الفصل الخامس عشر

الجمل الشرطية والقرارات

سنشرح في هذا الفصل استخدام عبارات التحكم في الإجراءات ونعنى بها تلك العبارات التى تساعد في تحديد الشروط واتخاذ القرارات في حالة وجود أكثر من بديل.

بالانتهاء من هذا الفصل ستكتسب المعارف وتتدرب على المهارات التى تجعلك قادرا على التعامل مع:

- عبارة الشرط ***IF***
- التركيب ***If...Then***
- التركيب ***If..Then...Else***
- عبارة ***If*** المتداخلة
- عبارة المقارنة ***Select Case***

يستخدم **Visual Basic** نوعان من عبارات الشرط هما:

- عبارة الشرط **IF** .
- عبارة المقارنة **Select Case**.

عبارة الشرط IF

دائماً ترتبط بتحقيق شرط معين، فإذا وقع الشرط صحيحاً يتم تنفيذ سطر أو مجموعة من السطور، وتستخدم بتركيبات عديدة نوضحها فيما يلى:

التركيب *IF...Then*

يستخدم لتنفيذ أمر واحد أو مجموعة أوامر فى حالة تحقق شرط معين. إذا كان المطلوب تنفيذ أمر واحد فى حالة وقوع الشرط صحيحاً فان التركيب يأخذ الصورة العامة التالية:

If <condition> Then <command>

وفى هذا التركيب يتم تقييم الشرط (**Condition**) الوارد بالتعليمة، فإذا كان صحيحا ينفذ **Visual Basic** الأمر **<Command>** الذي يلى كلمة **Then**، وإلا فانه يتجاهله،

ويصلح هذا التركيب عندما تريد تنفيذ أمر واحد فى حالة تحقق الشرط. والمثال التالى يوضح هذه الفكرة:

```
IF InAge > 80 Then MessageBox.Show( " You are too old")
```

فى هذا المثال الجملة الشرطية عبارة عن عبارة شرط هى IF. الشرط هو أن تكون قيمة InAge أكبر من ٨٠، ويرتبط جواب الشرط فى البداية بكلمة then ويتم إظهار الرسالة " You are too old" (جواب الشرط) فى حالة وقوع الشرط صحيحا (True).

لاحظ أن العبارة الشرطية تكتب بكاملها على نفس السطر لأن المطلوب تنفيذ أمر واحد فى حالة تحقق الشرط. فإذا أردت تنفيذ مجموعة من الأوامر إذا تحقق الشرط أى وقع صحيحا فان تركيب If...Then يأخذ الصورة العامة التالية:

```
If <condition> Then
        .......
        <commands>
        ........
End if
```

وكما تلاحظ فى هذه الصورة العامة أن الأوامر تكتب في السطر التالي لكلمة Then وأن التركيب ينتهى بعبارة End if

مثال

```
IF InAge>80 Then
        Beep
MessageBox.Show( "You are too old")
End if
```

التركيب IF...Then...Else

ويستخدم لتنفيذ مجموعة أوامر فى حالة تحقق شرط معين. أو مجموعة أخرى فى حالة عدم تحققه ويأخذ هذا التركيب الصورة العامة التالية:

```
If <condition> Then
        <commands>
Else
        <commands>
End if
```

وفى هذا التركيب يتم تقييم الشرط (Condition) الوارد به ، فإذا كان صحيحا ينفذ Visual Basic الأمر أو الأوامر (Commands) التى تلى كلمة Then ، والا فانه ينفذ الأمر أو الأوامر التى تلى كلمة Else، ويصلح هذا التركيب عندما تريد تنفيذ مجموعة

أوامر إذا وقع الشرط صحيحا ومجموعة أخرى إذا وقع الشرط خطئا.. والمثال التالى يوضح هذه الفكرة

```
IF InAge >= 80 Then
        Beep
        MessageBox.Show( "إنك لكبير السن ")
Else
        MessageBox.Show( "إن عمرك مناسب")
End If
```

فى المثال السابق تتم مقارنة محتويات المتغير InAge بالقيمة ٨٠ فإذا كانت InAge تساوي أو أكبر من ٨٠، تنفذ الأوامر التى تلى كلمة Then، أما إذا كان أقل من ٨٠، تنفذ الأوامر التى تلى كلمة Else.

التعامل مع الشروط غير المتحققة

تعاملنا فيما سبق مع الشروط التي تتحقق بتنفيذ أوامر معينة، ولكن ماذا عند الرغبة في تنفيذ أوامر عند *عدم تحقق* شروط معينة؟ . يتيح العامل المنطقي Not ذلك بأن نكتبه قبل الشرط المراد فإذا لم يتحقق تم تنفيذ الأوامر، مثلا:

```
If Not (stPasswd=txtPasswd) Then End
```

المثال السابق يقارن محتويات مربع نص بكلمة سر مخزنة فإن لم يتطابقا تم إنهاء البرنامج، كما ترى العامل Not هو الذي قام بعكس المعنى للتعامل مع عدم التحقق.

بصورة أعم يمكن التعامل مع كلا الحالتين أي التحقق وعدمه من خلال الصيغة الكاملة للعبارة الشرطية If و هي كالتالي:

```
If <condition> Then
        Statement1
Statement2      ≺condition=True هذا الجزء ينفذ في حالة التحقق
.....
Else
        Statement1
        Statement2  ≺condition=False ينفذ في حالة عدم تحقق الشرط
        .....
End If
```

هذه الصيغة يزيد عليها فقرة كاملة هي Else إلى End If وهي تنفذ في حالة عدم تحقق الشرط (condition=False) ، أي أن جزءاً واحداً من الجزئين ينفذ، الذي بعد Then أو الذي بعد Else بناءاً على قيمة condition هذا يوضح معنى التحكم الذي

تقوم به العبارة.

من الصيغة السابقة ندرك قيمة هذه العبارة. لنفترض أننا لا نملك وسيلة لتغيير خط سير البرنامج، على ذلك كان علينا إنشاء عدة برامج لعدة حالات، وكان على المستخدم تحديد البرنامج المناسب الأمر الذي يصبح عسيرا إن لم تكن الحالات واضحة للمستخدم أو كانت كثيرة. و لكن باستخدام عبارة التفرع المشروط أصبح بالإمكان أن نغير من خط سير البرنامج لكل حالة من المدخلات دون أن يضطر المستخدم لتشغيل عدة برامج.

عبارة If المتداخلة Nested If

العبارة الشرطية البسيطة سواء ذات السطر الواحد أو متعددة السطور تقوم باختبار شرط واحد ، ولكن إذا أردنا القيام بالعديد من الاختبارات كأن نقوم بمعرفة هل السن المدخل من المستخدم أكبر من ٢٥ مثلا ، ثم نختبر بعد ذلك هل هو أكبر من ٤٠ أو لا إن كان أكبر من ٢٥ وهكذا. بمعنى آخر هب أن لدينا برنامج يطلب من المستخدم إدخال عمره ثم يقوم بمعاملة المستخدم تبعا لفئته السنية أي يختبر كون العمر يقع في أي فئة ثم ينفذ الكود المناسب. أحد الصور لحل هذه المشكلة استخدام عبارات If المتداخلة كالتالي

```
If InAge>=25 Then
    If InAge>35 Then
        If InAge>65 Then
            MessageBox.Show( "إنك كبير السن")
        Else
            MessageBox.Show( "إن عمرك مناسب")
End if
    Else
        MessageBox.Show( "إنك لا تزال شاباً ")
    End if
Else
    MessageBox.Show( "لا زلت صغيرا جدا ")
End if
```

المثال السابق يتعامل مع فئات سنية (أقل من ٢٥، من ٢٥ حتى ٣٥ ، من ٣٥ حتى ٦٥ ، أكبر من ٦٥) ثم يقوم بإظهار رسالة عن عمر المستخدم. ولكن في المثال السابق كثير من العبارات الزائدة عن الحد المرهقة في كتابتها. لذا يتيح Visual Basic صورة أخرى لكتابة العبارات المتداخلة باستخدام ElseIf كالتالي:

```
If  InAge<25 Then
        MessageBox.Show( " لا زلت صغيرا جدا")
ElseIf  inAge<35 Then
        MessageBox.Show( " إنك لا تزال شابا")
ElseIf  inAge<65 Then
        MessageBox.Show( " إن عمرك مناسب")
Else
        MessageBox.Show( " إنك كبير السن")
End If
```

كما ترى سهلت العبارة السابقة وحسنت من شكل الكتابة ووضوح وظيفة الكود.

عند تكوين العبارات المنطقية المركبة يجب أن نعرف أولوية تنفيذ العوامل المختلفة حتى نضمن أن يتم تنفيذ التعبير كما ينبغي. أولوية تنفيذ العوامل تتم بالترتيب التالي:

١. العوامل الحسابية بترتيبها المعروف.
٢. عامل الوصل (للمتغيرات الحرفية) concatenation وهو & .
٣. العوامل العلائقية (Relational Operators) مثل > و< .
٤. العوامل المنطقية (Boolean operators) مثل Not و And .

استخدام عبارة Select Case

تصلح عبارة الشرط IF إذا كان جواب الشرط عبارة عن احتمالين أو ثلاثة. أما إذا كنت تتوقع عند تقييمك لشرط معين احتمالات كثيرة، فمن الأفضل أن تستخدم عبارة Select Case ، وتكون صيغتها العامة كما يلي:

```
Select Case <condVar>                  ≺ الشرط
        Case <value1>                  ≺ الاحتمال الأول
                Statement group 1
        Case <value2>                  ≺ الاحتمال الثاني
                Statement group 2
        ......
                ......
        Case Else
                Statement group n
End Select
```

حيث تبدأ العبارة بـ Select Case يليها اسم المتغير أو التعبير (expression) الذي سيتم اختباره و من الممكن أن يكون هذا المتغير أو التعبير من أي نمط عددي حتى الأعداد الحقيقية أي التي بها كسور أو متغير حرفي String .يوضح جدول ١٥-١ التالي أشهر أنماط المتغيرات المستخدمة مع عبارة Select Case.

جدول ١٥-١ أنماط المتغيرات المستخدمة مع عبارة Select Case.

نمط المتغير	مثال
علائقى	Case IS < = 25
تساوى	Case IS = 15.5
تساوى	Case 15.5
مدى	Case -5 To 5
متعدد	Case IS < 100 , IS >0
حرفى	Case "كمبيوتر"

بعد ذلك تأتي الاحتمالات (عبارات Case) بعد كل منها إحدى قيم المتغير الذي ستتم مقارنته ثم يعقبها التعليمات التي ستنفذ إذا كان الشرط صحيحا أو كان المتغير بهذه القيمة.

وأخيرا يأتي Case Else ومعناها إذا كان المتغير لا يساوي أيا من القيم السابقة أو إذا لم يقع الشرط صحيحا، فإن التعليمات التى تلي Else هي التى تنفذ.

مقطع واحد فقط من التعليمات في العبارة السابقة هو الذي سينفذ وهو الذي ينطبق على أول شرط صحيحا (أول Case تتحقق)، بعد ذلك ينتقل التنفيذ إلى آخر عبارة End Select، حتى ولو كان بعد هذه الحالة حالات أخرى متحققة فسوف يتجاهلها المترجم، علما بأنه ينبغي عليك التأكد من عدم وجود ذلك لأن هذا يعتبر أحد عيوب البرنامج الصورة السابقة للعبارة Select Case استخدمت قيمة واحدة بعد كل Case ، و لكن يمكن أيضا التعامل بصورة أكفأ باستخدام عدة قيم أو مدى من القيم أو استخدام المقارنات و التي تعطي عددا لا نهائيا من القيم. المثال التالي يستخدم ذلك في تحديد نسبة الخصم على المبيعات تبعا لعددها:

```
Select Case inQuantity
```

```
        Case Is < 0              ' استخدام المقارنة
            MessageBox.Show( "الكمية يجب ألا تكون سالبة")
            Exit Sub
        Case 1 , 2 , 3  ' استخدام قائمة
            SgDiscount=0
        Case 4  To 9             ' استخدام مدى من القيم
            SgDiscount=0.1       ' 10%
        Case 10 To 49
            SgDiscount=0.2       ' 20%
        Case Is > 15
            SgDiscount=0.3       ' 30%
End Select
```

لاحظ أننا استخدمنا في العبارة السابقة معامل علائقي في أول وآخر حالة (Case)، والقوائم في الحالة الثانية ،ومدىً أو نطاق من القيم في الحالة الثالثة والرابعة مما بسط من

كتابة العبارة كما هو واضح. القوائم تتكون من قيم مفصولة بفاصلة (،) و المدى يعرف بقيمتين بينهما (To) .

يستخدم المعامل IS غالبا مع المعاملات العلائقية مثل (< > <= >= =) . بالرجوع إلى المثال السابق تجد أننا استخدمنا المعامل IS في أول حالة لتحديد هل المتغير أقل من القيمة صفر، وهل هو أكبر من القيمة ١٥ في آخر حالة. في حالة استخدام Case مع القوائم ومدىً من القيم لم نستخدم العامل IS .

علاوةً على ما سبق، فإن القوائم لا يشترط أن تكون قيم منفصلة بل يمكن أن تكون مدى من القيم أو مقارنة مثل:

```
Case 1 To 4, 7 To 9, 11, 13, Is >23
```

اتفاقية القرص المدمج

نود التنبيه إلي أنك بمجرد فتح المغلف الذي يحتوي علي القرص المدمج المرفق بالغلاف الداخلي لهذا الكتاب،تكون موافقاً وملتزما بتنفيذ اتفاقية ضمنية بيننا وبينك تتضمن الآتى:

- لا يجوز نسخ محتويات القرص المدمج أو إعادة طباعتها ونشرها أو توزيعها بأي صورة لأن محتويات الكتاب والقرص المدمج ملك لك وحدك، والقرص المدمج (CD_ROM) جزء لا يتجزأ من الكتاب الذي دفعت ثمنه وينطبق عليها ما ينطبق علي الكتاب من حفظ حقوق الطبع والقوانين التي تضمنها. ومع ذلك يسمح لك بنسخ بعض المعلومات التي تحتاجها أنت لعملك وليس لغيرك.
- لقد بذلت قصارى جهدي للتأكد من صحة المعلومات الواردة بهذا الكتاب، و CD_ROM واختيار البرامج. إلا أنني لا أعتبر نفسي مسئولاً بأي شكل صريحا أو ضمنيا عن أي نتائج تترتب علي استخدام المعلومات الواردة بهذا الكتاب و CD_ROM المصاحبة له أو أي تعديلات يجريها القارئ عليها.
- نحن غير مسئولين (الناشر والموزع والبائع) عن التلف الذي يلحق بالقرص المرفق نتيجة لسوء استخدامك له.