الفصل الثالث

قواعـد اللغـة المتغيـرات والثوابـت
Constants & Variables

Programming Concepts في هذا الفصل

3

في هذا الدرس نبدأ في شرح قواعد لغات البرمجة التي نستعمل في كتابة الأوامر لتحقيق غرض المستخدم و ذلك بشرح النقاط التالية :-

- المتغيرات Variables و الثوابت Constants
- و أنواع البيانات Data Types
- طرق الإعلان عن المتغيرات
- دوال الإدخال و الإخراج Input/Output

فى لغة C#

لتوضيح بعض هذه الأنواع راجع الجدول التالى:

int	متغير يخزن رقم صحيح ولا يأخذ علامة عشرية
double	متغير يقبل رقم حقيقى يقبل العلامة العشرية
string	متغير يخزن عبارة حرفية
bool	متغير يأخذ احد قمتين True أو False
byte	متغير يأخذ Byte واحدة

والمتغيرات تأخذ أنواع كثيرة نعرضها كما ذكرتها ميكروسوفت فى الدليل الفنى للغة فى الجدول التالى:

C# Type	.NET Framework type
bool	System.Boolean
byte	System.Byte
sbyte	System.SByte
char	System.Char
decimal	System.Decimal
double	System.Double
float	System.Single
int	System.Int32
uint	System.UInt32
long	System.Int64
ulong	System.UInt64
object	System.Object
short	System.Int16
ushort	System.UInt16
string	System.String

الإعلان عن المتغيرات varible declaration

و يتم الإعلان عن المتغير بالصورة التالية :

```
int a;
```

و في هذا السطر يتم استعمال كلمة int في الإعلان عن متغير بالاسم a من نوع

INTEGER وكأنك تقول هناك متغير بالاسم a من النوع Integer.

وبنفس الأسلوب تستطيع الإعلان عن متغيرات من باقى الأنواع التى أشرنا إليها.

فى لغة Java

أنواع البيانات

تنقسم أنواع البيانات فى لغة جافا إلى فئتين رئيستين:

أ- بيانات مبنية ومعرفة فى أصل اللغة Primitive Data Types.

ب- بيانات يقوم المبرمج ببنائها User Defined Data Type.

والآن لنتعرف على النوع الأول:

أ- البيانات المعرفة فى أصل اللغة Primitive Data Types.

وكما قلنا فهى البيانات التى تأتى مبنيه فى اصل اللغة وهى تنقسم إلى أربعة أنواع:

1- البيانات الرقمية الصحيحة Integers مثل (155،10،5،-1،-1).

2- البيانات الرقمية ذات الفاصلة العشرية floating Point Integers مثل (1.33،0.55،....).

3- البيانات الحرفية مثل 'A','B','X' --- char.

4- الاختيار المنطقى (false , true) boolean.

ولكل نوع من الأنواع السابقة نطاق ومساحة تحتلها فى الذاكرة كما فى الجدول التالى:

خواص أنواع البيانات المبنية فى اللغة

النوع	الحجم الذى تحتله فى الذاكرة	النطاق
1- integer		
byte	8 Bits	-127 to 127 $(2^{-7}$ to $2^{7}-1)$
short	16 Bits	-32768 to 32767 $(2^{15}$ to $2^{15}-1)$
int	32 Bits	-2147483648 to 2147483647

long	64 Bits	$(2^{31}$ to $2^{31}-1)$ $=-9.2x10^{15}$ to $=9.2x10^{15}$ $(-2^{63}$ to $2^{63}-1)$
2- floating	Point numbers	
float	32 Bits	$=-3.4x10^{38}$ to $=3.4x10^{38}$
double	64 Bits	$=-1.8x10^{308}$ to $=1.8x10^{508}$
3- character		
char	16 Bit	0 to 65535
4- boolean		
boolean	1 Bit	*false or true*

النطاق والمساحة التى يحتلها اى متغير مذكور فى الجدول السابق لا تختلف باختلاف نظام التشغيل كما فى اللغات الأخرى

الإعلان عن المتغير Variable Declaration

يتم الإعلان عن المتغير بكتابة النوع ثم اسم المتغير مع إمكانية اعطائة قيم مبدأية initializing values .

ولتسمية متغير ما فلك حرية تسمية المتغير على ألا يبدأ الاسم برقم ولا يكون بين حروفه مسافة أو قوس أو الحروف الخاصة مثل (@و#و...) وفيما يلى أمثلة للإعلان عن المتغيرات.

```
int Salary = 500;
boolean male = false;
float age=35.6F;
char c='k';
```

وكما ترى فى الأمثلة السابقة لكى نقوم بالإعلان عن متغير مثل Salary قد سبقه النوع الذى ينتمي إليه int ثم اسم المتغير Salary أما قيمة المتغير فهى 500 وذلك عن طريق المؤثر "=" والمتغير Salary يحتل مساحة فى الذاكرة 32 بت لأنه من نوع int كما فى الجدول السابق وهكذا فلكى تستخدم متغير لابد وأن تحدد نوعه ثم تكتب نوعه ثم اسم المتغير.

فى Oracle PL/SQL

يتم الإعلان بكتابة اسم المتغير ثم النوع كما فى السطور

```
DECLARE
/* Declare variables which will be used in SQL statements */
v_EName         VARCHAR2(30) ;
v_EmpNo         Number := 7788;
BEGIN
/* Select Data from  Emp table. */
SELECT Ename
INTO  v_EName
FROM emp
WHERE EmpNo = v_EmpNo;
END;
```

فى هذه السطور

- فى السطر رقم 1 يبدأ البلوك بالكلمة المحجوزة DECLARE ومعناها إعلان وهى تسبق الإعلان عن المتغيرات المستعملة فى البرنامج.
- فى السطر رقم 3 و4 يتم الإعلان عن متغيرين الأول من النوع VARCHAR2 أى حرف والثانى من النوع Number مع اعطائه قيمة ابتدائية.
- فى السطر رقم 5 تبدأ العمليات داخل البلوك بالكلمة المحجوزة BEGIN وهى تسبق العمليات المطلوب إجرائها فى البلوك.
- فى السطور 7و8 يتم كتابة جملة الـ SELECT مع تعديلها بما يناسب الـ PL وفيها يتم احضار اسم الموظف ENAME من جدول الموظفين EMP ووضع هذا الاسم فى المتغير V_Ename بشرط أن يساوى كود الموظف EmpNo القيمة المكتوبة فى المتغير V_EmpNo.
- فى السطر رقم 11 يتم انهاء البلوك بالكلمة المحجوزة END;.
- وبالتالى ملخص البلوك هو:

كلمة DECLARE تسبق الإعلان عن المتغيرات التى يتم الإعلان عنها بين DECLARE و begin .

بين الكلمة المحجوزة BEGIN والكلمة المحجوزة END; يتم كتابة سطور العمليات.

أنواع البيانات الخاصة بقاعدة البيانات Database Data type

توجد مجموعة من أنواع البيانات التي يمكن استعمالها مع حقول قاعدة البيانات وهي:

Number (size [, precision])

ويستعمل لتخزين قيمة رقمية عدد الأرقام هي size وعدد الأرقام بعد العلامة (القوسين) هي precision.

CHAR (Size)

ويستعمل لتخزين حروف (مثل اسم الموظف) عدد هذه الحروف هي Size.

Varchar2 (size)

يستعمل مثل النوع CHAR ولكن الفرق أنه متغير الحجم أي يتم حجز مساحة مقدارها SIZE كحد أقصى ولكن المساحة الفعلية تقدر بعدد الحروف الحقيقية عكس النوع CHAR الذي يحجز عدد ثابت من الحروف بصرف النظر عن استعمال أم لا.

Date

تستعمل لتخزين قيمة تاريخية (تاريخ).

LONG

يستعمل لتخزين كمية كبيرة من النصوص TEXT تصل إلى 2 Gigabits.

LONG RAW

يستعمل لتخزين كمية كبيره من البيانات لحفظ في صورة BINARY ليست نصوص مثل ملفات الصور وغيره.

-RAW

تستعمل لتخزين كمية من البيانات في صورة BINAEY ولكن أقل من Long raw ويمكن استعمالها لتخزين صور وصوت وفيديو.

أنواع غير مرتبطة بقاعدة البيانات , NON Database Data type

هناك بعض أنواع البيانات غير مرتبطة بأنواع البيانات في قاعدة البيانات بل هي أنواع من البيانات تستعمل فقط مع أوامر البرمجة VHX]t ومنها ما يلي:

-DEC, DECIMAL, REAL, DOUBLE-PRECISIO

هي أنواع فرعية من النوع Number تستعمل داخل الـ PL/SQL.

-INTEGER, INT, SMALLINT, NATURAL, POSITIVE, NUMERIC

وهى بعض الأنواع الأخرى المشتقة أيضاً من النوع Number لتناسب بعض العمليات -

CHARACTER

وهو يقابل النوع CHAR فى أنواع البيانات الخاصة بقاعدة البيانات.

VARCHAR

وهو يقابل النوع VARCHAR2 فى متغيرات قاعدة البيانات.

BOOLEAN

ويستخدم لتخزين قيمة منطقية أما TRUE أو FOLSE.

TABLE

لتخزين مجموعة من القيم المقابل لمصفوفة ARRAY أى مجموعة بيانات من نفس النوع

RECORD

لتخزين مجموعة من القيم المختلفة مثل بيانات سجل فى جدول RECORD.

وسوف نتناول كيفية التعامل مع الانواع المختلفة فى الفصول التالية.

ولتوضيح ذلك تابع السطور التالية كما فى الشكل 15.

الإعلان عن متغيرات من نوع قواعد البيانات

النوع %TYPE

بالإضافة لأنواع البيانات التى عرضناها أضافت ORACLE فكرة جيدة لإلغاء الاختلاف بين نوع البيانات فى قاعدة البيانات ونوع البيانات للمتغير المستعمل فى الأوامر فأضافت ذلك لسهولة الإعلان عن هذه المتغيرات ومنها:

- الأمر TYPE % للإعلان عن متغير من نفس نوع حقل بدون أن تعرف.
- نوع الحقل ويتضح ذلك من السطر التالى:

```
S_SAL  EMP.SALARY % TYPE: = ;
```

فى هذا السطر يتم الإعلان عن متغير بالاسم S_SAL هذا المتغير يأخذ نفس نوع الحقل SALARY المعرف فى الجدول EMP وهذا يوفر عليك مجهود معرفة نوع بيانات الحقول وذلك بإضافة TYPE % للحقل EMP. SALARY ثم إعطائه قيمة ابتدائية.

و بالتالى عند الرغبة فى الإعلان عن المتغير فى السطر PL للتعامل مع قيمة حقل ولا تعرف مواصفات الحقل ليس عليك إلا إضافة %TYPE إلى اسم الحقل و الجدول لتحديد نوع المتغير.

النوع %ROWTYPE

يستخدم هذا النوع (الطريقة) لتعريف متغير من نوع سجل كامل يحتوى على نفس حقول سجل جدول قواعد البيانات بحيث يعبر المتغير عن السجل RECORD ويمكن التعامل مع حقوله، ويتم تعريف هذا المتغير بطريقتين :

الطريقة الأولى: DECLARE

```
MY_Employee EMP % ROWtype;
```

فى هذه السطور ثم تعريف متغير بالاسم my_employee.

هذا المتغير نوعه هو نفس تركيب سجل من الجدول EMP وذلك بإضافة RowType% إلى اسم الجدول EMP وبالتالى إذا كان الجدول يحتوى على الحقول EMPNO, name salary يصبح المتغير MY_employee له نفس التركيب أى نفس الحقول.

الطريقة الثانية:

وفيها يتم تكوين حقول المتغير بدون الإشارة إلى جدول موجود بالفعل وذلك كما فى السطور التالية:

```
   DECLARE
1: Type A_EMPREC IS RECORD (
2: MY_EMPID        EMP.EMPNO % TYPE,
3: MY_ENAME        EMP.ENAME % TYPE);
4: MY_EMPVAR       A-EMPREC;
BEGIN
```

فى هذه السطور:

- فى السطر رقم 1 تم استعمال الكلمة TYPE للإعلان عن نوع جديد غير الأنواع المعروفة ،هذا النوع أسمه A-EMPREC ثم بعد ذلك IS Record وتعنى أن المتغير عبارة عن سجل record.

- فى السطور رقم 2،3 يتم بناء تركيب النوع الجديد (السجل) وذلك بتحديد مجموعة من حقول الجدول EMP.
- فى السطر رقم 4 إعلان عن متغير بالاسم MY-EMPVAR من النوع الجديد الذى أنشأناه بالاسم A-EMPREC وبالتالى يأخذ المتغير نفس التركيب و النوع أى عدد ونوع حقوله.

التحويلات بين الأنواع

بالاضافة لامكانية الإعلان عن المتغيرات بالمستويات المختلفة واستعمالها توجد إمكانية التحويل من نوع إلى نوع إذا كنت بحاجة الى ذلك وتوجد اكثر من طريقة حسب اللغة.

فى لغة VB.NET

1. كلمات التحويل conversion keyword

توجد مجموعة ن الكلمات التى توفرها .NET Visual Basic للتحويل الى الأنواع المختلفة وهى تستعمل بسهولة كما فى الصورة التالية:

```
Dim K1 As Integer
Dim Q1 As Double
K1= 432
Q1= K1
Q1 = Math.Sqrt(Q1)
K1 = CInt(Q1)
```

- فى هذه السطور تم الإعلان عن متغيرين الأول من النوع Integer والثانى من النوع Double.
- ثم تم استدعاء الدالة Sqrt لحساب الجزر التربيعى لقيمة المتغير Q1.
- ثم تم وضع قيمة المتغير Q1 فى المتغير K1 ولكن بعد وضع المتغير Q1 كمعامل للكلمة CInt(Q1) التى تقوم بتحويله أولا إلى النوع Integer.

2. استعمال الفصيلة Convert Class

توجد فصيلة Class بالاسم Convert تحتوى على عدد من دوال التحويل التى يمكنك استعمالها لتحويل نوع إلى نوع أخر مع ملاحظة أننا سوف نشرح موضوع الفصائل classes فى الفصول 10و11و12.

وتستعمل دوال هذه الفصيلة class كما فى السطور التالية:

```
Dim d1 As Double
D1 = 23.15
Dim i1 As Integer
I1 = Convert.ToInt32(d1)
```

- فى هذه السطور تم الإعلان عن المتغير d1 من النوع Double ثم تم إعطاته القيمة 23.15 ثم تم الإعلان عن المتغير I1 من النوع Integer فى السطر الأخير تم تحويل قيمة المتغير d1 إلى النوع Integer باستدعاء الدالة ToInt32 من الفصيلة Convert ووضع النتيجة فى المتغير i1 فيأخذ القيمة 23

فى لغة C#

يجرى مل يجرى فى VB.NET مع تغيير الطريقة كما يلى:

1. كلمات التحويل conversion keyword

توجد مجموعة من الدوال المجمودة بفصائل الأنواع توفرها C# للتحويل الى الأنواع المختلفة وهى تستعمل بسهولة كما فى الصورة التالية:

```
Int k;
String q='144';
K=INT32.Parse(q)
```

فى هذه السطور تم الإعلان عن متغيرين الاول من النوع Integer والثانى من النوع string ثم تم وضع قيمة المتغير q فى المتغير K ولكن بعد وضع المتغير q كمعامل للدالة Parse التى تقوم بتحويله أولا إلى النوع Integer.

2. استعمال الفصيلة Convert Class

توجد فصيلة Class بالاسم Convert تحتوى على عدد من دوال التحويل التى يمكنك استعمالها لتحويل نوع إلى نوع أخر مع ملاحظة إننا سوف نشرح موضوع الفصائل classes فى الفصول 10و11و12.

وتستعمل دوال هذه الفصيلة class كما فى السطور التالية:

```
Double d1;
d1 = 23.15;
int i1;
I1 = Convert.ToInt32(d1)
```

فى هذه السطور تم الإعلان عن المتغير d1 من النوع Double ثم تم إعطائه القيمة 23.15 ثم تم الإعلان عن المتغير I1 من النوع Integer فى السطر الأخير تم تحويل قيمة المتغير d1 الى النوع Integer باستدعاء الدالة ToInt32 من الفصيلة Convert ووضع النتيجة فى المتغير i1 فيأخذ القيمة 23

فى لغة Java

يتم استعمال طريقتين من التحويل:

1. استعمال طريقة Casting

وهو اما تحويل تلقائى كما فى السطور التالية:

```
byte a=20;
short b;
int c;
long d;
float e;
double f;
b=a;
c=b;
d=c;
E=d;
F=e;
```

أو باستعمال أوامر التحويل.

2. استعمال دوال التحويل فى الفصائل

حيث تحتوى الفصائل classes على دوال للتحويل الى أنواعها مثل الفصيلة Integer التى تحتوى على الدالة ()ParseInt و هكذا

المؤثرات Operatros

المؤثر هى رمز خاص يستخدم فى العمليات الحسابية والمنطقية التى تجرى على المتغيرات.

فى لغة Java ولغات C-Like

أنواع المؤثرات Operators Types

توجد عدة أنواع من المؤثرات فى لغة جافا Java وهم

1. المؤثرات الحسابية Arithemetic Operator
2. المؤثرات المنطقية Logical Operator
3. المؤثرات العلاقية Relational Operator
4. المؤثرات على مستوى البت Bit-Wise Operator

أ- المؤثرات الحسابية

المؤثرات الحسابية وهى (*,+,/,%,++,--)

أمثلة على المؤثرات الحسابية

المؤثر (+)

```
int a = 10;
int b =20;
int Sum;
Sum = a + b; // Sum =30
```

المؤثر (-)

```
int salary = 500;
int deduction = 85;
int netsalary;
netasalary = salary – deduction;
```

المؤثر (*)

```
int Salary = 500;
int Bonus = 2* Salary;
```

المؤثر (/)

```
int Salary = 500;
int halfsal =500/2;
```

المؤثر (%) وهو باقى القسمة Modulo

```
int x = 10% 3 ;          // x = 1
```

المؤثر (++) مؤثر الزيادة

وهو يعمل على زيادة المتغير بقيمة 1 فمثلا كى نزيد المتغير Count بقيمة 1 كنا نقوم بعمل الآتى:

```
int Count;
Count = Count +1;
```

ونستطيع فعل ذلك بطريقة مختصرة عن طريق (++)

```
Count ++;      // doing the Same Count = Count t+1
```

وتنطبق هذه الطريقة أيضا على المؤثر (- -) الذى يعمل على نقصان المتغير بقيمة 1

أمثلة مختلفة على اختصارات المؤثرات

يقوم هذا المثال بالإعلان عن متغير من نوع int ثم يقوم بإضافة 5

```
int x=10;
x=x+5;
```

يمكننا فعل ذلك بطريقة مختصرة كالآتي

```
x+=5;
```

وينطبق ذلك على بقية المتغيرات

```
x=x*5;
x*=5;
x=x/3;
x/=3;
```

أسبقية التعامل مع المؤثرات Operator Precedence

والآن بعد أن تعرفنا على المؤثرات الحسابية نعالج معى نتعرف على أسبقية معالجة المؤثرات من خلال هذا المثال.

```
int answer;
answer = 3+5*3;
```

إذا كانت النتيجة 24 فأنت مخطئ.

لأن عزيزي القارئ فى هذا المثال ما هو ترتيب تنفيذ العمليات السابقة هل الجمع أم الضرب؟ الحل بسيط ويخضع للجدول التالى الذى يرتب تنفيذ معالجة العمليات.

دالة الإدخال INPUTBOX

تستخدم هذه الدالة لاستقبال قيمة من المستخدم و ذلك في مربع الرسالة حيث تأخذ الصورة العامة التالية :

```
M = INPUTBOX (MSG , DEFAULT , TIT )
InputBox(prompt[, title] [, default] [, xpos] [, ypos] [, helpfile, context])
```

في هذه الصورة تأخذ الدالة ثلاثة معاملات .

- الأول MSG و يعرض الرسالة المطلوبة.
- الثاني DEFAULT و يعرض قيمة افتراضية في مربع الاستقبال، إذا قبلها المستخدم تختار OK وإلا يغيرها أولاً.
- الثالث TIT و يعرض عنوان مربع الرسالة في شريط العنوان.

فإذا وافق المستخدم على هذه القيمة ينقر زر OK و إلا يكتب ما شاء ثم OK.

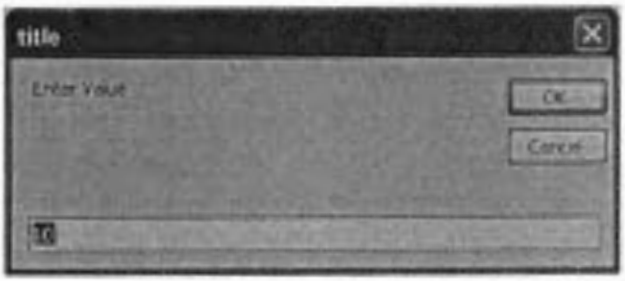
جرب استعمال هذه الدالة كما يلي :

1- ابدأ برنامج جديد وقع زر أوامر و غير عنوانه إلى CLICK.

2- اضغط زر الأمر مرتين و أكتب فيه ما يلي:

```
Dim M
M=INPUTBOX ( "Enter Value","title" ,10)
MSGBOX " YOU ENTERD                " &M
```

- في هذا السطر ثم أستدعاء الدالة INPUTBOX بمعاملاتها مع وضع القيمة التي أدخلها المستخدم في المتغير M ويظهر مربع الإدخال كما بالشكل التالي ثم عرض هذه القيمة بالدالة MSGBOX.



هذا بالاضافة للطريقة الطبيعية والمشهورة فى جميع اللغات الحديثة مثل .....VB.Net,C#,Java

وهى استعمال مربع النص Text Box فى ادخال المدخلات وكذلك إظهار النتائج.

فى لغة Java

هى بالطبع الجمل الأولية التى تستخدم للتدريب ولكن واقعيا يتم الإدخال والإخراج من برامج النوافذ Windows Applications أو صفحات مواقع الإنترنت.
ولكن هذه الدوال تستعمل فى بيئة نظام التشغيل DOS .

أوامر المخرجات

تستعمل الدالة println() فى طباعة المخرجات على الشاشة وبالطبع هى دالة Method بداخل فصيلة class بداخل حزمة Package وهذا ما سوف نتناوله فيما بعد بالتفصيل وبالطبع هى تطبع العبارات الحرفية String وتضيف اليها أى قيمة رقمية وتطبع الدالة print().

أوامر الادخال

تستخدم الدالة readLine() وتستخدم لاستقبال قيمة حرفية String ، ولاستعمالها لاستقبال قيم رقمية يتم استقبالها كقيم حرفية ثم تحويلها

بلوك الأوامر Block Statements

البلوك هو طريقة جمع مجموعة من سطور الأوامر معا للتعامل معها كوحدة وتسمى block.

فى لغات C-Like

يتم وضع مجموعه من الجمل او الأوامر محصورة بين الأقواس { } كما بالشكل:

```
{
    :
    :
    :
Statement 1
Statement 2
    :
    :
}
```

المؤثر	Description
( )	
+ +	Increment
- -	Decrement
*	Multi Placation
/	Division
%	Modulus
+	Addition
-	Subtraction
=	Assignment

ونتيجة تنفيذ المثال السابق هى 18 لأنه كما فى الجدول السابق فإن الضرب لـه أسبقية فى التنفيذ عن الجمع أي أن عمليات الضرب تتم معالجتها أولا.

وإذا أردنا المترجم أن يقوم بعملية الجمع أولا فى حالة المثال السابق نقوم بعمل الآتي

```
answer = (3+5) *3
```

وهنا لأن الأقواس لها أسبقية عن الضرب.

دوال الإدخال و الإخراج I/O Functions

كما هو مشهور في جميع لغات البرمجة توجد جمل (دوال) الإدخال و الإخراج I/O Functions التى تستعمل في استقبال قيم من المستخدم و كذلك عرض الرسائل و القيم له .

و بالرغم من اختلاف ذلك شيئا ما مع برمجة النوافذ إلا أنه توجد دوال كما يلي:

فى لغة VB.NET

دوال الإخراج (عرض الرسائل)

الدالة MSGBOX

و تأخذ الصورتين

```
1-MSGBOX MSG
2-MsgBox(prompt[, buttons] [, title] [, helpfile, context])
```

في الصورة الأولي :تم استعمال الدالة MSGBOX مع الرسالة المطلوب عرضها و هي محتوي المتغير MSG و هذه الصورة التي استعملناها في البرنامج السابق وهو صورة بسيطة يحدد فيها الرسالة المطلوب عرضها فقط مع الدالة ()MsgBox.

في الصورة الثانية :

يتم استعمال الدالة مع إرسال ثلاثة معاملات فا و هي :

- الأول : MSG و هو الرسالة المطلوب عرضها كما سبق .
- الثاني : رقم أو ثابت يعبر عن شكل مربع الرسالة و عدد الأزرار به هل يعرض مربع الرسالة زر أمر Button واحد بالعنوان Ok أم يعرض زرى أمر Ok,Cancle.
- الثالث : TITEL و يعبر عن العنوان الذي تريد أعطاءه لمربع الرسالة.
- المتغير K : عبارة عن قيمة أو ثابت يعبر عن اختيار المستخدم هل اختار زر الامر Ok أم اختار زر الامر Yes وهكذا.
- ويظهر ذلك من النص التالى:

```
1:Dim v1, tit, r
2:v1 = TextBox1.Text
3: TIT = TextBox2.Text
4:r = MsgBox("this Msg WITH NO ", Val(v1), tit)
5:MsgBox(" YOU SELECT NO " & R)
```

ق هذه السطور :

- في السطر رقم 1 و رقم 2 تم وضع قيم أدوات الكتابة في المتغيران TIT , V1وقد تم الإعلان عنها ق السطر الأول.
- في السطر رقم 3 يتم استدعاء الدالة MSGBOX مع إرسال معاملات هي الأول الرسالة "this Msg WITH NO " و الثاني القيمة V1 التي أدخلها المستخدم ق مربع النص TextBox1 والثالث المتغير TIT و هو عنوان مربع الرسالة الذى ادخله المستخدم ق مربع النص الثانى TextBox2.
- في السطر رقم 4 يتم عرض رسالة توضح الزر الذي اختاره المستخدم من مربع الرسالة و ذلك بعرض قيمة المتغير R الذي أخذ هذه القيمة في السطر رقم 3 من نتيجة استدعاء الدالة.

فى هذا الشكل يتم حصر جمل Statement 1 و Statement 2 بين الأقواس ليتعامل كبلوك.

ولكن السؤال ما الفائدة وراء إنشاء بلوك جمل ؟

الفائدة هى إمكانية الإعلان عن متغيرات داخل البلوك ليس لها علاقة بما قبل أو بعد البلوك مثل تعريف متغير داخل بلوك داخل دالة فهو خاص بها.

وكذلك إمكانية تنفيذ عملية مثل التكرار looping وعملية الشرط Condation على هذه المجموعة من الأوامر معا.

فى لغة VB.NET

يتم استعمال كلمات لإنهاء البلوكات ولكن حسب نوع البلوك.

فمثلا لإنهاء بلوك التكرار for يتم استعمال كلمة Next.

ولإنهاء بلوك الشرط If يتم استعمال الكلمة End If وسوف يتضح ذلك فى الفصول التالية.