغيرات في المصفوفة وبشكل افتراضي تكون نوع المتغيرات Integer ، مثال :

```
DIM A(20)
FOR I = 0 TO 20
A(I) = I * 2
NEXT I
FOR I = 0 TO 20
PRINT A(I)
NEXT I
```

مثال :

```
DIM x(5) as INTEGER
REDIM x(10) as INTEGER
```

حيث هنا في هذا المثال استخدمنا الامر REDIM لإعادة تعريف المصفوفة في وقت لاحق في البرنامج .
يمكن تعريف المصفوفة بهذا الشكل ايضا، مثال :

```
DIM table%(100)
```

❖ ويمكن تعريف مصفوفة متعددة الابعاد بهذا الشكل :

```
DIM housenames(25,25) as STRING
```

لاحظ انه لا يمكن استخدام الامر REDIM لإعادة تعريف المصفوفة متعددة الابعاد.

يبدأ ترتيب عناصر المصفوفة من 0 ولكن يمكن تغيير هذا الترتيب بالقيمة التي نريد وننهيها بالترتيب التي نريد

```
DIM deltas(-5 TO 5)
```

ويمكن تغير هذه القيم فيما بعد عن طريق الامر OPTION BASE

الامر TYPE

في بعض الاحيان نحتاج الى تعريف انواع مختلفة من البيانات وهذا الامر يساعدنا في ذلك، مثال :

```
TYPE Friend
   name AS STRING * 20
   phone AS STRING * 14
END TYPE
```

ويمكن استدعاء هذا النوع من المتغير بهذا الشكل :

```
DIM ArrayFriend AS Friend
ArrayFriend.name = "Ahmed"
ArrayFriend.phone = “964-7715916792”
PRINT ArrayFriend.name ; “:” ; ArrayFriend.phone
```

مثال لعمل قاعدة بيانات

```
TYPE FriendType
FullName AS STRING * 20
PhoneNumber AS STRING * 14
END TYPE
' The database
DIM Friends(2) AS FriendType
' Fill the database with names and numbers
Friends(0).FullName = "Joe Blow"
Friends(0).PhoneNumber = "1-310-555-1212"
Friends(1).FullName = "Jack Sprat"
Friends(1).PhoneNumber = "1-340-555-6545"
Friends(2).FullName = "Carol Christmas"
Friends(2).PhoneNumber = "1-350-555-2421"
' Print out the entire database
FOR I = 0 TO 2
```

```
PRINT Friends(I).FullName; ": "; Friends(I).PhoneNumber
NEXT I
```

الامر DATA ... READ

يستخدم هذا الامر لتخزين البيانات والوصول اليها بشكل اسرع، مثال :

```
DATA 10,24,31,15,67,34,87,92,14
FOR I = 0 TO 8
   READ A(I)
   PRINT A(I)
NEXT I
```

في هذا المثال سيتكون عندنا مصفوفة اسمها A وتخزن بها البيانات بهذا الشكل حيث القيمة الاولى ستكون 10 والثانية ستكون 24 وهكذا، واذا كان عندنا اكثر من امر DATA نستخدم الاسماء والامر RESTIRE للتمييز بينها، مثال :

```
Names:
DATA Fred, Joe, Jack, Sue
Values:
DATA 10, 24, 31, 15, 67, 34, 87, 92, 14
' Start with the DATA statement after "Values:"
RESTORE Values
FOR I = 0 TO 8
READ A(I)
NEXT I
' Start with the DATA statement after "Names:"
RESTORE Names
FOR I = 0 TO 8
READ N$(I)
NEXT I
```

الفصل الرابع

الشروط وحلقات التكرار

❖ تستخدم الرموز التالية للمقارنة بين القيم

>	اكبر من
>=	اكبر من او يساوي
<	اصغر من
<=	اصغر من او يساوي
<>	لا يساوي
=	يساوي

❖ لاستخدام اكثر من شرط نستخدم :

AND	
OR	
NOT	

IF

والصيغة العامة لها :

```
IF [condition] THEN
        [do this ]
END IF
```

مثال :

```
x = 5
IF x = 5 THEN PRINT 5
```

عند تنفيذ جملة IF في سطر واحد لا نضع END IF

IF … ELSE

```
IF [condition] THEN
        [do this ]
ELSE
        [do that]
END IF
```

مثال :

```
x = 5
IF x = 4 THEN
   PRINT 4
```

```
ELSEIF x = 5 THEN
   PRINT 5
END IF
```

SELECT CASE

تستخدم للتحقق من صحة شرط معين من بين عدة شروط، والصيغة العامة لها :

```
SELECT CASE <variable expression>
  CASE <value>
   [do this]
  CASE <value 2>
   [do instead]
  ...
  CASE ELSE
  ...
 END SELECT
```

حيث ستتم مقارنة <variable expression> مع <value> فاذا كانوا متساوين تنفذ ما بداخل CASE وتخرج وان لم تتطابق تنتقل للمقارنة CASE التالية واذا لم يتطابق مع اي واحدة سينفذ CASE ELSE، والـ CASE ELSE اختيارية أي يمكن عدم وضعها .

مثال :

```
x = 5
SELECT CASE x
   CASE 2
      PRINT 2
   CASE 4
      PRINT 4
   CASE ELSE
      PRINT "else"
END SELECT
```

FOR … NEXT

تستخدم لتكرار كود برمجي بعد معين من المرات، والصيغة العامة لها :

```
FOR <variable name> = <start value> TO <end value> STEP <increment>
```

```
 [do this]
NEXT
```

مثال :

```
FOR I = 1 TO 5 STEP 1
PRINT “Hello”
NEXT
```

مثال :

```
FOR X = 5 TO 1 STEP -1
PRINT “Hello”
NEXT X
```

WHILE...WEND

هي عبارة عن حلقة تكرار ستبقى تستمر في الدوران طالما ان الشرط متحقق (صحيح)، والصيغة العامة لها :

```
WHILE <condition is true>
  [do this]
WEND
```

مثال :

```
PRINT "Press any key to continue"
WHILE INKEY$=" "
WEND
```

DO...LOOP

وهي حلقة تكرار تنفذ طالما الشرط غير متحقق وستتوقف اذا تحقق الشرط، الصيغة العامة لها هي :

```
DO
  [and this]
LOOP UNTIL <condition is true>
```

او تكتب بطريقة اخرى :

```
DO
   [and this]
 LOOP WHILE <condition is true>
```

مثال :

```
DO
    x = x + 1
    PRINT x
LOOP UNTIL x >= 10
```

الفصل الخامس

الملفات الخارجية

👆 الامر OPEN

يستخدم هذا الامر لفتح ملف خارجي او انشاء الملف اذا لم يكن موجود، ويمكن الكتابة او قراءة هذا الملف وهو يأخذ ثلاثة بارامترات الاول يمثل اسم الملف ومسار تواجد الملف اذا كان في مجلد غير المجلد الموجود به البرنامج والثاني هو ما نود فعله هل الكتابة OUTPUT او القراءة INPUT والثالث تمثل رقم نضعه لتمييز ملف عن ملف اخر ومن خلال هذا الرقم يمكننا الوصول للملف لاحقا، الصيغة العامة له :

```
OPEN "(path)file name.ext" FOR {INPUT/OUTPUT} AS #n
```

مثال :

```
OPEN "file.txt" FOR OUTPUT AS #1
```

- ❖ يمكن وضع APPEND بدلا من OUTPUT وهي تؤدي نفس غرض الكتابة على الملف لكن الفارق هو ان APPEND يضيف النص المدخل الى نهاية النص في الملف المفتوح بينما OUTPUT تحذف النص القديم الموجود في الملف القديم وتضيف النص الجديد

👆 الامر WRITE

يستخدم هذا الامر للكتابة على ملف خارجي بعد فتحه بالامر OPEN وهو يحتاج الى بارامترين او اكثر الاول يمثل رقم الملف والاخرى تمثل النص المراد كتابته ويجب الفصل بينها بفارزة، مثال :

```
WRITE #1, “Hello”, “world”
```

- ❖ يمكن استخدام الامر PRINT بدلا من WRITE وهو سيؤدي نفس المهمة ولكن الفرق هو ان الامر WRITE سيكتب النص المدخل بين علامتي تنصيص على خلاف الامر PRINT الذي يكتب النص كما هو بدون علامات تنصيص، اضافة الى ان الامر PRINT يأخذ البارامتر الاول وهو رقم الملف وبعده فاصلة ثم البارامتر الثاني وهو يمثل ما سيخزن في الملف وبعده يمكن ان نفصل البارامترات الاخر (اذا كانت موجودة) بفارزة او فارزة منقوطة، مثال لتخزين وقت تشغيل واطفاء البرنامج :

```
OPEN "logfile.txt" FOR APPEND AS #1
PRINT #1, "Program Run: "; TIME$
CLOSE #1
CLS
INPUT "What is your name"; Name$
PRINT "Hello, "; Name$
OPEN "logfile.txt" FOR APPEND AS #1
PRINT #1, "Program Stopped: "; TIME$
```

```
CLOSE #1
```

👉 الامر INPUT

يستخدم هذا الامر لقراءة ملف خارجي بعد فتحه بالامر OPEN وهو ياخذ بارامترين الاول يمثل رقم الملف والثاني يمثل المتغير الذي ستخزن فيه القيمة، مثال :

```
OPEN "file.txt" FOR INPUT AS #1
INPUT #1, Name$
CLOSE #1
PRINT Name$
```

👉 الامر CLOSE

يستخدم هذا الامر بعد فتح ملف خارجي بالامر OPEN واجراء العمليات التي نريدها عليه لنغلق الملف بهذا الامر بعد الانتهاء وهو يأخذ باراميتر واحد يمثل رقم الملف المراد اغلاقه، مثال :

```
INPUT "Enter your name: ", Name$
OPEN "file.txt" FOR OUTPUT AS #1
WRITE #1, Name$
CLOSE #1
```

الفصل السادس

الرسومات والصوتيات

الامر COLOR

يستخدم لتلوين النص وتلوين خلفية النص، حيث يأخذ قيمتين رقميتين، القيمة الاولى تمثل لون النص والثانية وهي اختيارية تمثل لون خلفية النص، والجدول التالي يوضح كل لون والرقم المعبر عنه:

0	اسود	8	رمادي غامق
1	ازرق	9	ازرق فاتح
2	اخضر	10	اخضر فاتح
3	سمائي	11	سمائي فاتح
4	احمر	12	احمر فاتح
5	بنفسجي	13	بنفسجي فاتح
6	برتقالي	14	اصفر
7	رمادي فاتح	15	ابيض

مثال:

```
COLOR 9, 1
PRINT “Hello”
```

- لاحظ أن تأثير اللون سيبقى مطبق الى نهاية البرنامج او عند استخدام امر COLOR اخر وتغيير اللون او استخدام الامر CLS ومسح الشاشة.
- يمكن جعل النص يومض من خلال اضافة الرقم 16 الى اللون المراد عرض النص به مثلا لعرض نص بلون ازرق ويومض 17 = 16 + 1 او لعرض نص بلون برتقالي ويومض 22 = 16 + 6 ، ويسند الرقم الناتج الى الامر في القيمة الاولى والتي هي لون النص ولا يمكن تطبيق هذه الملاحظة على لون خلفية النص، مثال لعرض نص بلون احمر ويومض وبخلفية زرقاء :

```
COLOR 20, 1
PRINT “Hello”
```

الامر SCREEN

يستخدم هذا الامر مع الرسومات ويحدد حجم الشاشة وهو يأخذ احد هذه الارقام وهي :

0 وهذه هو نمط النصوص ولا يمكن استخدامه للرسومات وهو الحجم الافتراضي الذي يعمل به البرنامج

1 ويكون حجم الشاشة 320 x 200 , وهو يأخذ اربع الوان
2 ويكون حجم الشاشة 640 x 200 , وهو يأخذ لونين (ابيض و اسود)
7 ويكون حجم الشاشة 320 x 200 , وهو يأخذ ستة عشر لون
8 ويكون حجم الشاشة 640 x 200 , وهو يأخذ ستة عشر لون
9 ويكون حجم الشاشة 640 x 350 , وهو يأخذ ستة عشر لون
10 ويكون حجم الشاشة 640 x 350 , وهو يأخذ لونين (ابيض واسود)
11 ويكون حجم الشاشة 640 x 480 , وهو يأخذ لونين
12 ويكون حجم الشاشة 640 x 480 , وهو يأخذ ستة عشر لون
13 ويكون حجم الشاشة 320 x 200 , وهو يأخذ 256 لون

مثال :

```
SCREEN 7
```

👆 الامر PSET

يستخدم هذا الامر لرسم نقطة على الشاشة وهو يجب ان يستخدم مع الامر screen ، والصيغة العامة له :

```
PSET ([X coordinate],[Y coordinate]), [Pixel Colour]
```

مثال :

```
SCREEN 7
PSET (23, 66), 6
```

👆 الامر LINE

يستخدم هذا الامر لرسم خط، الصيغة العامة له هي :

```
LINE (x, y)-(x, y), color number
```

حيث ان اول x,y تمثل احداثيات نقطة البداية والـ x,y الثانية تمثل احداثيات نقطة النهاية و color number تمثل رقم لون الخط، ويجب استخدام الامر SCREEN قبل هذا الامر، مثال :

```
SCREEN 7
LINE (3, 3)-(55, 55), 6
```

اذا اضفنا الحرف B الى نهاية الامر LINE فإنه سيرسم صندوق حيث الاحداثيات الاولى تمثل النقطة العلوية اليمنى من الصندوق والإحداثيات الثانية تمثل النقطة اليسرى السفلى من الصندوق، مثال :

```
LINE (0, 0)-(320, 240), 15, B
```

وإذا اضفنا الحرفين BF بدلا من الحرف B سوف يرسم صندوق ولمنه سيكون ممتلئ باللون المختار

الامر CIRCLE

يستخدم هذا الامر لرسم دائرة وهو يجب ان يستخدم مع الامر SCREEN ، والصيغة العامة له هي :

```
CIRCLE (x, y), r, color number
```

حيث ان (x, y) تمثل احداثيات نقطة المركز للدائرة و r يمثل نصف قطر الدائرة و بعده نضع رقم اللون، مثال :

```
SCREEN 12
CIRCLE (320, 240), 100, 15
```

الامر PAINT

يستخدم هذا الامر لملئ شكل معين بلون، مثال :

```
SCREEN 12
CIRCLE (320, 240), 100, 15
PAINT (320, 240), 15, 15
```

الامر DRAW

يستخدم هذا الامر لرسم اشكال توضع ابعادها بين علامتي تنصيص هذا الامر ويسبق كل بعد حرف يمثل الاتجاه، كما موضح في الجدول :

U	اعلى	E	اعلى - يمين
D	اسفل	F	اسفل - يمين
L	يسار	G	اسفل - يسار
R	يمين	H	اعلى - يسار
C	لون الخط ويأتي بعد رقم اللون		
TA	تعني انحناء الزاوية وتأخذ رقم من 0 الى 360		

ويجب ان يسبق هذا الامر بالامر SCREEN ، مثال لرسم مربع بلون خط احمر

```
SCREEN 7
PSET (50, 50), 4
```

```
DRAW "u50 r50 d50 l50"
```

مثال لرسم مربع مليء باللون الابيض

```
SCREEN 12
DRAW "C15 D100 R100 U100 L100 BF1 P15,15"
```

حيث ان C15 تمثل اللون وهنا ابيض و BF1 تعني الانتقال الى المربع و P15,15 تعني ملء المربع بالون الابيض، ويجب ان يكون رقم اللون المسند الى C هو نفس الرقم مع P

مثال لرسم شكل جميل

```
SCREEN 12
FOR I = 0 TO 360 STEP 10
   DRAW "D100 R100 U100 L100 TA" + STR$(I)
NEXT I
```

👆 الامر WIDTH

يستخدم هذا الامر لاعطاء حجم للخط وللشاشة ويسند له رقم يمثل الحجم، مثال :

```
WIDTH 10
PRINT "Wow! This is big!"
```

👆 الامر BEEP

يستخدم هذا الامر لاصدار صوت من الجهاز، ويكتب هذا الامر فقط بدون اي اضافات اخرى

👆 الامر PLAY

يستخدم هذا الامر لاصدار نغمات صوتية، وتكتب النغمات على شكل حروف من A الى G وتضاف العلامات # او - او + مباشرتا بعد الحرف لاحظ هذا المثال :

```
PLAY "C C# A B+"
```

المسافة الفارغة بين الاحرف سيتم تجاهلها، ويمكن كتابة بعض الحروف الاخرى لاجراء بعض التاثيرات على النغمات، الاشارة > او < ستثخن النغمة وتخففها وهي تكتب قبل الحرف، الحرفين MB يكتبان اذا اردنا من البرنامج ان يستمر في العمل وفي نفس الوقت يصدر الصوت كخلفية واقصى حد هو 32 حرف، مثال :

```
PLAY “MB A <<<C D- >G”
PLAY "<d12d12d12g2>d2"
```

👉 الامر SOUND

يستخدم هذا الامر لاصدار صوت بتردد معين ولفترة زمنية معينة, حيث القيمة الاولى التي تسند له تمثل التردد ويجب ان يكون رقم بين 37 و 32767 والقيمة الثانية تمثل المدة الزمنية التي يستمر بها الصوت، مثال

```
SOUND 54, 86
```

الفصل السابع

الدوال

تستخدم الدوال عندما نحتاج الى كتابة نفس الكود برمجي البرمجي اكثر من مرة في داخل البرنامج، لذلك نكتب الكود في داخل الدالة ونستدعيها عندما نريد تنفيذ الكود، ويوجد نوعين :

Function -1

هذا النوع من الدوال يعيد لنا قيمة ويتم استدعاءها فقط بذكر اسمها، والصيغة العامة لها هي :

```
FUNCTION name ( parameters )
the code
END FUNCTION
```

بما ان هذا النوع من الدوال يعيد قيمة فإنها مثل المتغير بمعنى انه يجب ان نحدد نوع البيانات التي ستعيدها الدالة فإما ان تكون بيانات رقمية او نصية وهنا يجب ان نسبقها بعلامة الدولار

Subroutines -2

هذا النوع من الدوال لا يعيد اي قيمة ولكنه مجرد ينفذ الكود في داخله عند استدعائه، والصيغة العامة لها هي

```
SUB name ( parameters )
{ SHARED variables     'if any }
the code
{ STATIC variables      'if any, to be saved for use next time }
END SUB
```

- نستخدم العبارة SHARED اذا اردنا للمتغير ان يعرف فقط داخل الدالة ولكن اذا اردنا ان يستخدم المتغير حتى في خارج الدالة فإننا نستخدم العبارة STATIC

الامر GOSUB

يستخدم هذا الامر لجعل البرنامج يذهب الى الدالة SUBROTIN وينفذها ثم يعود الى السطر الذي يلي هذا الامر ويكمل، الصيغة العامة لها هي :

```
GOSUB [subroutine line number / label]
```

الفصل الثامن

اوامر منوعة

الامر INPUT

يستخدم هذا الامر لاخذ معلومات من المستخدم وتخزن هذه البيانات في متغير حيث يسند للامر INPUT قيمتين الاولى وتوضع بين علامتي تنصيص مزدوجة وهي تمثل السؤال الذي سيخرج في شاشة العرض والقيمة الثانية تمثل اسم المتغير الذي ستخزن فيه البيانات المدخلة من قبل المستخدم، ويفصل بين القيمتين فارزة منقوطة لتظهر علامة السؤال بعد النص المكتوب او يمكن وضع فارزة لكي لا تظهر علامة الاستفهام بعد النص، مثال :

```
INPUT “What is your name”; name$
INPUT “What is your age”, age
```

الامر INKEY$

يستخدم هذا الامر لمعرفة اي مفتاح ضغطه المستخدم من لوحة المفاتيح، حيث سيخزن في هذا الامر قيمة المفتاح، ولاحظ ان هناك اختلاف بين الحروف الكبيرة والصغيرة، وبامكان هذا الامر تخزين قيمة مفتاح واحدة واذا ضغط المستخدم مفتاح اخر سيحذف قيمة المفتاح القديم ويخزن قيمة المفتاح الجديد، وغالبا ما يستخدم عبارة تكرار مع هذا الامر، مثال :

```
DO
   LET k$ = INKEY$
LOOP UNTIL k$ <> ""
SELECT CASE k$
   CASE "q"
      PRINT "You Pressed Q"
END SELECT
```

- بعض المفاتيح لا يمكن لهذا الامر الوصول اليها بشكل مباشر، لذلك ستخزن قيمة المفتاح المدخل على شكل حرفين الاول عبارة عن رقم ASCII والثانية عبارة عن scancode .

الامر LOCATE

يستخدم هذا الامر لتحديد موضع المؤشر في شاشة العرض، حيث ياخذ قيمتين الاولى البعد العمودي والثانية البعد الافقي، والقيمة الثانية اختيارية اي يمكن عدم وضعها ليبدأ من بداية السطر، مثال :

```
LOCATE 14, 34
PRINT “Hello”
```

الامر ASC ( )

يستخدم هذا الامر لإعطاء رقم ASCII الخاص بالحرف المكتوب بين قوسيه وإذا كتب اكثر من حرف بين قوسيه سيعطي قيمة اول حرف فقط، مثال :

```
PRINT ASC("t")  'Will print 116
```

الامر CHR$( )

يمرر لهذا الامر رقم ASCII لحرف وهو يقوم بتحويله الى شكل مكتوب، مثال :

```
PRINT CHR$(34)
```

غالبا ما يستخدم هذا الامر لكتابة بعض الحروف الخاصة وهذا الجدول ويوضح ارقام لبعض الحروف

07	beep (same as BEEP)
08	Backspace
09	Tab
27	Esc
34	Double quote
72	Up Arrow
75	Left Arrow
77	Right Arrow
80	Down Arrow

الامر GOTO

يستخدم هذا الامر للذهاب الى سطر معين حيث نضع رقم في بداية السطر الذي نريد الذهاب اليه وعند هذا الامر نكتب ذلك الرقم، مثال :

```
4 PRINT “Hello”
GOTO 4
```

الامر DATE$

يستخدم لمعرفة التاريخ الان وتظهر النتيجة على هذا الترتيب mm-dd-yy ، مثال :

```
a$ = DATE$
```

الامر TIME$

يستخدم هذا الامر لإعادة الوقت الحالي وتظهر النتيجة بهذا الترتيب (ثانية : دقيقة : ساعة) ، مثال :

```
DO
   LOCATE 1, 1
   PRINT TIME$
   SLEEP 1
LOOP
```

الامر EXIT

يستخدم هذا الامر للخروج من حلقة التكرار او الدوال ويكتب بهذا الشكل حسب نوع الدالة او حلقة التكرار

EXIT DEF
للخروج من DEF FN function
EXIT DO
للخروج من حلقة DO
EXIT FOR
للخروج من حلقة FOR
EXIT FUNCTION
للخروج من الدالة FUNCTION
EXIT SUB
للخروج من الدالة SUBROTIN

الامر SLEEP

يعطل هذا الامر البرنامج لعدد من الثواني يسند له، مثال :

```
SLEEP 6
```

الامر STR$

يحول القيمة العددية الى نصية، مثال :

```
d$ = STR$(44)
```

الامر VAL

يحول القيمة النصية الى عددية، مثال :

```
A$ = “4+5”
B = VAL(A$)
```

الامر INSTR

يستخدم هذا الامر للبحث عن نص في داخل نص ويمرر له ثلاثة بارامترات الاول يمثل رقم الحرف الذي نريد ان يبدأ منه البحث وهذا البارامتر اختياري أي يمكن عدم وضعة وستكون قيمته الافتراضية واحد، البارامتر الثاني يمثل النص الذي سنبحث فيه، البارامتر الثالث يمثل النص الذي سنبحث عنه، مثال :

```
Pos = INSTR ("abcdefghi", "de")
```

سيعيد القيمة 4

الامر LEFT$

يستخدم هذا الامر لاخذ عدد من الاحرف من نص ويبدأ من اليسار، مثال :

```
A$ = “Hello”
B$ = LEFT$(A$, 2)
PRINT B$
```

سيطبع الحرفين He

الامر RIGHT$

يستخدم هذا الامر لاخذ عدد من الاحرف من نص ويبدأ من اليمين، مثال :

```
A$ = “Hello”
B$ = RIGHT$(A$, 2)
PRINT B$
```

سيطبع الحرفين lo

الامر MID$

يستخدم هذا الامر لاخذ عدد معين من الحروف من وسط نص، ويسند له ثلاثة بارامترات الاول يمثل النص المراد الوصول اليه والثاني يمثل رقم الحرف الذي سيبدأ منه العد والثالث يمثل عدد الاحرف، مثال :

```
A$ = "one two three"
B$ = MID$(A$, 5, 3)
PRINT B$
```

سيظهر الكلمة two ، ويمكن ايضا استبدال نص بنص اخر لاحظ هذا المثال :

```
A$ = "cabinet"
PRINT A$
MID$(A$, 4, 2) = "ar"
PRINT A$
```

هذا المثال سيطبع الكلمتين cabinet و cabaret

الامر LCASE$( )

يستخدم هذا الامر لتحويل النص من حالة الحروف الكبيرة الى صغيرة، مثال :

```
A$ = "Fly Away With Me"
PRINT LCASE$(A$)
```

الامر UCASE$( )

يستخدم هذا الامر لتحويل النص من حالة الحروف الصغيرة الى الكبيرة، مثال :

```
A$ = "Fly Away With Me"
PRINT UCASE$(A$)
```

الامر STRING$

يستخدم هذا الامر ليكرر لنا نص معين عدة مرات، وهو يأخذ بارامترين الاول هو عدد التكرار والثاني هو النص المراد تكراره، مثال لطباعة علامة النجمة 20 مرة :

```
PRINT STRING$(20, "*")
```

الامر SPACE$

يستخدم هذا الامر لطباعة الفراغ بعدد معين من المرات، مثال لطباعة 20 فراغ بين الحرفين :

```
PRINT "A"; SPACE$(20); "B"
```

الامر LEN( )

يستخدم هذا الامر لاعطاء طول النص (أي عدد حروفه بما فيها المسافات الفارغة) ، مثال :

```
A$ = "Hello"
PRINT LEN(A$)
```

سيطبع الرقم 5

الامر RAN

يستخدم هذا الامر لتوليد رقم عشوائي بين 0 و 1 وهو سيولد نفس الرقم اذا اعدنا تشغيل البرنامج ولحل هذه المشكلة نستخدم الامر RANDOMIZE TIMER ، مثال :

```
PRINT INT(RND * 10 + 1)
```

في هذا المثال سيتكون عندنا عدد صحيح عشوائي بين 0 و 10

الامر RANDOMIZE TIMER

يستخدم هذا الامر مع الامر RAN لتوليد ارقام عشوائية لان الامر RAN لوحده سيعيد تكرار نفس القيمة عند اعادة تشغيل البرنامج، مثال :

```
RANDOMIZE TIMER
PRINT RND
```

الفصل التاسع

ملاحظات عامة

❖ تستخدم الفارزة المنقوطة في نهاية السطر البرمجي لجعل السطر التالي من البرنامج يظهر في نفس السطر الاول، أي وكأن السطرين كتبوا في سطر واحد، مثال :

```
PRINT “Hello”;
PRINT “World”
```

سيعرض البرنامج بهذا الشكل :

Hello World

❖ تستخدم الفارزة لترك مسافة، وايضا تجلب االسطر اللاحق مع السطر المكتوبة فيه، مثال:

```
PRINT “Hello”, “World”
PRINT “Hello”,
PRINT “World”
```

سيعرض البرنامج بهذا الشكل :

Hello World

Hello World

❖ لكتابة تعليق يظهر فقط للمبرمج ولا يظهر في البرنامج عند التنفيذ نستخدم علامة التنصيص المفردة، مثال:

```
PRINT “Hi”  ‘ The Comment Here
```

او يمكن كتابة الكلمة المحجوزة REM وبعدها نكتب التعليق

❖ لغة كوك بيسك لا تفرق بين الحروف الكبيرة والصغيرة .

النـهايـة