الفصل السابع
الحلقات التكرارية

الفصل السابع

الحلقات التكرارية

الحلقات التكرارية من الموضوعات الهامة والأساسية في لغات البرمجة وذلك لم يمكن أن تتيحه لك من تحكم في مسار تنفيذ الأوامر داخل البرنامج ، كذلك فإن هذه الحلقات يمكن لك من خلالها اختصار العشرات من السطور داخل الكود ، ويمكن تقسيم هذه الحلقات إلى نوعين أساسيين :

1- التكرار المحدد .

2- التكرار المشروط .

وفيما يلى سنقوم باستعراض كل نوع منهما على حدى مع والصور المختلفة له مع إيضاح ذلك بالأمثلة .

التكرار المحدد :

المقصود بالتكرار المحدد هو تكرار مجموعة من الأوامر داخل الكود لعدد محدد من المرات تقوم بتحديده مسبقاً وذلك عن طريق الأمر For ... Next والصيغة القياسية لإستخدام هذا الأمر كما يلى :

```
For Variable = Start To End
Code
Next Variable
```

ويمكنك التعرف على المعاملات الخاصة بالصيغة السابقة من خلال التالي :

For : الأمر الخاص ببداية التكرار .

Variable : هذا المعامل هو عبارة عن متغير يخزن بداخله القيمة الخاصة بعدد مرات التكرار مع ملاحظة أن هذا المتغير يكون من النوع الرقمى .

Start : القيمة التي يبدأ منها التكرار .

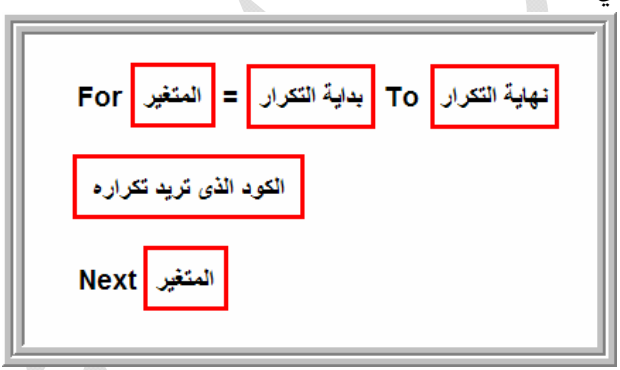
To : أمر تحديد أقصى قيمة للتكرار .

End : القيمة التي ينتهى عندها التكرار .

Code : الكود الذي سيتم تكراره .

Next Variable : الأمر الخاص بالذهاب للمتغير لإعادة تكرار الكود فإذا كانت قيمة المتغير وصلت إلى أقصى قيمة (قيمة النهاية التي قمنا بتحديدها) فسيتم الخروج من هذه الحلقة .

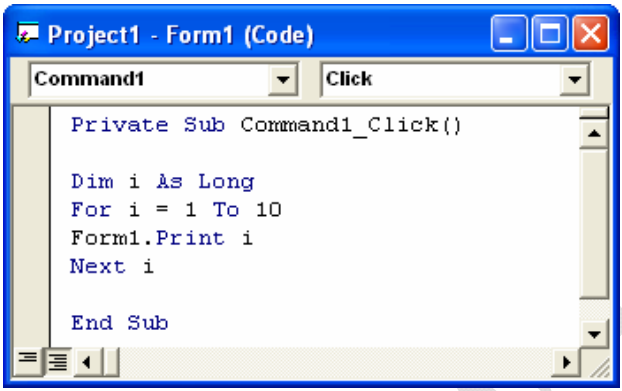
ومما سبق يمكن إيضاح عمل هذه الحلقة التكرارية والصيغة القياسة لها من خلال الشكل التالي :



ولتوضيح طريقة عمل هذه الحلقة بصورة عملية تعالى قم بفتح مشروع جديد ثم قم بإضافة مفتاح CommandButton إلى الفورم مع تغيير خاصية Caption له إلى Print ثم قم بإضافة الكود التالي داخل الحدث Click لهذا المفتاح .

```
Dim i As Long
For i = 1 To 10
Form1.Print i
Next i
```

لتصبح نافذة الكود كما بالشكل التالي :



شرح الكود :

في السطر الأول قمنا بتعريف متغير من النوع Long يتم استخدامه كعداد أو مخزن لعدد مرات تكرار الكود .

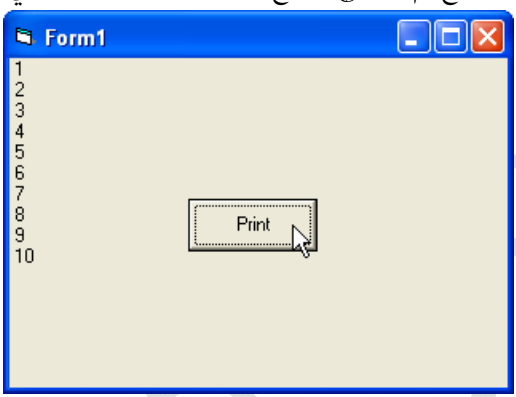
في السطر الثاني نبدأ عمل الحلقة التكرارية عن طريق الأمر For مع المتغير I مع تحديد قيمة البداية بـ (1) وقيمة النهاية بـ (10) أي أن عملية التكرار ستتم عشر مرات تبدأ بالرقم واحد وتنتهي بالرقم عشرة .

السطر الثالث هو عبارة عن الكود الذي سيتم تكراره وفي هذا الكود قمنا **باستخدام** الأمر Print لنقوم بطباعة قيمة المتغير I على الفورم .

في السطر الرابع نقوم بإعادة تكرارالكود مرة أخرى عن طريق الأمر Next I وبذلك سيتم طباعة قيمة المتغير I على الفورم أول مرة 1 وهي قيمة البداية التي قمنا بتحديدها ثم يعاد تكرار الكود مرة أخرى فتصبح قيمة المتغير I هي 2 فيتم طباعة الرقم 2 وهكذا إلى أن تصل قيمة المتغير I إلى عشرة (10) فيتم الخروج من الحلقة

وذلك لان الرقم عشرة (10) هي قيمة النهاية التي قمنا بتحديدها لعملية تكرار الكود .

قم الآن بتشغيل البرنامج ثم أنقر مفتاح Print كما بالشكل التالي :

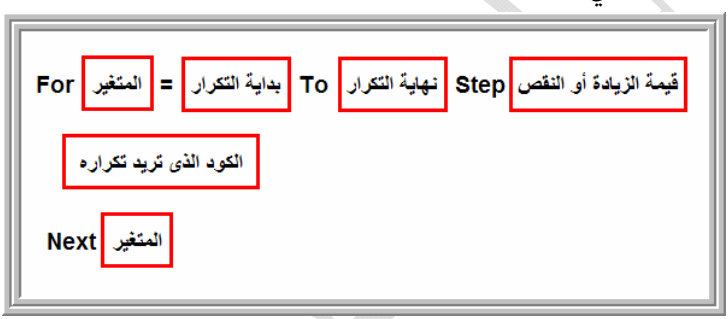


لاحظ أنه تم طباعة قيمة المتغير I على الفورم وبنفس عدد المرات التي قمنا بتحديدها داخل الكود (10 مرات) ، ولكن ماذا إذا أردنا تغيير القيمة التي يتم بها تكرار الكود السابق لتصبح مزدوجة ؟ بمعنى أن لا يتم التكرار بطريقة تسلسلية كما بالمثال السابق (1 2 3 ...) ولكن يكون التكرار بطريقة زوجية (2 4 6 ...) ، وفي الواقع أن القيمة الافتراضية للزيادة داخل الحلقة التكرارية هي واحد (1) أما إذا أردت تغيير هذه القيمة فسنحتاج إلى إضافة الأمر Step الذي يمكن من خلاله تحديد مقدار الزيادة أو النقص ، والمقصود بالنقص هنا أن بإمكانك أن تجعل الأمر For ... Next يقوم بعكس عملية التكرار أو العد وذلك بأن يبدأ من القيمة الأكبر إلى القيمة الأصغر ولكن لاحظ أنك إذا أردت القيام بذلك لا بد وأن تكون القيمة التي ستأتي بعد الأمر Step هي قيمة سالبة وهذا كله سيتضح لك مع الأمثلة فلا تنزعك إذا لم تستوعب هذا الجزء جيداً فالأمر أبسط من ذلك بكثير ولكن تعالى نتعرف أولا على

الصيغة العامة التي ستصبح عليها الحلقة التكرارية بعد إضافة الأمر Step اليها من خلال ما يلى :

```
For Variable = Start To End Step Value
Code
Next Variable
```

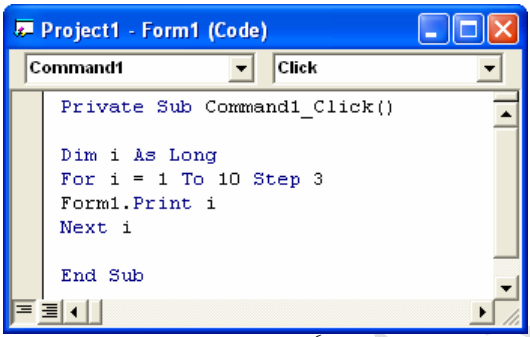
وبعد إضافة الامر Step يمكن إيضاح الصيغة القياسية لهذه الحلقة التكرارية من خلال الشكل التالي :



تعالى معى الآن نقوم بتعديل قيمة الزيادة في المثال الذي تناولناه سابقاً لتصبح ثلاثة (3) وذلك **باستخدام** الأمر Step كما بالكود التالي .

```
Dim i As Long
For i = 1 To 10 Step 3
Form1.Print i
Next i
```

ولاحظ أن نافذة الكود ستصبح كما بالشكل التالي :

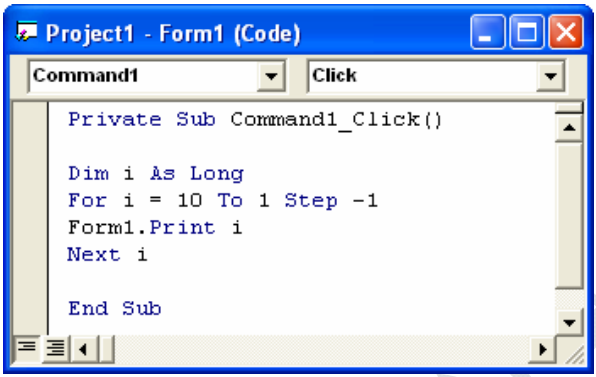


وهذه المرة عند تشغيل البرنامج ونقر مفتاح Print سيتم ستظهر لك الأرقام على الفورم كما بالشكل التالي :

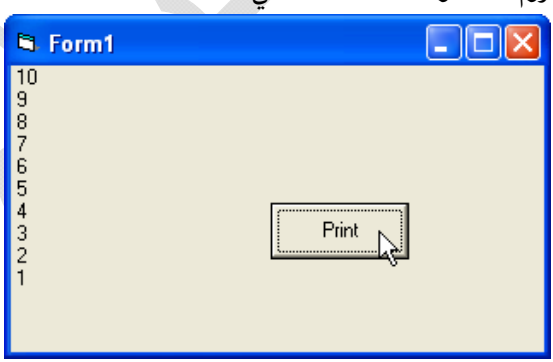
كما ترى أنه تم طباعة الأرقام بزيادة ثلاثة (3) بين كل رقم وأخر ولكن تعالى نقوم بتعديل هذا المثال مرة أخرى لكى نقوم بطباعة الأرقام من واحد إلى عشرة (1 : 10) ولكن بطرقة عكسية بحيث تظهر على الشكل (10 9 8 7 ...) وذلك من خلال الكود التالي :

```
Dim i As Long
For i = 10 To 1 Step -1
Form1.Print i
Next i
```

ولاحظ أن نافذة الكود ستصبح كما بالشكل التالي :



لاحظ ... في الكود السابق أننا قمنا بتحديد الرقم الأول الذي سيبدأ منه التكرار بالرقم عشرة (10) والرقم الذي ينتهي به التكرار بالرقم واحد (1) كما قمنا بتحديد مقدار الزيادة بسالب واحد (-1) لكي يتم التكرار بطريقة تنازلية أي أن في كل مرة يتم التكرار يتم إنقاص واحد ، وعند تشغيل البرنامج ونقر مفتاح Print ستظهر الأرقام على الفورم هذه المرة كما بالشكل التالي :



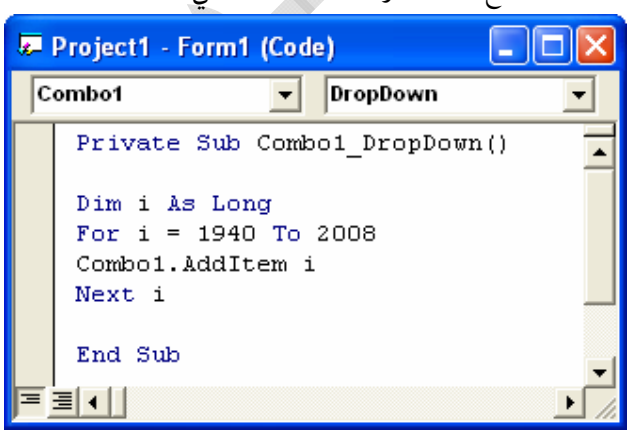
ما رأيك الآن أن نقوم بعمل مثال أخر أكثر واقعية يتضح لك من خلاله مدى أهمية هذه الحلقة التكرارية ، فتخيل معى أنك تريد تصميم نموذج يتم من خلاله إدخال البيانات الشخصية لموظفى احدى الشركات مثل الأسم والعنوان وتاريخ الميلاد بحيث أن يتم

اختيار سنة الميلاد من خلال ComboBox وسنفرض أن الأعوام التي على المستخدم الإختيار منها من داخل ComboBox هي من عام 1940 إلى 2008 فكيف سنقوم بذلك ؟

لاحظ ... أنك إذا قمت باضافة تلك العناصر لأداة ComboBox بالطريقة العادية فستحتاج إلى كتابة ما يقرب من سبعين سطراً أما إذا قمت **باستخدام** الحلقة التكرارية For ... Next في هذا الكود فكل ما ستحتاج إليه هو أربعة أسطر فقط مهما كان عدد العناصر التي تريد اضافتها كما يلى :

```
Dim i As Long
For i = 1940 To 2008
Combo1.AddItem i
Next i
```

ولاحظ أننا سنقوم بكتابة هذا الكود داخل الحدث DropDown لأداة ComboBox لتصبح نافذة الكود كما بالشكل التالي :



شرح الكود :

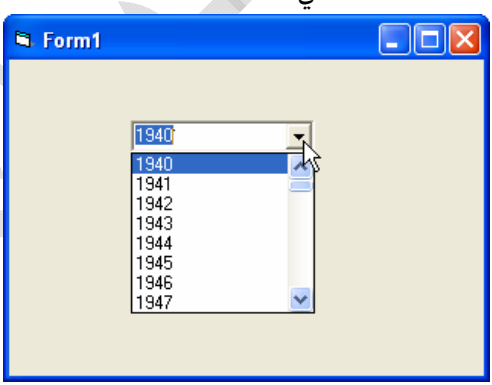
في السطر الأول قمنا بتعريف متغير من النوع Long يتم استخدامه كعداد أو مخزن لعدد مرات تكرار الكود .

في السطر الثاني نبدأ عمل الحلقة التكرارية عن طريق الأمر For مع المتغير I مع تحديد قيمة البداية بـ (1940) وقيمة النهاية بـ (2008) .

في السطر الثالث نقوم بإضافة عنصر إلى أداة ComboBox وهذا العنصر سيكون هو قيمة المتغير I والتى تبدأ بـ (1940) .

في السطر الرابع نقوم بإعادة تكرار الكود مرة أخرى وبذلك ومع تكرار الكود في كل مرة سيتم إضافة قيمة المتغير I الجديدة إلى عناصر أداة ComboBox إلى أن تصل قيمة المتغير I إلى أقصى قيمة (2008) فيتم الخروج من الحلقة التكرارية .

بعد ذلك قم بتشغيل البرنامج ثم أنقر فوق سهم القائمة ComboBox لتظهر لك عناصرها كما بالشكل التالي :



التكرار المشروط :

التكرار المشروط يتم من خلاله تكرار مجموعة من الأوامر داخل الكود لحين تحقق شرط ما ، وكذلك يمكن من خلال التكرار المشروط هذا تكرار مجموعة من الأوامر داخل الكود لحين انتفاء شرط ما وهذا يكون خلال الحلقة التكرارية Do While ... Loop والحلقة التكرارية Do Until ... Loop مع ملاحظة أن كل منهما يمكن استخدامها بأكثر من صورة نستعرضها فيما يلى .

الحلقة التكرارية Do While ... Loop :

الصورة الأولى للحلقة التكرارية Do While تكون على الشكل التالي :

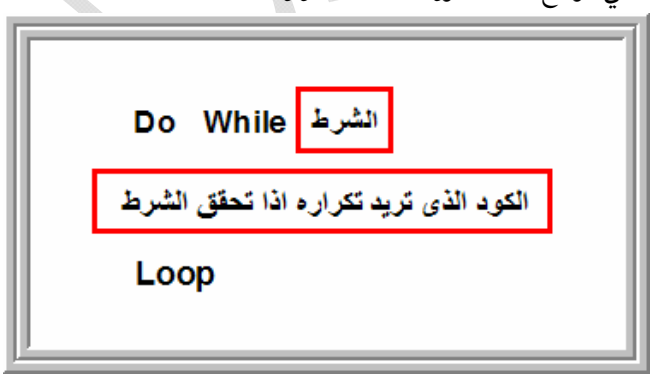
```
Do While condition
Code
Loop
```

حيث يمكن التعرف على المعاملات السابقة من خلال ما يلى :

Condition : الشرط الذي سيتم اختبار تحققه .

Code : الكود الذي سيتم تكراره إذا تحقق الشرط .

والشكل التالي يوضح تلك الصورة للحلقة التكرارية .



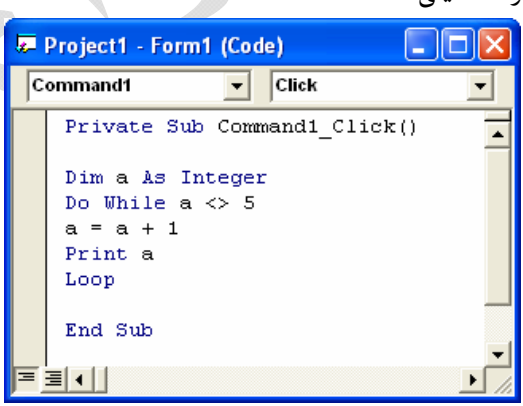
حيث أن في هذه الصورة سيتم التأكد من تحقق الشرط قبل الدخول في الحلقة التكرارية فاذا كان الشرط غير متحقق سيتم تجاوز الحلقة التكرارية وتجاهل الكود الموجود بداخلها ، أما إذا كان الشرط متحقق فسيتم تنفيذ الكود الموجود داخل الحلقة التكرارية ثم تعاود الحلقة اختبار الشرط مرة أخرى فإذا كان الشرط ما زال متحققاً فسيتم تنفيذ الكود مرة أخرى وهكذا إلى أن يصبح الشرط غير متحقق فيتم الخروج من الحلقة التكرارية .

مثال :

قم بفتح مشروع جديد ثم قم بإضافة مفتاح CommandButton إلى الفورم مع تغيير خاصية Caption إلى موافق ثم قم بكتابة الكود التالي للحدث Click لهذا المفتاح :

```
Dim a As Integer
Do While a <> 5
a = a + 1
Print a
Loop
```

لتصبح نافذة الكود كما يلى :



شرح الكود :

في السطر الأول قمنا بتعريف متغير بأسم a من النوع Integer .

في السطر الثاني نقوم **باستخدام** الأمر Do While لنبدأ الحلقة التكرارية باختبار قيمة المتغير a فإذا كانت قيمة هذا المتغير لا تساوى خمسة سيتم تنفيذ الكود الموجود في السطر الثالث والسطر الرابع وهو إضافة واحد إلى قيمة المتغير a ثم طباعة قيمة المتغير عن طريق الكلمة Print وهكذا إلى أن تصبح قيمة المتغير a تساوى خمسة فيتم الخروج من الحلقة التكرارية ، وعند تشغيل البرنامج ثم النقر فوق مفتاح موافق ستكون النتيجة كما بالشكل التالي :

الصورة الثانية للحلقة التكرارية Do While تكون على الشكل التالي :

```
Do
Code
Loop While condition
```

والشكل التالي يوضح تلك الصورة للحلقة التكرارية .

في هذه الصورة سيتم تنفيذ الكود الموجود داخل الحلقة التكرارية ثم يتم اختبار الشرط فإذا كان الشرط متحققاً يعاد تنفيذ الكود مرة أخرى وهكذا إلى أن يصبح الشرط غير متحقق فيتم الخروج من الحلقة التكرارية .

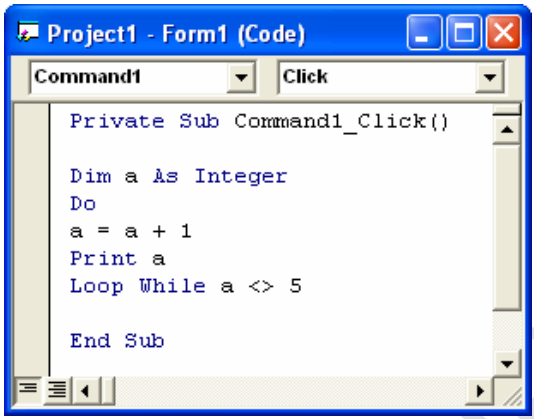
لاحظ ... أن الفرق بين هذه الصورة والصورة السابقة أن في هذه الصورة يتم تنفيذ الكود قبل التأكد من تحقق الشرط وبذلك فإن من خلال هذه الصورة سيتم تنفيذ الكود مرة واحدة على الأقل حتى إذا كان الشرط غير متحقق .

مثال :

سنقوم بتعديل المثال السابق ليصبح كما يلى :

```
Dim a As Integer
Do
a = a + 1
Print a
Loop While a <> 5
```

وتصبح نافذة الكود كما يلى :

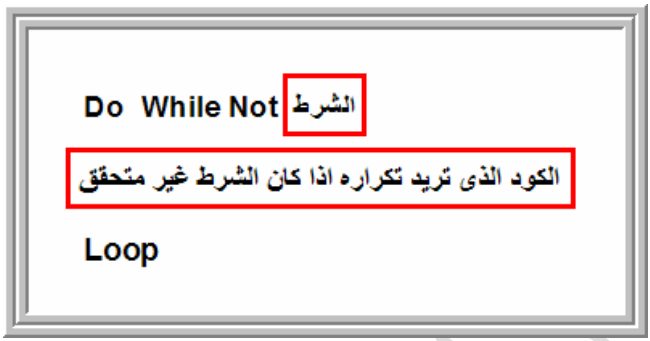


لاحظ أن النتيجة في هذه المرة ستصبح أيضاً كالتالى :

الصورة الثالثة للحلقة التكرارية Do While تكون على الشكل التالي :

```
Do While Not condition
Code
Loop
```

والشكل التالي يوضح تلك الصورة للحلقة التكرارية .



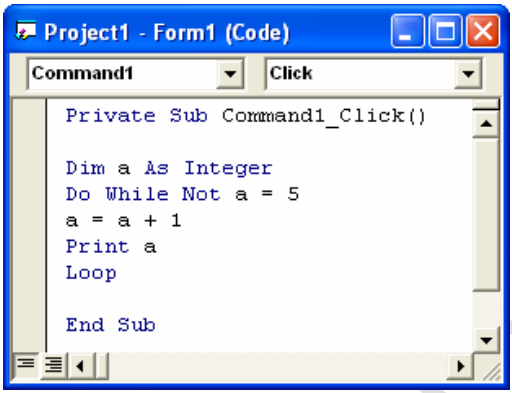
لاحظ أن في هذه الصورة تم إضافة الأمر Not لنفى الشرط وبذلك سيتم التأكد من عدم تحقق الشرط قبل الدخول في الحلقة التكرارية فاذا كان الشرط متحقق سيتم تجاوز الحلقة وتجاهل الكود الموجود بداخلها أما إذا كان الشرط غير متحقق فسيتم تنفيذ الكود الموجود داخل الحلقة التكرارية ثم تعاود الحلقة اختبار الشرط مرة أخرى فإذا كان الشرط ما زال غير متحقق فسيتم تنفيذ الكود مرة أخرى وهكذا إلى أن يصبح الشرط متحقق فيتم الخروج من الحلقة التكرارية .

مثال :

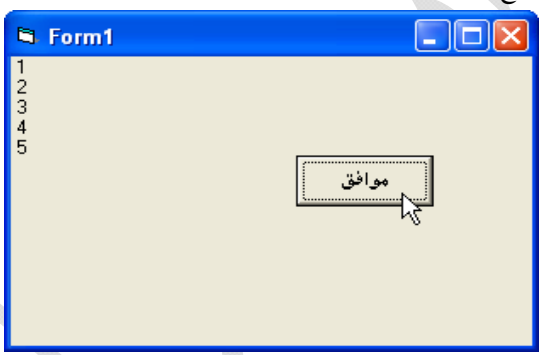
قم بتعديل الكود السابق ليصبح كما يلى :

```
Dim a As Integer
Do While Not a = 5
a = a + 1
Print a
Loop
```

ولاحظ أن نافذة الكود ستصبح كما بالشكل التالي :



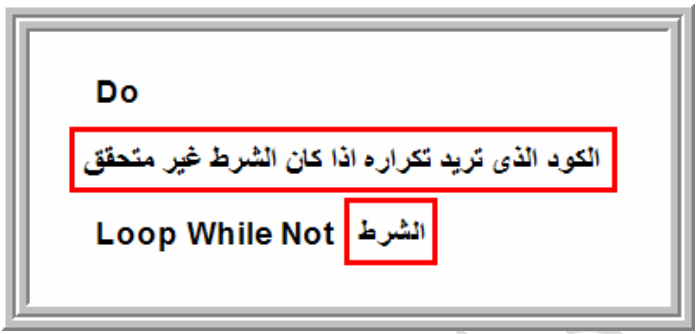
قم بتشغيل البرنامج لتظهر ولاحظ النتيجة كما يلى :



الصورة الرابعة للحلقة التكرارية Do While تكون على الشكل التالي :

```
Do
Code
Loop While Not condition
```

والشكل التالي يوضح تلك الصورة للحلقة التكرارية .



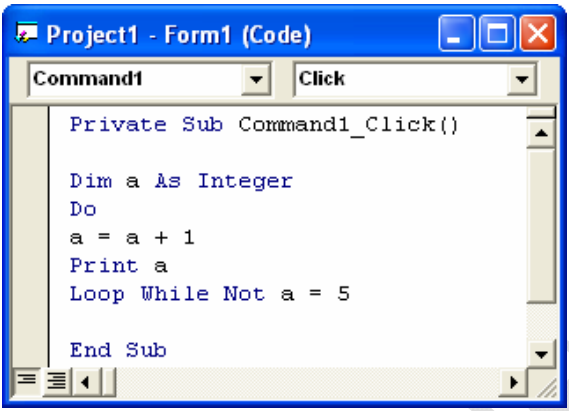
هذه الصورة سيتم فيها تنفيذ الكود ثم يتم اختبار عدم تحقق الشرط فإذا كان الشرط غير متحقق فسيعاد تنفيذ الكود مرة أخرى وهكذا إلى أن يصبح الشرط متحقق فيـتم الخـروج من الحلقة التكرارية .

مثال :

قم بتعديل الكود الخاص بالمثال السابق ليصبح كما يلى :

```
Dim a As Integer
Do
a = a + 1
Print a
Loop While Not a = 5
```

وتصبح نافذة الكود كما بالشكل التالي :



ولاحظ عند بتشغيل البرنامج ستكون النتيجة كالتالى :

الحلقة التكرارية Do Until ... Loop :

الصورة الأولى للحلقة التكرارية Do Until تكون على الشكل التالي :

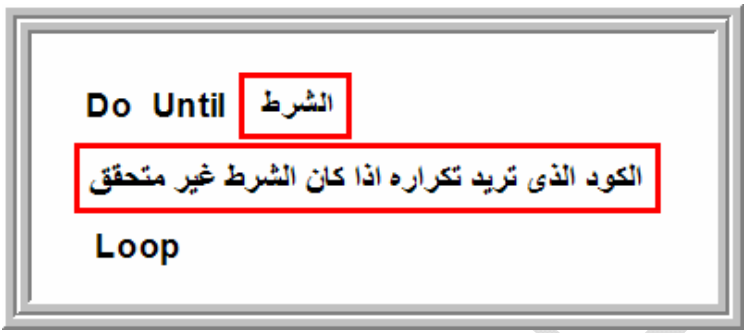
```
Do Until condition
Code
Loop
```

حيث يمكن التعرف على المعاملات السابقة من خلال ما يلى :

Condition : الشرط الذي سيتم اختبار عدم تحققه .

Code : الكود الذي سيتم تكراره إذا لم يتحقق الشرط .

والشكل التالي يوضح تلك الصورة للحلقة التكرارية .



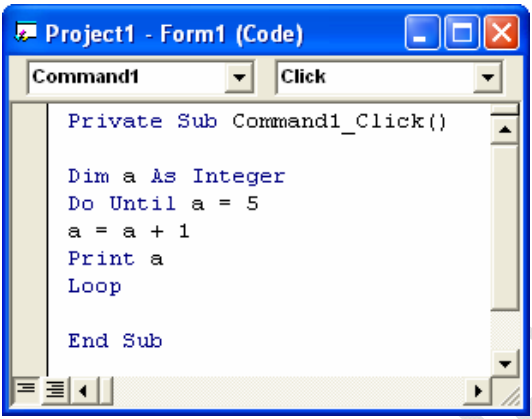
في هذه الصورة للأمر Do Until يتم التأكد من عدم تحقق الشرط قبل الدخول في الحلقة التكرارية فاذا كان الشرط غير متحقق فسيتم تنفيذ الكود الموجود داخل الحلقة التكرارية ثم تعاود الحلقة اختبار الشرط مرة أخرى فإذا كان الشرط ما زال غير متحققاً فسيتم تنفيذ الكود مرة أخرى وهكذا إلى أن يصبح الشرط متحقق فيتم الخروج من الحلقة التكرارية .

مثال :

سنقوم هنا أيضاً **باستخدام** نفس المثال السابق مع تغيير الكود ليصبح كما يلى :

```
Dim a As Integer
Do Until a = 5
a = a + 1
Print a
Loop
```

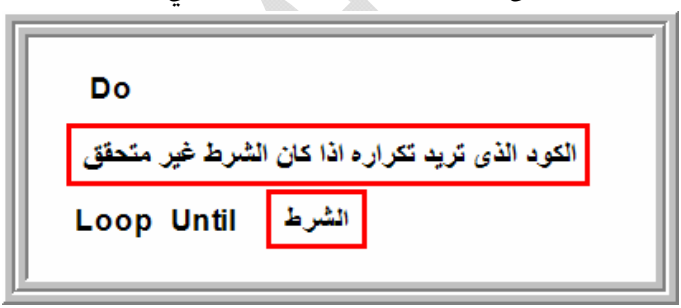
ولاحظ أن نافذة الكود ستصبح كما بالشكل التالي :



الصورة الثانية للحلقة التكرارية Do Until تكون على الشكل التالي :

```
Do
Code
Loop Until condition
```

ويمكن كذلك التعرف على هذه الصورة من خلال الشكل التالي :



هذه الصورة تشبه الصورة السابقة غير أن في هذه الصورة يتم تنفيذ الكود أول مرة ثم يتم اختبار الشرط فإذا كان الشرط غير متحققاً يعاد تنفيذ الكود مرة أخرى وهكذا إلى أن يتحقق الشرط فيتم الخروج من الحلقة التكرارية .

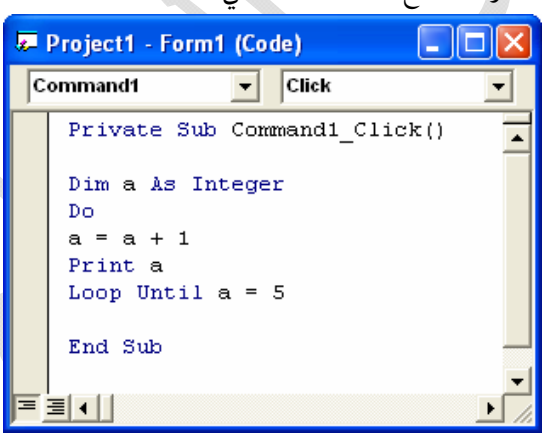
لاحظ ... أن الفرق بين هذه الصورة والصورة السابقة أن في هذه الصورة يتم تنفيذ الكود قبل التأكد من عدم تحقق الشرط وبذلك فإن من خلال هذه الصورة سيتم تنفيذ الكود مرة واحدة على الأقل .

مثال :

قم بتعديل الكود السابق ليصبح كما يلي :

```
Dim a As Integer
Do
a = a + 1
Print a
Loop Until a = 5
```

ولاحظ أن نافذة الكود ستصبح كما بالشكل التالي :

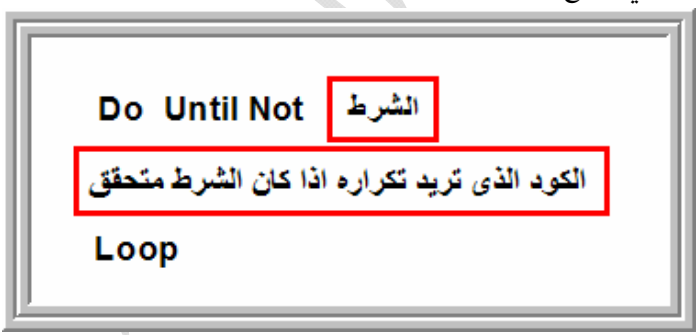


وعند تشغيل البرنامج ستظهر لك النتيجة التالية :

الصورة الثالثة للحلقة التكرارية Do Until تكون على الشكل التالي :

```
Do Until Not condition
Code
Loop
```

والشكل التالي يوضح لك تلك الصورة للحلقة التكرارية .



في هذه الصورة تم إضافة الأمر Not لنفى عدم تحقق الشرط وبذلك سيتم التأكد من تحقق الشرط قبل الدخول في الحلقة التكرارية فاذا كان الشرط غير متحقق سيتم تجاوز الحلقة وتجاهل الكود الموجود بداخلها أما إذا كان الشرط متحقق فسيتم تنفيذ الكود الموجود داخل الحلقة التكرارية ثم تعاود الحلقة اختبار الشرط مرة أخرى فإذا كان

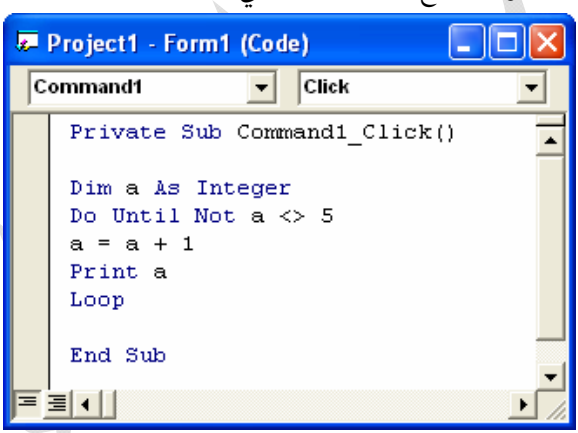
الشرط ما زال متحق فسيتم تنفيذ الكود مرة أخرى وهكذا إلى أن يصبح الشرط غير متحقق فيتم الخروج من الحلقة التكرارية .

مثال :

قم بتعديل الكود السابق ليصبح كما يلى :

```
Dim a As Integer
Do Until Not a <> 5
a = a + 1
Print a
Loop
```

ولاحظ أن نافذة الكود ستصبح كما بالشكل التالي :

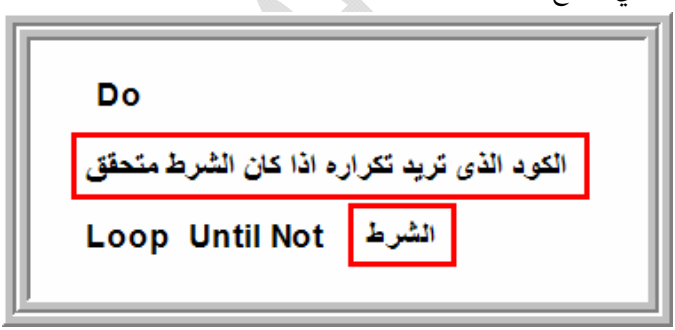


قم بتشغيل البرنامج ولاحظ النتيجة كما بالشكل التالي :

الصورة الرابعة للحلقة التكرارية Do Until تكون على الشكل التالي :

```
Do
Code
Loop Until Not condition
```

والشكل التالي يوضح تلك الصورة للحلقة التكرارية .



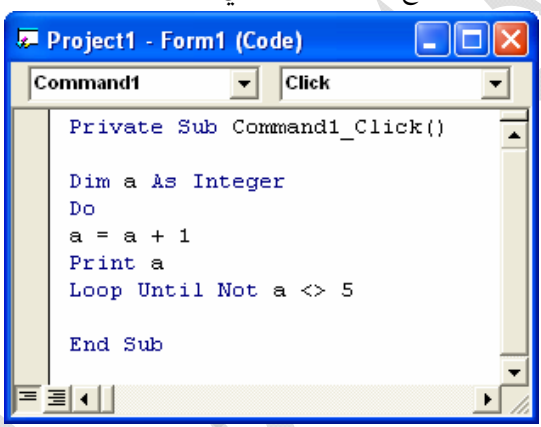
في هذه الصورة سيتم تنفيذ الكود الموجود داخل الحلقة التكرارية ثم يتم اختبار الشرط فإذا كان الشرط متحقق فسيعاد تنفيذ الكود مرة أخرى وهكذا إلى أن يصبح الشرط غير متحقق فسيتم الخروج من الحلقة التكرارية .

مثال :

قم بتعديل الكود السابق ليصبح كما يلى :

```
Dim a As Integer
Do
a = a + 1
Print a
Loop Until Not a <> 5
```

ولاحظ أن نافذة الكود ستصبح كما بالشكل التالي :



قم بتشغيل البرنامج ولاحظ أن النتيجة كالعادة ستكون كالتالى :

لاحظ ...

أننا خلال الحلقات التكرارية قمنا بالتركيز على مثال واحد مع جميع صور الحلقات المختلفة وكنا نقوم بتعديل الكود وتطويعه للحصول على نفس النتيجة في كل مرة وكان ذلك بهدف القاء الضوء على مرونة الحلقات التكرارية وامكانية استعمال أي صورة منها في تنفيذ ما تريد ، وفى الواقع أن هناك بعض من هذه الصور منتشرة وتستخدم على نطاق واسع وصور أخرى غير شائعة الاستخدام بين أوساط المبرمجين بل أن هناك صور لا يتم استخدامها وهذا ليس سببه عدم أهمية احدى الصور أو جدواها ولكن ذلك قد يكون بسبب أن نفس النتيجه يمكن الحصول عليها بصورة أخرى بسيطه .