الصيغة التكرارية الشرطية Do..Loop :

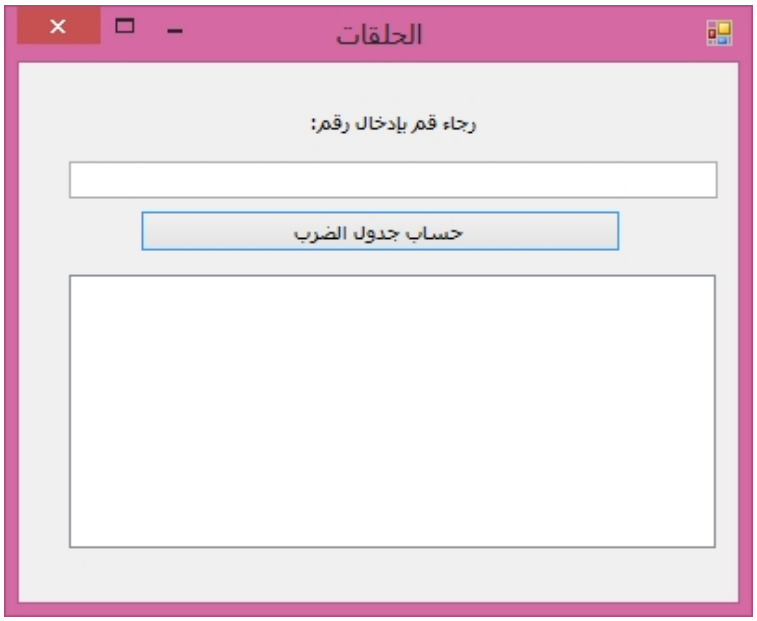
ترتكز البنية التكرارية من نوع Do..Loop على شرط التكرار وحسب نتيجته يتم إعادة تنفيذ الأوامر، ويوجد بها عدة أنواع سوف نتعرف عليها بالترتيب:

الصيغة التكرارية الشرطية Do..While :

هذه البنية التكرارية تعتمد على نتيجة الشرط الذي يأتي بعد الكلمة While، ويتم تكرار تنفيذ الأوامر مادامت نتيجة الشرط متحققة True، وهذه صيغتها:

```
Do
    'الأوامر
Loop While 'الشرط
```

سنتعرف الآن بحول الله على مثال لاستخدام هذه البنية التكرارية، قم بإنشاء مشروع جديد من نوع Windows Forms Application واسحب على الفورم الأدوات التالية:



الأداة	اسمها	دورها
Label	lblNumber	من أجل عرض السؤال
TextBox	txtNumber	من أجل إدخال الرقم
Button	btnCalculate	من أجل حساب جدول ضرب الرقم
ListBox	lbDetails	من أجل عرض نتائج جدول الضرب

الآن أدخل إلى الحدث Click الخاص بالزر btnCalculate وقم بكتابة الكود الآتي:

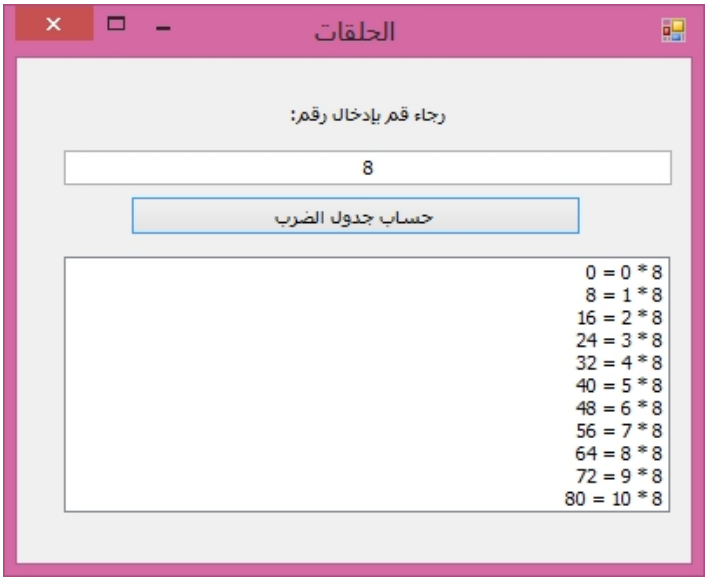
```
    Private Sub btnCheck_Click(sender As Object, e As
EventArgs) Handles btnCalculate.Click

        Dim Number As Integer = Val(txtNumber.Text)
        Dim Counter As Integer = 0
        Do
            lbDetails.Items.Add(Number & " * " & Counter & "
= " & Number * Counter)
            Counter += 1
        Loop While Counter <= 10

    End Sub
```

أعلنا في الأول عن متغير اسمه Number يستقبل القيمة المدخلة في مربع النص txtNumber بعد أن يقوم بتحويلها إلى قيمة رقمية من خلال الدالة Val ثم بعد ذلك أهلنا عن متغير عداد اسمه Counter أعطيناه القيمة 0 كقيمة بدئية، ثم بدأنا عملية التكرر التي تقوم بعرض نتيجة ضرب قيمة العدد المدخل Number مع قيمة العداد Counter التي تتغير تزايديا عند كل تكرار، ويتم تكرار هذه العملية التكرارية إلى أن تصبح قيمة العداد 10 أي أن جدول الضرب الذي سيعرض سيظهر لنا

نتيجة جداء العدد المدخل مع الأعداد الإحدى عشر الأولى بدء من 0 وانتهاء ب 10، أي بعد تنفيذ البرنامج لو أدخلنا القيمة 8 سنحصل على النتيجة الآتية:



يمكننا تقديم الكلمة While بعد الكلمة Do وسنحصل على نفس النتيجة كما يلي:

```
        Dim Number As Integer = Val(txtNumber.Text)
        Dim Counter As Integer = 0
        Do While Counter <= 10
            lbDetails.Items.Add(Number & " * " & Counter & "
= " & Number * Counter)
            Counter += 1
        Loop
```

الفرق بين تقديم While وبين تأخيرها يتمثل في أن الأولى تمنع تنفيذ الأمر إلا بعد أن يتحقق الشرط، بينما While التي تأتي في الأخير بعد Loop تسمح بتنفيذ الأمر للمرة الأولى حتى مع عدم تحقق الشرط، والكود التالي يوضح الفرق بين الأمرين.

تقديم While:

```
Do While 1 = 3
    MsgBox("هذا الكود لن ينفذ أبدا")
Loop
```

في الكود أعلاه لن يتم عرض الرسالة لأن شرط While غير متحقق لأن 1 لا يساوي ثلاثة ولأن While جاءت أولا في بداية التكرار.

تأخير While:

```
Do
    MsgBox("هذا الكود سينفذ مرة واحدة")
Loop While 1 = 3
```

في الكود أعلاه سيتم عرض الرسالة في كل مرة يتم تنفيذه لأن While أتت في نهاية التكرار على خلاف الكود الأول.

الصيغة التكرارية الشرطية Do..Until :

مثلها مثل البنية السابقة Do..While إلا أن هذه البنية تسمح بتكرار الأوامر إلى أن يتحقق الشرط وليس مادام الشرط متحققا، بمعنى أن التعليمات التي ستأتي بعد الكلمة Do سيتم تكرارها إلى غاية تحقق الشرط الذي يأتي بعد الكلمة Until.

الفرق بين Until و While:

Until	While
تسمح بتكرار الأوامر إلى أن يتحقق الشرط أي تصبح نتيجته True	تسمح بالتكرار مادام الشرط متحققا ويخرج من التكرار حينما لا يتحقق الشرط أي تصبح نتيجته False

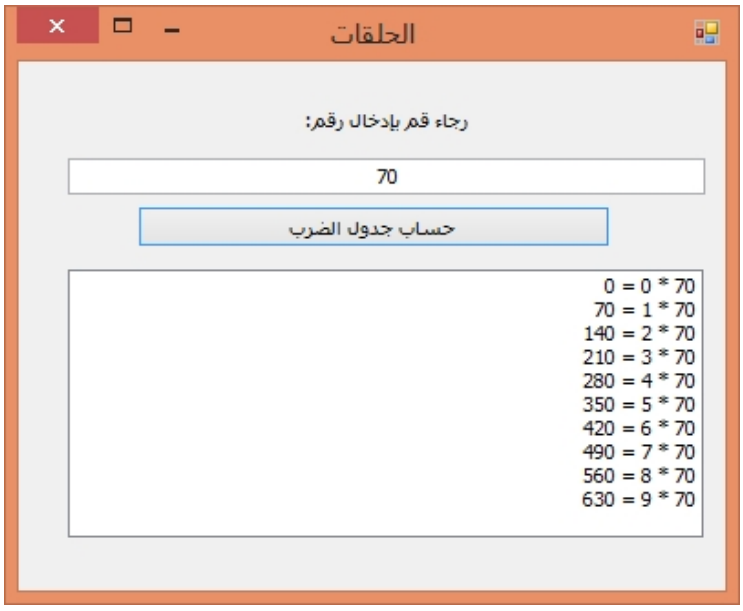
سنشتغل على نفس المثال السابق الذي يقوم بحساب جدول الضرب، لكن هذه المرة مع استخدام الأمر Until بدل While، أي أن الشفرة ستكون كما يلي:

```
    Private Sub btnCheck_Click(sender As Object, e As
EventArgs) Handles btnCalculate.Click

        Dim Number As Integer = Val(txtNumber.Text)
        Dim Counter As Integer = 0
        Do Until Counter = 10
            lbDetails.Items.Add(Number & " * " & Counter & "
= " & Number * Counter)
            Counter += 1
        Loop

    End Sub
```

لاحظ أننا غيرنا الشرط، بدل أن يكون العداد أصغر من أو يساوي 10 أصبح معنى الشرط، قم بتكرار الأوامر إلى أن يصبح العداد Counter مساويا ل 10 وفي كل مرة تزداد قيمته بواحد إلى أن يتحقق الشرط فينتهي التكرار، لو نفذنا الكود أعلاه وأدخلنا العدد 70 مثلا سوف نحصل على النتيجة التالية:



رأينا فيما تقدم أنه يمكننا تسبيق الكلمة While وتأخيرها، نفس الكلام ينطبق على الكلمة Until يمكننا وضعها في البداية بعد الكلمة Do ويمكننا كذلك وضعها في نهاية التكرار بعد الكلمة Loop، في الحالتين معا سيتم تنفيذ الأوامر مادام الشرط غير متحققا، بمعنى أن الكود التالي:

```
Do Until 1 = 3
    MsgBox("هذا الكود سينفذ من دون توقف لأن الشرط غير متحقق ولن يتحقق")
Loop
```

هو نفسه الكود التالي:

```
Do
    MsgBox("هذا الكود سينفذ من دون توقف لأن الشرط غير متحقق ولن يتحقق")
Loop Until 1 = 3
```

ففي الحالتين معا سيتم عرض الرسالة من دون توقف لأن الشرط غير متحقق دائما، ومادام الأمر كذلك فستسمر عملية تكرار الأوامر.

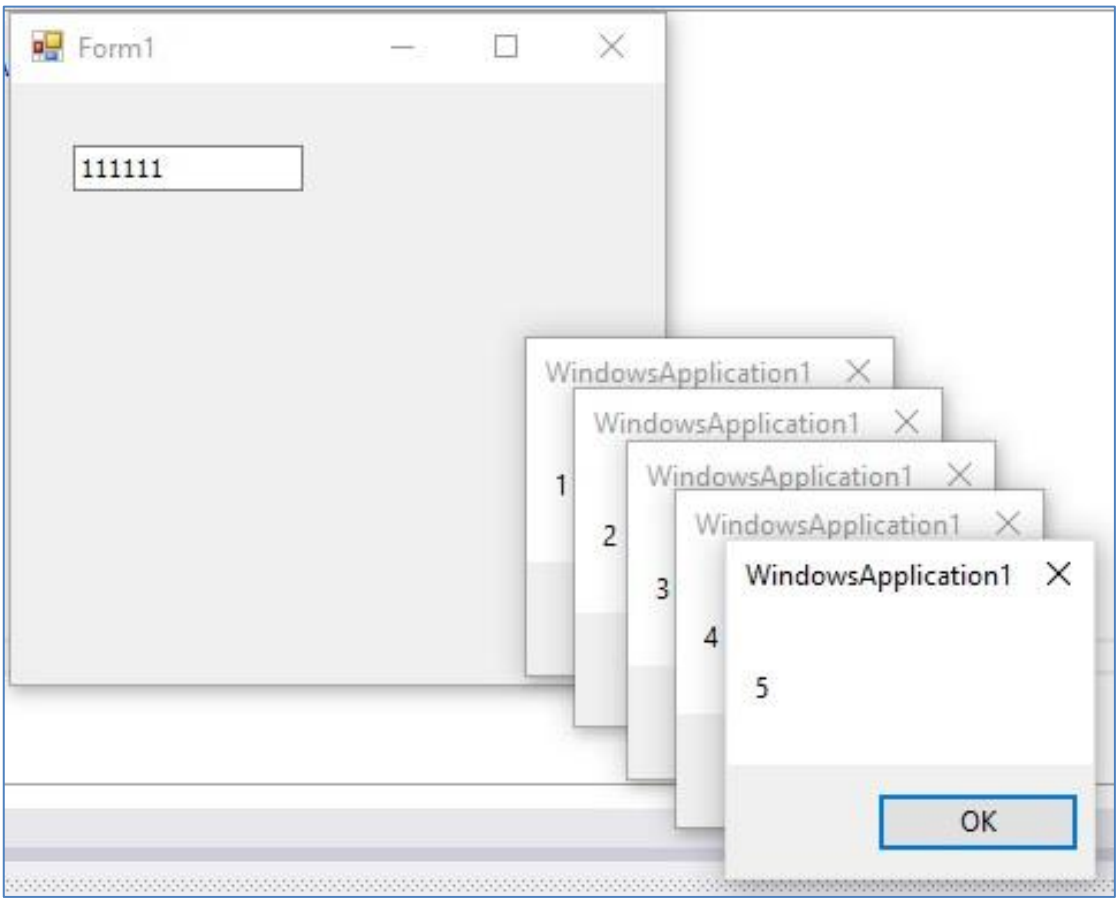
س١ / باستخدام الأدوات التي درستها فقط صمم ونفذ برنامج يقوم بإظهار خمس رسائل نصية خلال عشر ثواني (رسالة كل ثانيتين) يكون محتوى الرسائل الأرقام من ١ الى ٥ حسب التسلسل ثم أوقف المهمة .

ملاحظة : غيّر الخاصية (**Name**) للاداة **Textbox1** التي تحتاجها في كتابة الكود الى اسمك باللغة الإنكليزية.

ملاحظة ٢ : الاجابة تكون صورة تجمع بين شاشة التنفيذ النهائية وشاشة تحرير الكود يظهر من خلالها الكود + التنفيذ بصورة واضحة.

الجواب :

١- التصميم : يتم استخدام الأدوات (Textbox) و (Timer) فقط



٢- كتابة الكود : يتم كتابة الكود باستخدام if statement وذلك بفرض وجود عداد يحسب الوقت المحدد لاظهار الرسائل النصية المطلوبة في الوقت المحدد ، بعد ضبط interval الخاص بالاداة timer1 الى (2000) .

- اذا لم يحدد الـ interval في السؤال يمكن اعتماده من خلال تقسيم الوقت الكلي على عدد المهام المطلوبة.
- يتم كتابة كود المهمة في الأداة timer1 ، اما دالة تشغيل المؤقت timer1 فتوضع Form_load .

```
Private Sub Form1_Load(sender As Object, e As EventArgs) Handles
MyBase.Load
        Timer1.Start()
    End Sub

    Private Sub Timer1_Tick(sender As Object, e As EventArgs)
Handles Timer1.Tick
        If dawood.Text = "111111" Then
            Timer1.Stop()
        Else
            dawood.Text += "1"
            If dawood.Text = "1" Then
                MsgBox("1")
            ElseIf dawood.Text = "11" Then
                MsgBox("2")
            ElseIf dawood.Text = "111" Then
                MsgBox("3")
            ElseIf dawood.Text = "1111" Then
                MsgBox("4")
            ElseIf dawood.Text = "11111" Then
                MsgBox("5")
            End If

        End If
    End Sub
End Class
```

س٢ / باستخدام الأدوات التي درستها فقط صمم ونفذ برنامج يقوم بتغيير لون خلفية Textbox1 اربع مرات نصية خلال ١٢ ثانية (لون كل ٣ ثواني) دون توقف.

ملاحظة : غيّر الخاصية (Name) للأداة Textbox1 التي تحتاجها في كتابة الكود الى اسمك باللغة الإنكليزية.

ملاحظة ٢ : الاجابة تكون صورة تجمع بين شاشة التنفيذ النهائية وشاشة تحرير الكود يظهر من خلالها الكود + التنفيذ بصورة واضحة.

٣- التصميم : يتم استخدام الأدوات (Textbox) و (Timer) فقط لتكون شاشة التنفيذ بعد كتابة الكود بالحالات التالية:

٤- كتابة الكود : يتم كتابة الكود باستخدام if statement وذلك بفرض وجود عداد يحسب الوقت المحدد لتغيير لون خلفية مربع النص ضمن الأوقات المطلوبة ، بعد ضبط interval الخاص بالاداة timer1 الى (3000) .

- اذا لم يحدد الـ interval في السؤال يمكن اعتماده من خلال تقسيم الوقت الكلي على عدد المهام المطلوبة.
- يتم كتابة كود المهمة في الأداة timer1 ، اما دالة تشغيل المؤقت timer1 فتوضع Form_load .

```
Public Class Form1


    Private Sub Form1_Load(sender As Object, e As EventArgs)
Handles MyBase.Load
        Timer1.Start()
    End Sub

    Private Sub Timer1_Tick(sender As Object, e As EventArgs)
Handles Timer1.Tick
        If dawood.Text = "1111" Then
            dawood.Text = ""
        Else
            dawood.Text += "1"
            If dawood.Text = "1" Then
                dawood.BackColor = Color.Red
            ElseIf dawood.Text = "11" Then
                dawood.BackColor = Color.Green
            ElseIf dawood.Text = "111" Then
                dawood.BackColor = Color.Yellow
            ElseIf dawood.Text = "1111" Then
                dawood.BackColor = Color.Blue

            End If

        End If
```

البيانات :

هي مجموعة من الحروف أو الكلمات أو الأرقام أو الرموز أو الصور المتعلقة بموضوع معين، ومثال ذلك: بيانات الموظفين (الأسماء - الأرقام الوظيفية - المهن - الصور) بدون ترتيب، وينتج عن هذه البيانات بعد المعالجة ما يطلق عليه مصطلح معلومات.

انواع البيانات

- المتغيرات Variables
- الثوابت Constant

المتغيرات **Variables**

المتغيرات عبارة عن مخازن مؤقتة في الذاكرة تخزن فيها البيانات أثناء تنفيذ خطوات البرنامج لتجري عليها العمليات المطلوبة وكذلك تخزن فيها نتائج تلك العمليات.
والمتغيرات هي أحد القواعد الأساسية في كل لغات البرمجة سواء في الإنترنت أم في لغات برمجة الحاسب . لدرجة انك لا تجد برنامج مهما كان حجمه أو وظيفته ويخلو من متغير أو اكثر.

انواع المتغيرات : تنقسم المتغيرات الى عدة انواع وحسب حاجة المبرمج , ونذكر هنا أبرزها :

1- المتغيرات الرقمية (Integer , single , long , double) والتي تختلف باختلاف سعتها .
2- المتغيرات الحرفية string تستخدم لتخزين قيم البيانات الحرفية مثل الاسماء وتشمل كل حروف لوحة المفاتيح.
3- المتغيرات المنطقية Boolean تستخدم لتخزين قيم البيانات المنطقية (yes , no) او (true , false).

الاعلان عن المتغير ات : للاعلان عن المتغير الذي تحتاجه في البرنامج هناك صيغة عامة وهي :

Dim اسم المتغير as نوع المتغير

Dim Variable As type of Variable

مثلا : للاعلان عن متغير اسمه x وهو متغير حرفي يكون الاعلان كالتالي:

Dim x As string

وللاعلان عن متغير رقمي كبير اسمه willeam يكون :

Dim willeam as double

بعض الامثلة الصحيحة :

dim n as string - هنا نحجز مكان فى الذاكرة اسمه n ونوعة string **حرفى**

dim a1 as integer كما تعلمنا ان الاسم ياتى بعد كلمة dim اذن اسم المتغير هنا a1 ونوعه**integer**

dim abc as double اسم المتغير abc ونوعة**double**

dim ahmed as single اسم المتغير ahmed ونوعة**single**

بعض الامثلة على الاستدعاءات الخاطئة لتجنبها :

- dim 1ahmed as integer خطا لانه يبدا برقم
- dim a 1 as integer خطا لا نة يحتوى على مسافة

الدالتين Val و Str

من المهام الشائعة في مشروعات فيجوال بيسك أن يدخل المستخدم أرقام لتنفيذ بعض العمليات علي تلك الأعداد ، فقد تستخدم المؤثرات الحسابية لحساب النتيجة وأداة العنوان Label لعرض النتائج ولكن لدينا مشكلتين هما :

- أن المؤثرات الحسابية تعمل فقط مع الأعداد (مثل المتغيرات الصحيحة والخواص الصحيحة) ولكن القيم التي يدخلها المستخدم من خلال أداة النص تكون حرفية String وبالتالي لا يمكنك أن تضرب المتغيرات الحرفية أو القيم الحرفية .
- أن نتيجة إجراء العمليات الحسابية تكون أعداد بينما أداة العنوان أو نص تقبل قيم حرفية ، فلا يمكنك أن تختزن قيم عددين في متغيرات حرفية مباشرة .

لذا فنحن نحتاج لحل تلك المشكلتين ، والحل موجود في دوال الفيجوال بيسك ، فيمكننا أن نحول القيم العددية إلي حرفية بالدالة STR ، ويمكننا استرجاع تلك الأعداد التي تحولت لحروف بالدالة Val .

مثال :

هنا عبارة عن قيمة نصية Textbox.text=x

اما هنا فعبارة عن قيمة عددية Val (Textbox.text)=x

تمرين / صمم واجهة آلة حاسبة للقيام بالعمليات الحسابية بين عددين (الجمع , الطرح , الضرب والقسمة) على ان تكون كل عملية في زر مختلف والاعداد المدخلة والناتج في مربعات نصوص مختلفة وكما موضحة في الشكل التالي ثم نفذ :

- برمجة العمليات ومخرجاتها بالصورة الاعتيادية.
- برمجتها باستخدام المتغيرات .

خطوات الحل :

1- التخطيط و التصميم :

عند فتح برنامج الفيجوال بيسك سوف يظهر لك شكل الفورم فارغ كما مبين سابقاً , الان نقوم بتصميم البرنامج حسب الواجهة المطلوبة (يجب الاعتناء بتصميم واجهة البرنامج لأنها مهمة) وذلك :

- بالذهاب الى صندوق الادوات ToolBox والضغط على احدى الادوات بالزر الايمن (اي اختيار الاداة المناسبة للبرنامج) .
- نذهب الى الـ Form النموذج ونضع الاداة بالمكان المناسب .
- تغيير شكل او حجم الاداة بما يناسب البرنامج .
- تغيير خصائص الاداة بعد تحديدها وذلك من خلال الخصائص الموجودة على يمين البرنامج (تم شرحها مسبقاً) , وهنا نحتاج الى الادوات التالية:

الاداة	الخاصية	القيمة الجديدة
Label1	text	الالة الحاسبة
Button1	text	جمع
Button2	text	طرح
Button3	text	ضرب
Button4	text	قسمة
Button5	text	مسح
Button6	text	خروج
Textbox1 Textbox2 Textbox2	Text TextAlign	“ “ center

2- كتابة الشيفرة والاختبار والتنفيذ

- كتابة شفرة الاداة بعد الضغط عليها مرتين او من خلال نافذة مستكشف المشروع Project Explorer Window من الخانة View Code وكتابة الشفرة في نافذة Code Window (وسوف يتم التطرق اليها لاحقاً) .
- تنفيذ البرنامج وذلك من خلال الايعاز Run الموجود في Standard Toolbar شريط الادوات القياسي او من خلال قائمة Run ومن ثم Start او F5 .
- لإيقاف تشغيل البرنامج نضغط على End الموجودة على ■ شريط الادوات القياسي او من خلال قائمة Run ومن ثم End .

استجابة للمطلوب في التمرين سنكتب الكود في الحالتين وستكون الاولى بدون استخدام المتغيرات :

- الكود الخاص بزر الجمع (Button1) :

```
TextBox3.Text = Val(TextBox1.Text) + Val(TextBox2.Text)
```

- الكود الخاص بزر الطرح (Button2) :

```
TextBox3.Text = Val(TextBox1.Text) - Val(TextBox2.Text)
```

- الكود الخاص بزر الضرب (Button3) :

```
TextBox3.Text = Val(TextBox1.Text) * Val(TextBox2.Text)
```

- الكود الخاص بزر الطرح (Button4) :

```
TextBox3.Text = Val(TextBox1.Text) / Val(TextBox2.Text)
```

اما الكود الخاص بزر المسح (Button5) :

```
TextBox1.Text=""
TextBox2.Text=""
TextBox3.Text=""
```

اما كود الخروج (Button6) :

```
End
```

الكود الثاني باستخدام المتغيرات:

بداية يتم تعريف واستدعاء المتغيرات في القسم العام ثم تسند للمتغيرات في كل اداة :

```
  Public Class Form1
    Dim x, y, z As Integer


    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
        x = Val(TextBox1.Text)
        y = Val(TextBox2.Text)
        TextBox3.Text = x + y
    End Sub

    Private Sub Button2_Click(sender As Object, e As EventArgs) Handles Button2.Click
        x = Val(TextBox1.Text)
        y = Val(TextBox2.Text)
        TextBox3.Text = x - y
    End Sub

    Private Sub Button3_Click(sender As Object, e As EventArgs) Handles Button3.Click
        x = Val(TextBox1.Text)
        y = Val(TextBox2.Text)
        TextBox3.Text = x * y
    End Sub

    Private Sub Button4_Click(sender As Object, e As EventArgs) Handles Button4.Click
        x = Val(TextBox1.Text)
        y = Val(TextBox2.Text)
        TextBox3.Text = x / y
    End Sub

    Private Sub Button5_Click(sender As Object, e As EventArgs) Handles Button5.Click
        TextBox1.Text = ""
        TextBox2.Text = ""
        TextBox3.Text = ""
    End Sub

    Private Sub Button6_Click(sender As Object, e As EventArgs) Handles Button6.Click
        End
    End Sub
End Class
```

انواع البيانات :

انواع البيانات تمثل نوع القيم التي نريد تخزينها في المتغيرات في المتغيرات التي نعلن عنها في برامجنا اذ لو كانت القيمة رقمية نستخدم بيانات رقمية Integer وان كانت نصية نستخدم String وهكذا , هذه القيم يتم تخزينها في الذاكرة اثناء تنفيذ البرنامج بواسطة Common Language Runtime(CLR) .

بالنسبة للقيم الرقمية سوف نتعامل مع قسمين من انواع البيانات وهي :

- الانواع الرقمية الصحيحة Integer
- الانواع الرقمية العشرية Floating-Point

الانواع الطبيعية الصحيحة تشمل الارقام التي لا تحتوي على فاصلة عشرية وتنقسم في لغة فيجوال بيسك الى عدة اقسام حسب المجال الرقمي الذي تختص به , مثلا نوع البيانات الرقمي Byte الذي يشمل الارقام الصحيحة الموجودة في المجال من 0 الى 255 , والنوع الرقمي Short الذي يبدأ من العدد 32,768- الى 32,767 .

بينما تشمل الانواع العشرية Floating-Point كل الانواع الرقمية بما فيها تلك التي تحتوي على فاصلة عشرية ومن بين الانواع العشرية المستخدمة في لغة فيجوال بيسك نوع Double .

في المثال التالي سوف نعلن عن متغير بأسم (Ali) بانواع رقمية صحيحة طبيعية وعشرية مختلفة :

```
Dim ali As Integer
Dim ali As Byte
Dim ali As SByte
Dim ali As Long
Dim ali As Single
Dim ali As UInteger
Dim ali As ULong
Dim ali As UShort
```

ملاحظة : بعض الانواع الرقمية اعلاه كانت مسبوقة بالحرف U مثل UInteger وهي اختصار لكلمة Unsigned والتي تعني ان هذا النوع الرقمي خاص باحتواء القيم الموجبة فقط , بمعنى اننا لو احتجنا الى متغير لتخزين القيم الموجبة فقط دون القيم السالبة نقوم بالاعلان عنه من نوع Unsigned لانه سياخذ نفس مساحة التخزين في الذاكرة التي ياخذها النوع الاصلي اضافة الى انه يتيح لك مجالا رقميا يضاعف المجال الرقمي الخاص بالنوع الاصلي.

مثلا النوع Short يسمح باستعمال المجال الرقمي من -32,768 الى 32,767 فلو استعملنا النوع Unsigned من هذا النوع أي (UShort) سيسمح لنا بتخزين ضعف هذا المجال لانه لا يقبل القيم السلبية وبالتالي يتيح لنا مجالا رقميا اكبر للاعداد الايجابية , اي ان النوع UShort سيسمح لنا بتخزين القيم الرقمية من 0 الى 65,535 .

نجد في لغة الفيجوال بيسك بعض الانواع الرقمية الاخرى مثل Int64 , Int32 , Int16 , وهي ليست انواع مختلفة بل هي تسميات اخرى لبعض الانواع الرقمية التي ذكرناها نحو :

short	Int16
Integer	Int32
Long	Int64

اما الانواع النصية فيوجد لدينا النوع String الذي يمكننا تخصيصه لتخزين القيم النصية , ويوجد كذلك نوع Char الذي يسمح بتخزين رممز واحد وليس سلسلة نصية . والجدول التالي يبين مختلف انواع البيانات المتوفرة في لغة بيسك :

Boolean	2 بايت	يقبل القيمتين True و False فقط.
Byte	1 بايت	من 0 إلى 255
SByte	1 بايت	من 128- إلى 127
Char	2 بايت	أي رمز أحادي Unicode Symbol
Date	8 بايت	من 1 يناير 0001 إلى 31 ديسمبر 9999
Decimal	16 بايت	
Double	8 بايت	
Integer	4 بايت	من 2,147,483,648- إلى 2,147,483,647
UInteger	4 بايت	من 0 إلى 4,294,967,295
Long	8 بايت	من -9,223,372,036,854,775,808 إلى 9,223,372,036,854,775,807
ULong	8 بايت	من 0 إلى 18,446,744,073,709,551,615
Object	4 بايت	القيم من جميع الأنواع ممكن تخزينها في النوع Object
Short	2 بايت	من 32,768 - إلى 32,767
UShort	2 بايت	من 0 إلى 65,535
Single	4 بايت	من -3.4028235E38 إلى 3.4028235E38
String	غالبا 2 بايت لكل رمز	من 0 إلى 2 بليون رمز أحادي Unicode Character

اسناد القيمة الى المتغير :

لاسناد قيمة ابتدائية لاي متغير فان الطريقة تكون عبر استخدام رمز (=) ثم بعده القيمة المراد اعطاؤها للمتغير ويمكن عمل ذلك مع الاعلان او بعده , نحو :

```
Dim ali As Integer = 6
```

شرح الكود :

عرفنا متغير رقمي اسمه (ali) من نوع (Integer) واسندنا له قيمة ابتدائية = 6 .

ومن خلال الكود اعلاه نلاحظ ان المتغيرات الرقمية تستقبل القيم بشكل مباشر , بينما المتغيرات النصية يلزم استخدام علامات التنصيص " " ووضع النص داخلهما كما في الكود التالي :

```
Dim x As String = " Dawood Zahi "
```

شرح الكود :

عرفنا متغير نصي اسمه (ali) من نوع (String) واسندنا له قيمة ابتدائية = Dawood Zahi

ملاحظة : يمكننا اسناد قيمة متغير الى متغير اخر كما في الكود التالي :

```
Dim x As String = " Dawood Zahi "
Dim g As String
g = x
```

شرح الكود :

عرفنا متغير نصي اسمه (x) من نوع (String) واسندنا له قيمة ابتدائية = Dawood Zahi , ثم عرفنا متغير نصي اخر اسمه (g) من نوع (String) ثم اسندنا قيمة المتغير (x) الى المتغير (g) .

او تختصر بهذا الشكل :

```
Dim x As String = " Dawood Zahi "
Dim g As String = x
```

الروابط (المعاملات) :

هي رموز تستخدم لاجراء بعض العمليات على المتغيرات او على القيم بشكل عام مثل العمليات الحسابية او عمليات المقارنة (تحديد القيمة الكبرى او الصغرى).

الروابط الحسابية او الرياضية Arithmetic operators :

هي روابط تستخدم للقيام بالعمليات الرياضية كالجمع , الطرح , الضرب , القسمة , باقي القسمة و الأُس. وفي المثال التالي نعرف متغيرين ونسند لهما قيم اولية ثم نعرف متغير لكل عملية نسند له قيم العمليات الرياضية بين المتغيرين الاول والثاني , ليكون الكود كالتالي :

```
Dim x As Integer = 6
Dim y As Integer = 2
رابط الجمع'
```

```
Dim Sum As Integer = x + y
'رابط الطرح
Dim Dif As Integer = x - y
'رابط الضرب
Dim Mul As Integer = x * y
'رابط القسمة
Dim Div As Integer = x / y
'رابط باقي القسمة
Dim Modulus As Integer = x Mod y
'رابط القوة
Dim Exp As Integer = x ^ y
```

روابط دمج النصوص String Concatenation operators:

تستخدم لدمج نصين او اكثر مع بعضهم البعض من خلال استخدام + او & نحو :

```
Dim x As String = "Ali "
Dim y As String = "hasan"
Dim z As String = x & y
'أو
Dim z As String = x + y
```

ليكون ناتج (z) هو دمج قيمتي المتغيرير (x) و (y) بالشكل التالي : Ali hasan

مثال1 : صمم ونفذ برنامج اشارة مرور متكامل تعمل باستمرار وبالتقسيمات التالية :

اللون الاحمر 10 ثواني

اللون الاصفر 3 ثواني

اللون الاخضر 8 ثواني

الجواب –

مرحلة التصميم :

نحتاج لتصميم الواجهة الادوات التالية :

الاداة	عنوانها	وظيفتها
Form	اشارة المرور	يضم مجموعة الادوات
Button	Start	بدء تشغيل اشارة المرور
Textbox	بلا	اللون الاحمر في الاشارة
Textbox	بلا	اللون الاصفر في الاشارة
Textbox	بلا	اللون الاخضر في الاشارة
Textbox	بلا	عداد الوقت
Picturebox	بلا	صورة اشارة المرور
Timer	بلا	يحتوي الكود وينفذ المهمة

كتابة الكود :

- الفكرة هي ايجاد textbox وظيفتها كعداد يكرر رمز او رقم او حرف معين بعدد مجموع الوقت للألوان الثلاثة وفي مثالنا هذا (10 + 3 + 8 = 21) اي اذا وصل عدد مرات طباعة الرمز او الحرف في textbox 21 مرة نصفرها ونبدأ العد من جديد واذا لم يكن يساوي 21 نزيد قراءة واحدة في كل مرة ، ويكتب الكود الخاص بها في Timer.
- ايجاد Button لتشغيل Timer

الكود :

```
Public Class Form1

    Private Sub Timer1_Tick(sender As Object, e As EventArgs) Handles
Timer1.Tick
        If TextBox4.Text = "11111111111111111111" Then
            TextBox4.Text = ""
        Else
            TextBox4.Text = TextBox4.Text + "1"
        End If
        If TextBox4.Text = "111" Then
            TextBox1.BackColor = Color.Red
            TextBox2.BackColor = Color.White
            TextBox3.BackColor = Color.White
        ElseIf TextBox4.Text = "111111111" Then
            TextBox1.BackColor = Color.White
            TextBox2.BackColor = Color.Yellow
            TextBox3.BackColor = Color.White
        ElseIf TextBox4.Text = "111111111111" Then
            TextBox1.BackColor = Color.White
            TextBox2.BackColor = Color.White
            TextBox3.BackColor = Color.Green

        End If

    End Sub

    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles
Button1.Click
        Timer1.Start()

    End Sub


End Class
```

مثال2 : صمم برنامج يعمل بمؤقت Timer 30 ثانية ويقوم بملئ ListBox بالتفاصيل التالية { احمد عند الثانية 5 ، علي عند الثانية 10 ، حسن عند الثانية 15 ، فاطمة عند الثانية 20 اما عند الثانية 25 يضيف حسين و ثم يوقف المؤقت.

الجواب

مرحلة التصميم

نحتاج في هذا المثال الى الادوات التالية :

الاداة	عنوانها	وظيفتها
Form	الاسماء	يضم مجموعة الادوات بدء تشغيل
Listbox	-	طباعة الاسماء
Textbox	بلا	عداد الوقت
timer	بلا	يحتوي الكود وينفذ المهمة

كتابة الكود

```
Public Class Form1

    Private Sub Form1_Load(sender As Object, e As EventArgs) Handles
MyBase.Load
        Timer1.Start()
    End Sub
    Private Sub Timer1_Tick(sender As Object, e As EventArgs) Handles
Timer1.Tick
        If TextBox1.Text = "*****" Then
            TextBox1.Text = ""
        Else
            TextBox1.Text = TextBox1.Text + "*"
        End If
        If TextBox1.Text = "*" Then
            ListBox1.Items.Add("احمد")
        ElseIf TextBox1.Text = "**" Then
            ListBox1.Items.Add("علي")
        ElseIf TextBox1.Text = "***" Then
            ListBox1.Items.Add("حسن")
        ElseIf TextBox1.Text = "****" Then
            ListBox1.Items.Add("فاطمة")
        ElseIf TextBox1.Text = "*****" Then
            ListBox1.Items.Add("حسين")
            Timer1.Stop()
        End If
    End Sub
```

التعليقات comments

هي عبارة عن جمل يكتبها المبرمج ولكنها لا تنفذ ولكن تستخدم في شرح الغرض من الكود أو لتمكين من يقرأ الكود لفهم الغرض منه ووظيفته ويتم تجاهل التعليقات من قبل معالج البرنامج compiler ولكنها مهمة في عملية تنظيم الكود وفهمه

وتبدأ التعليقات في الفيجوال بيسك بالفاصلة المفردة single quotation mark (') وكل النصوص التي تأتي بعدها إلى نهاية السطر هي عبارة عن تعليق لفهم الكود وغرضه ولا يتم تنفيذها.

والفيجوال بيسك دوت نت لا يدعم التعليقات لأكثر من سطر مرة واحدة ولكن إذا أردت ان تقوم بعمل مجموعة من السطور كتعليقات يجب تظليل تللك الأسطر ثم الذهاب إلى قائمة edit في الفيجوال بيسك ثم advanced ثم اختيار Comment Selection ولالغاء التعليقات اختيار Uncoment Slection

ويمكن كتابة التعليقات باللغة العربية أو الإنجليزية.

الاداة (ListBox)

هي اداة تستخدم لغرض كتابة قائمة من العناصر يستطيع المستخدم اختيار احد هذه العناصر ، وما يميزها عن اداة ComboBox ان العناصر فيها تكون ظاهرة للمستخدم.

من ابرز الخصائص التي يتم التعامل معها للاداة هي الخصائص التالية:

-انشاء عناصر للاداة و يتم من خلال الصيغة التالية:

```
ListBox1.Items.Add(" العنصر")
```

-اضافة عنصر جديد لقائمة خيارات مبنية مسبقا ويتم عن طريق الصيغة التالية:

```
ListBox1.Items.Insert( موقع العنصر الجديد , " العنصر")
```

-حذف عنصر من قائمة خيارات مبنية مسبقا ويتم عن طريق الصيغة التالية:

```
ListBox1.Items.Remove(" حذفه المراد العنصر")
```

الاداة ComboBox

هي اداة تستخدم لعرض خيارات ويمكن اختيار خيار واحد فقط من الخيارات الموجوده فيها .

من ابرز الخصائص التي يتم التعامل معها للاداة هي الخصائص التالية:

- انشاء عناصر للاداة ويتم من خلال الصيغة التالية :

ComboBox1.Items.Add("العنصر")

- اضافة عنصر جديد لقائمة خيارات مبنية مسبقا ويتم عن طريق الصيغة التالية :

ComboBox1.Items.Insert(موقع العنصر الجديد ,"العنصر")

- حذف عنصر من قائمة خيارات مبنية مسبقا ويتم عن طريق الصيغة التالية :

ComboBox1.Items.Remove("العنصر المراد حذفه")

- تحديد او اختيار عنصر من القائمة

ComboBox1.SelectedIndex = موقع العنصر

ملاحظة : موقع العنصر او ما يسمى (Index) هو رقم يشير الى موقع العنصر في القائمة ويبدأ العد من الصفر وليس من الواحد أي يكون موقع او (Index) العنصر الاول هو 0 والعنصر الثاني هو 1 وهكذا .

مثال :

صمم ونفذ برنامج لادخال معدل طالب وايجاد تقديره بضغطة زر واحدة علما ان التقدير (90-100 امتياز) ، (80-89 جيد جدا)،(70-79 جيد) ، (60- 69 متوسط) ، (50 – 59 مقبول) ، (0 – 49 ضعيف) ورسالة تنبيه اذا كان غير ذلك .

ملاحظة : لعرض التقدير استخدم ComboBox

الجواب

- **تصميم الواجهة**

- كتابة الكود

```
Public Class Form1
    Dim x As Integer

    Private Sub Form1_Load(sender As Object, e As EventArgs) Handles
MyBase.Load

' اضافة التقديرات الى الكومبو اثناء تحميل الفورم
        ComboBox1.Items.Add("امتياز")
        ComboBox1.Items.Add("جدا جيد")
        ComboBox1.Items.Add("جيد")
        ComboBox1.Items.Add("متوسط")
        ComboBox1.Items.Add("مقبول")
        ComboBox1.Items.Add("ضعيف")

    End Sub

    Private Sub Button1_Click(sender As Object, e As EventArgs)
Handles Button1.Click
        x = Val(TextBox1.Text)
        If x <= 100 And x >= 90 Then
            ComboBox1.SelectedIndex = 0
        ElseIf x <= 89 And x >= 80 Then
            ComboBox1.SelectedIndex = 1
        ElseIf x <= 79 And x >= 70 Then
            ComboBox1.SelectedIndex = 2
        ElseIf x <= 69 And x >= 60 Then
            ComboBox1.SelectedIndex = 3
        ElseIf x <= 59 And x >= 50 Then
            ComboBox1.SelectedIndex = 4
        ElseIf x <= 49 And x >= 0 Then
            ComboBox1.SelectedIndex = 5
        Else
            MsgBox("error")

        End If
    End Sub


End Class
```

أداة المؤقت Timer

الاداة Timer من اهم ادوات الفيجوال بيسك وهى اداة التحكم في الوقت . وهذه الاداة لا تضاف على ال Formولا تظهر في التنفيذ. اهم خصائص الاداة Timer هي الخاصية interval و توضع فيها قيم الوقت المراد عمل الاداة بها وفى هذه الاداة 1000 يعني ثانية واحدة و 2000 تساوي ثانيتين واذا اردنا كتابه ساعه ستكون 3600000 وهكذا.

ولتشغيل المؤقت يوضع الكود التالي في المكان الذي ترغب بتشغيل المؤقت عنده:

Timer1.Start()

مثال 1:

اضف 1 الى عنوان مربع النص كل ثانيتين وعندما يساوي العنوان 111111 اعد العمل من جديد ..

الجواب:

في البداية يجب ضبط خاصية **interval** عند 2000 . ثم التصميم بادراج textbox & timer وكتابة الكود التالي :

```
Public Class Form1
    Private Sub Form1_Load(sender As Object, e As EventArgs) Handles
MyBase.Load
        Timer1.Start()
    End Sub
```

```
    Private Sub Timer1_Tick(sender As Object, e As EventArgs) Handles
Timer1.Tick


        If TextBox1.Text = "111111" Then
            TextBox1.Text = ""
        Else
            TextBox1.Text = TextBox1.Text + "1"
        End If
    End Sub
End Class
```

البنية التكرارية

أحيانا نحتاج في تطبيقاتنا الى تكرار امر برمجي معين اكثر من مرة ، وهذا لا يعني إعادة تنفيذ نفس الامر فقط ، بل من الممكن اجراء بعض العمليات على كل عنصر ينفذ عليه التكرار.

لو فرضنا ان لدينا قاعدة بيانات تحتوي على بيانات العملاء واردنا إضافة ارقام الهواتف لهم جميعا ستكون عملية اسناد رقم الهاتف الى كل عميل طويلة وتستنزف وقتا وجهدا لاسيما اذا كان عدد العملاء كبيرا جدا ، لذلك تعتبر اليات التكرار او الحلقات افضل خيار لتكرار عملية معينة بقدر محدود مع إمكانية إضافة لمسة خاصة على كل عنصر تنفذ عليه العملية التكرارية.

او لنفترض اننا بصدد برمجة تطبيق صغير يخزن قيمة رقمية في متغير معين ويطلب من المستخدم تخمين هذه القيمة ، حتما نحن لانعرف عدد الاحتمالات سيقوم بها المستخدم لكي يصل الى نفس القيمة المخزنة عندنا لذلك يتوجب طرح نفس السؤال في كل مرة يدخل امستخدم فيها قيمة مخالفة للقيمة المخزنة.

توفر لنا لغة الفيجوال بيسك عدة أنواع من البنيات التكرارية هي :

الصيغة التكرارية الشرطية : For Next

تمكننا البنية التكرارية For … Next من تكرار امر برمجي معين لعدد مرات محدد ، حيث تسمح لنا بتحديد نقطتي بداية ونهاية التكرار مع إمكانية تحديد معامل التدرج ، وتكون صيغتها :

For count = start **To** end [**step**]

[statements]

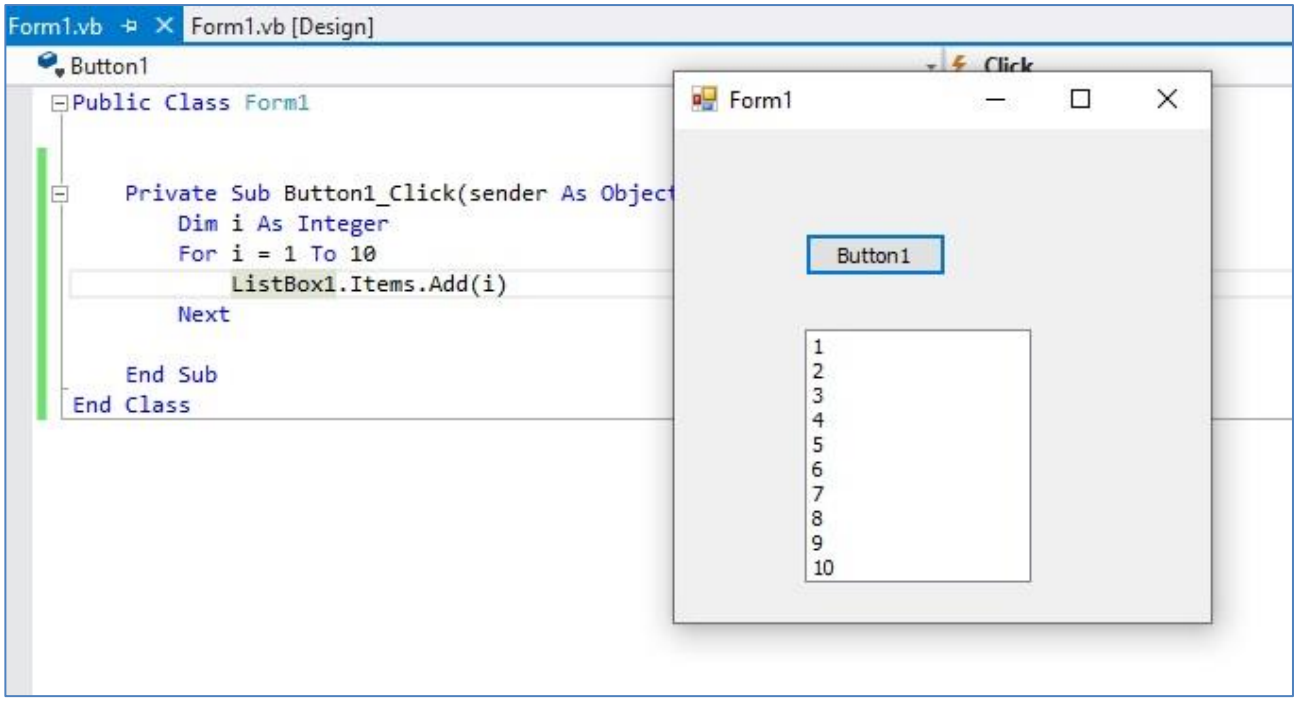
Next

حيث المتغير count هو العداد الي سيحدد عدد مرات تكرار الامر البرمجي ويبدا من القيمة الرقمية start وينتهي بالقيمة الرقمية end مع إمكانية تحديد معمل التدرج (مقدار زيادة العداد في كل تكرار ، والقيمة الافتراضية = ١) في الامر **step** . بعد ذلك نقوم بكتابة الامر البرمجي الذي نريد تكراره ويمكننا الخروج من التكرار متى ما اردنا بكتابة Exit For ، وفي ختام البنية التكرارية نضع Next متبوعة باسم العداد (اختياري)

مثال :

```
Dim i As Integer
        For i = 1 To 10
            ListBox1.Items.Add(i)
        Next
```

قمنا بالإعلان عن متغير رقمي اسمه I ثم استعملناه في عملية التكرار ليبدا من العدد 0 وينتهي بالعدد 10 مع الزيادة بواحد عند كل تكرار ، وفي كل مرة يقوم التكرار بإضافة قيمة العداد الى ListBox1 عن الضغط على Button.



ويمكن التعريف عن المتغير واسناد القيمة له في بنية التكرار:

For I as integer = 1 to 10

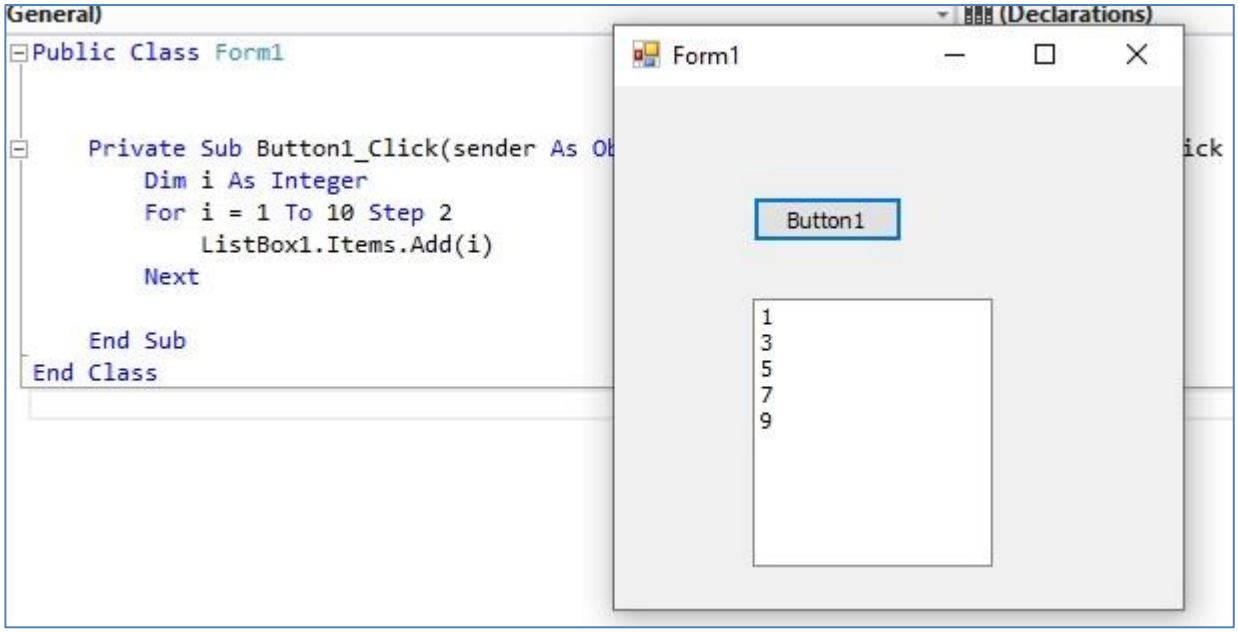
Listbox1.items.add(i)

Next

ملاحظة : الامر step في المثال أعلاه لا دور له لان نسبة الزيادة الافتراضية هي 1 ، ولو تم تغيير قيمته الى 2 مثلا "

```
Dim i As Integer
        For i = 1 To 10 Step 2
            ListBox1.Items.Add(i)
        Next
```

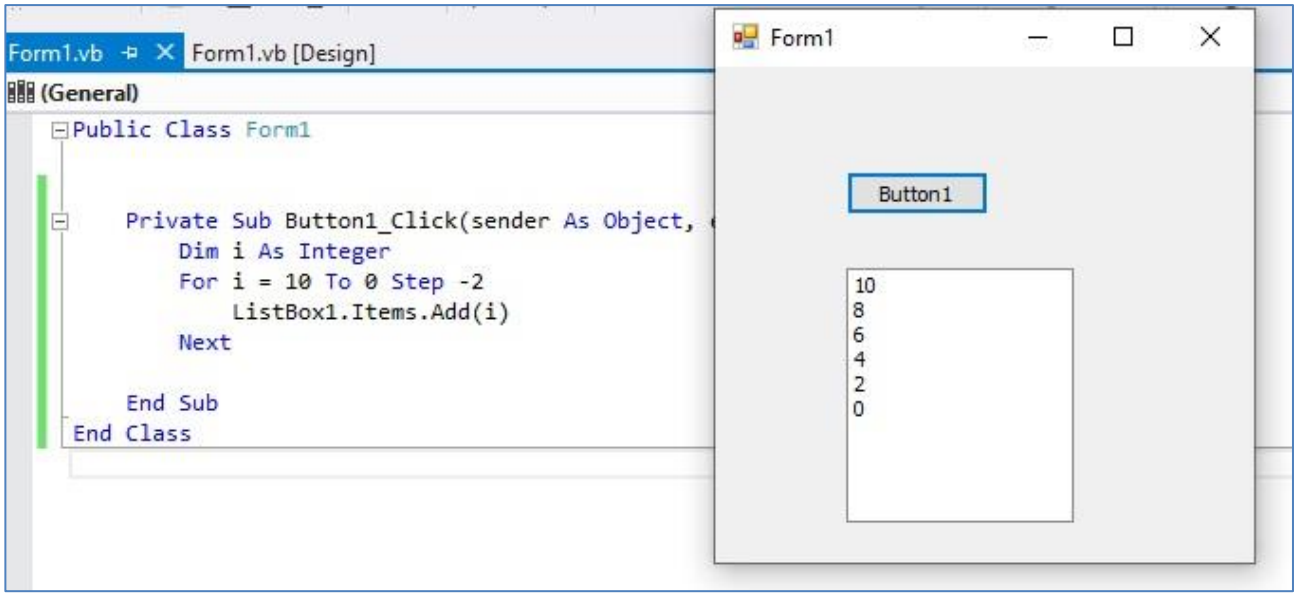
يكون الناتج كالتالي :



ويمكن جعل قيمة التدرج سلبية (تراجعية) وهذا ما يسمى بالتكرار المعكوس كالتالي :

```
Dim i As Integer
        For i = 10 To 0 Step -2
            ListBox1.Items.Add(i)
        Next
```

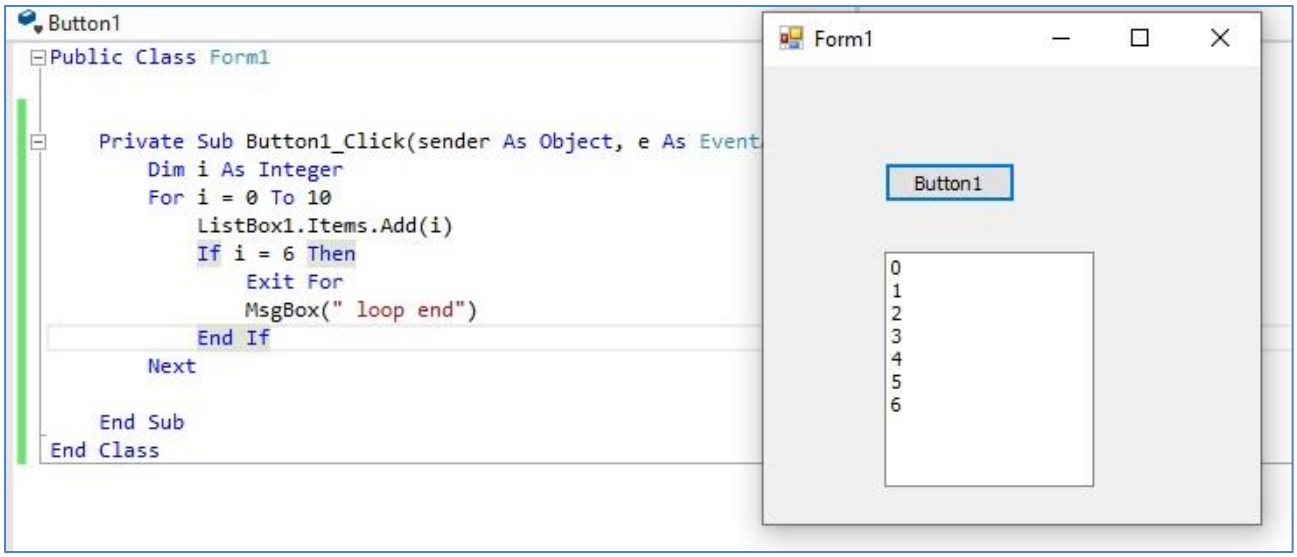
ليكون الناتج كالتالي :



الخروج من التكرار :

يمكن إيقاف التكرار عند توافر شرط معين باستخدام الامر (Exit For) كما في المثال التالي الذي نضع فيه شرط توقف التكرار عند وصول قيمة المتغير (العداد) الى 6 وارسال رسالة اشعار بإنهاء التكرار :

```
Dim i As Integer
        For i = 0 To 10
            ListBox1.Items.Add(i)
            If i = 6 Then
                Exit For
                MsgBox(" loop end")
            End If
        Next
```



عبارة الشرط (Select case)

تعتبر هذه الاداة من اهم الادوات واكثرها استخداما :

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 نستخدم عبارة Select Case وتكون صيغتها العامة ما يلى : تبدأ العبارة بـ Select Case يليها اسم المتغير او التعبير الذى سيتم اختباره .تأتي بعد ذلك الاحتمالات Case بعد كل منها احدى قيم المتغير الذى ستتم مقارنته ثم يعقبها التعليمات التى ستنفذ إذا كان الشرط صحيحًا أو كان المتغير بهذه القيمة . واخيرًا يأتى Case else ومعناها إذا كان المتغير لا يساوي أيًا من القيم السابقة أو إذا لم يكن الشرط صحيحًا فإن التعليمات التى تلى Elseهي التى تنفذ :

Select case المتغير او التعبير الذي سوف يتم اختباره

Caseالحالة (القيمة) الاولى

Case الحالة (القيمة) الثانية

.

.

.

Case else

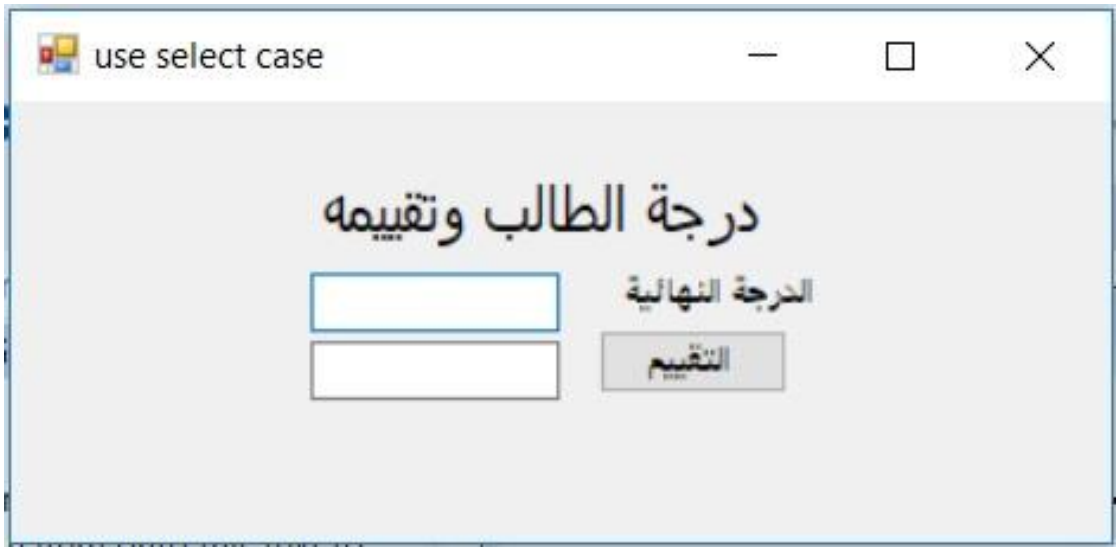
End select

مثال :

صمم ونفذ برنامج لادخال درجة طالب بمادة واحدة وايجاد تقديره بضغطة زر واحدة علما ان التقدير (90-100 امتياز) ، (80-89 جيد جدا)،(70-79 جيد) ، (60- 69 متوسط) ، (50 – 59 مقبول) ، (0 – 49 ضعيف) ورسالة تنبيه اذا كان غير ذلك (استخدم عبارة الشرط Select case).

الجواب

- **التصميم**



- Code in Button1:

```
Private Sub Button1_Click(sender As Object, e As EventArgs) Handles
Button1.Click
  Dim aver As Integer
    aver = Val(TextBox1.Text)
    Select Case aver
      Case 90 To 100
        TextBox2.Text = "امتياز"
      Case 80 To 89
        TextBox2.Text = "جدا جيد"
      Case 70 To 79
        TextBox2.Text = "جيد"
      Case 60 To 69
        TextBox2.Text = "متوسط"
      Case 50 To 59
        TextBox2.Text = "مقبول"
      Case 0 To 49
        TextBox2.Text = "ضعيف"
      Case Else
        MsgBox("خاطئ الرقم")
    End Select
  End Sub
End Class
```

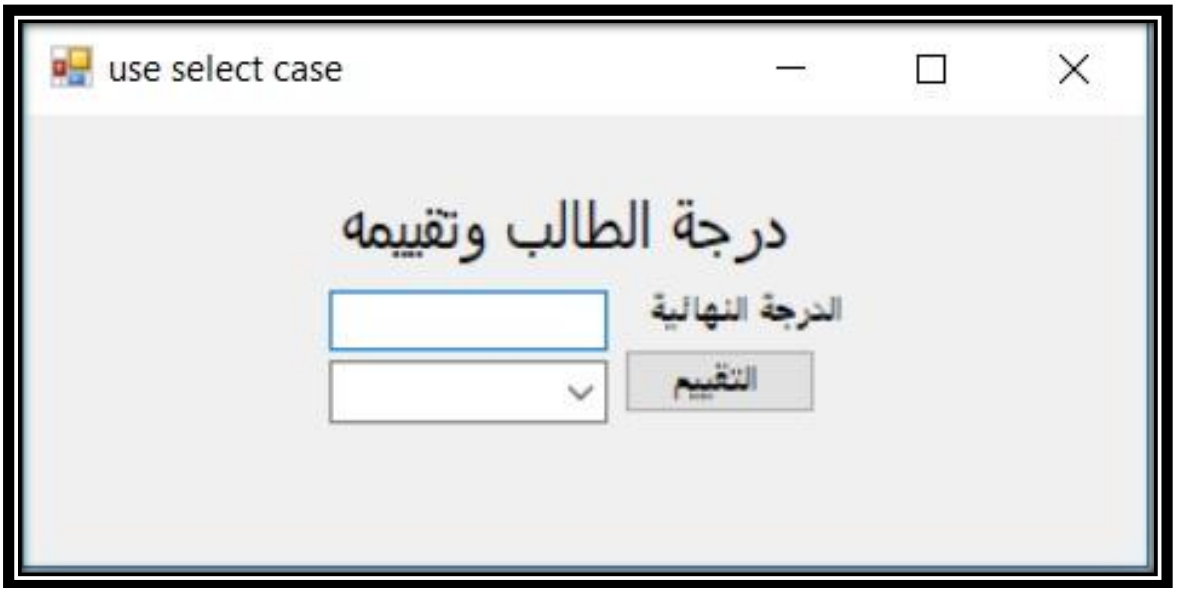
مثال :

صمم ونفذ برنامج لادخال درجة طالب بمادة واحدة وايجاد تقديره بضغطة زر واحدة علما ان التقدير (90-100 امتياز) ، (80-89 جيد جدا)،(70-79 جيد) ، (60- 69 متوسط) ، (50 – 59 مقبول) ، (0 – 49 ضعيف) ورسالة تنبيه اذا كان غير ذلك (استخدم عبارة الشرط Select case)

ملاحظة : لعرض التقدير انشئ قائمة خيارات combobox تحتوي حالة التقدير .

الجواب:

التصميم



الكود

```
Public Class Form1

    Private Sub Form1_Load(sender As Object, e As EventArgs)
Handles MyBase.Load
        ComboBox1.Items.Add("امتياز")
        ComboBox1.Items.Add("جدا جيد")
        ComboBox1.Items.Add("جيد")
        ComboBox1.Items.Add("متوسط")
        ComboBox1.Items.Add("مقبول")
        ComboBox1.Items.Add("ضعيف")

    End Sub

    Private Sub Button1_Click(sender As Object, e As
EventArgs) Handles Button1.Click
        Dim aver As Integer
        aver = Val(TextBox1.Text)
        Select Case aver
            Case 90 To 100
                ComboBox1.SelectedIndex = 0
            Case 80 To 89
                ComboBox1.SelectedIndex = 1
            Case 70 To 79
                ComboBox1.SelectedIndex = 2
            Case 60 To 69
                ComboBox1.SelectedIndex = 3
            Case 50 To 59
                ComboBox1.SelectedIndex = 4
            Case 0 To 49
                ComboBox1.SelectedIndex = 5
            Case Else
                MsgBox("الرقم خاطئ ، اعد المحاولة")
        End Select
    End Sub
End Class
```

البرمجة

سلسلة من آلاف او ملايين الاوامر التي تطلب من الحاسوب (computer) اجراء عمليات معينة مثل عرض معلومات او اجراء حسابات او تخزين بيانات وغيرها . وتتم البرمجة عن طريق لغات برمجة ذات مستوى معين تقوم عن طريق المترجم (compiler) او المفسر(assembler) على تحويل الاوامر التي كتبها المبرمج (الانسان) الى اوامر بلغة الالة (0-1) ليفهمها الكومبيوتر وينفذ المطلوب منه.

البرمجيات هي بمثابة الروح من الجسد في النظام الحاسوبي وهي في توسع دائم وازدياد في التعقيد والمتطلبات والمهام التي تقوم بتنفيذها كبرمجة تطبيقات web وبرمجة تطبيقات desktop وبرمجة تطبيقات mobile وبرمجة قواعد البيانات والبرمجة المدمجة والتي تختلف في لغات البرمجة المستخدمة .

لغات البرمجة

لغات البرمجة هى مجموعة من الأوامر والتعليمات لجهاز الكمبيوتر لعمل مهمة معينة فهى نقطة الوصل بين الأنسان والكمبيوتر. وعلى خلاف لغات البشر التى يمكن فهمها حتى مع بعض الاخطاء ، فإن لغات البرمجة ليست بتلك المرونة فمجرد خطأ صغير فى البرنامج مثل نسيان (؛) او اي خطأ بسيط فأن البرنامج لن يعمل. وتشير كلمة لغات البرمجة غالبًا إلى اللغات عالية المستوى High-Level Languages وهى اللغات الأقرب إلى لغة البشر وتتكون من كلمات يفهمها البشر مثل If , While وغيرها امثلة على هذه اللغات FORTRAN , Python , C++ وغيرها هذه اللغات عالية المستوى وان كانت اقرب إلى لغة البشر فهى معقدة بالنسبة للغات التى يفهمها الكمبيوتر فعليًا وهى لغة الآلة او Machine Language وهى مكونة من صفر او واحد .

مراحل بناء النظام البرمجي

بناء النظام البرمجي ليس مجرد كتابة شفرة، وإنما هي عملية إنتاجية لها عدة مراحل أساسية وضرورية للحصول على المنتج، وهو البرنامج بأقل كلفة ممكنة وأفضل أداء محتمل. يطلق على هذه المراحل اسم دورة حياة النظام البرمجي (Software Lifecycle). وهذه المراحل هي:

1. كتابة وثيقة الشروط الخارجية والداخلية

وثيقة الشروط الخارجية يتم أخذها من الزبون وتحتوي الوثيقة على متطلبات الزبون في ما يخص مواصفات البرنامج الذي يجب إنشاؤه.

2. التحليل

في هذه العملية تجمع المعلومات بدقة ثم تحدد المتطلبات والمهام التي سيقوم بها البرنامج، وتوصف هذه المهام بدقة تامة، كما تدرس الجدوى المرجوة من البرنامج.

3. التصميم

تصميم البرمجيات هي مرحلة من مراحل دورة حياة النظام، تساعدنا في تحديد كيفية حل المشكلة "كيف سنحل المشكلة؟"، والتخطيط للتوصل إلى حلول للمشكلة، والدخول في تفاصيل النظام.

4. الترميز(كتابة الكود)

تحول الخوارزميات والمخططات Diagrams التي تم انتاجها في مرحلة التصميم إلى إحدى اللغات البرمجية، وذلك لانتاج برنامج او نظام قابل للاستخدام من قبل الزبون، يلبي احتياجتة الموضحة في وثيقة الشروط.

5. الاختبار والتكاملية

تجمع الكتل مع بعضها ويختبر النظام للتأكد من موافقته لجدول الشروط والمواصفات، وخاصة إذا كانت الكتل قد كتبت من قبل عدة أعضاء في الفريق.

6. التوثيق

وهي مرحلة هامة من مراحل بناء النظام البرمجي حيث يتم توثيق البناء الداخلي للبرنامج؛ وذلك بغرض الصيانة والتطوير. يفضل عادة أن يترافق التوثيق مع كل مرحلة من المراحل السابقة واللاحقة.

7. الصيانة والتطوير

إن هذه المرحلة هي المرحلة الأطول في حياة النظام البرمجي لبقاء النظام قادراً على مواكبة التطورات والمعدات الحديثة، جزء من هذه المرحلة يكون في تصحيح الأخطاء، والجزء الآخر يكون في التطوير وإضافة تقنيات جديدة.

مايكروسوفت فيجوال ستوديو (Microsoft Visual Studio)

هي بيئة التطوير المتكاملة الرئيسية من مايكروسوفت والتي تتيح برمجة واجهة المستخدم الرسومية والبرامج النصية إلى جانب ويندوز فورم ومواقع ويب وتطبيقات ويب وخدمات وب مدعومة ب مايكروسوفت ويندوز وويندوز موبايلوإطار عمل دوت نت ومايكروسوفت سيلفرلايت.

يحتوى فيجيوال استوديو على محرر أكواد يدعم تقنية انتليسنس واعادة كتابة الكود، ويحتوى أيضا على مترجم يكشف أخطاء وقت التشغيل ومفسر يكشف الأخطاء الاملائية في الأكواد ويحتوى أيضا على مصمم نماذج لبناء واجهة مستخدم رسومية ومصمم ويب. يدعم فيجيوالـ استوديو العديد من لغات البرمجة مثل:

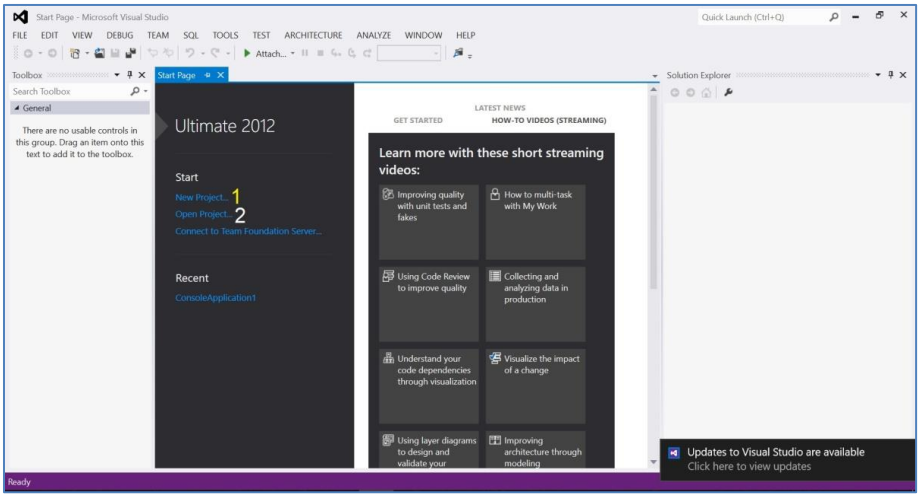
- Microsoft visual c++
- Microsoft visual c#
- Visual basic
- JavaScript

فيجوال بيسك Visual Basic

هي بيئة تطوير ولغة برمجة من مايكروسوفت تستند إلى لغة البيسك الشهيرة. وهي تصنف ضمن لغات البرمجة بالكائنات. منذ أن بدأت مايكروسوفت في اصدار الفيجوال بيسك وهي تلاقي نجاحا باهرا وشعبية لا بأس بها بين المبرمجين نظرا لسهولتها الشديدة في مقابل التعقيد الشديد الذي يواجهه أي مبرمج يسعى لبرمجة ويندوز باستخدام السي أو عموما تناسب الفيجوال بيسك تطبيقات قواعد بيانات والتطبيقات المخصصة للشركات الصغيرة وبرامج الحسابات وهي مريحة وسهلة وتؤدي الغرض بالإضافة إلى أنها تسمح للمبرمج بالتركيز على حل المشكلة فغالباً ما لا يواجه صعوبات فنية أثناء كتابة برنامج بالفيجوال بيسك. ولتشغيل برنامج فيجوال ستوديو وانشاء مشروع بلغة فيجوال بيسك عليك اتباع الخطوات التالية :

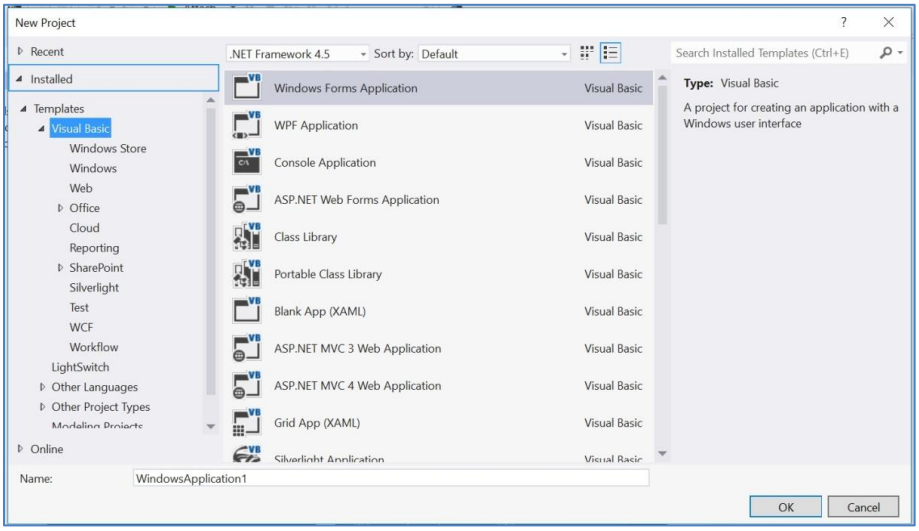
Start menu ⟶ All programs ⟶ Microsoft Visual Studio2012 ⟶ Visual Studio2012

عندها ستفتح النافذة الموضحة بالشكل التالي ، فاذا اردت فتح مشروع سابق تم خزنه فيجب اختيار خطوة (2) (open project) ، واذا اردت فتح مشروع جديد اتبع الخطوة رقم (1) (new project) ، كما موضح في الشكل التالي:



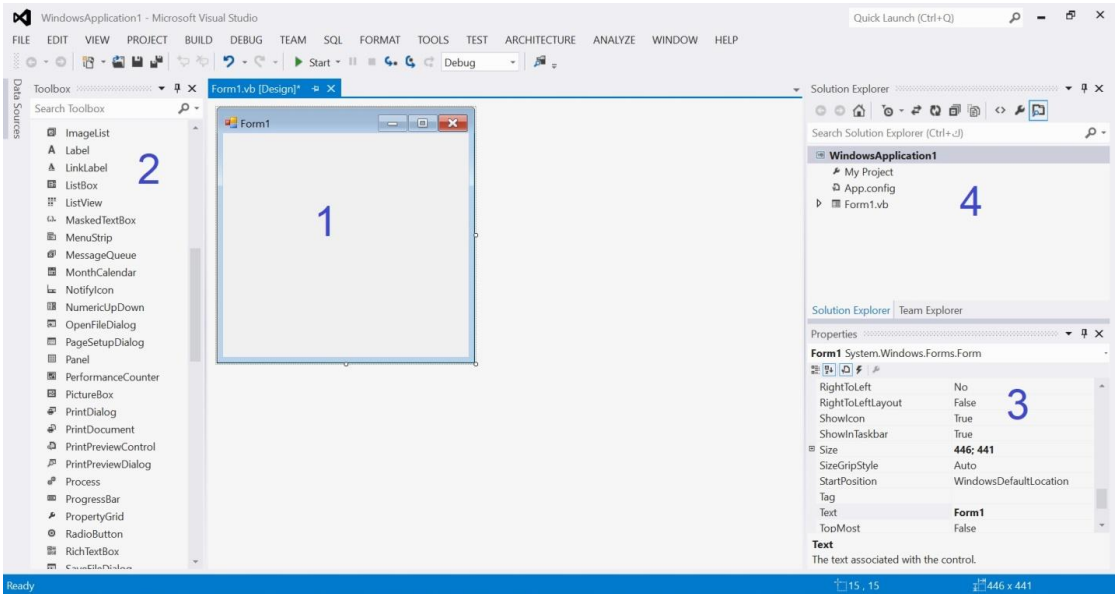
وعند اختيار مشروع جديد سوف تظهر النافذة الموضحة في شكل(2) حيث توجد على الجانب الايسر قائمة لغات البرمجة التي يدعمها اصدار فيجوال ستوديو المستخدم ، وفي الوسط نوع الملف الذي ستتعامل معه من لغة البرمجة . ولانشاء تطبيق بلغة الفيجوال بيسك يجب الاختيار كما موضح في الشكل التالي :

Visual Basic ⟶ Windows Form Application ⟶ OK



التعامل مع المشروع :

عند فتح مشروع جديد بواسطة لغة فيجوال بيسك تكون واجهة البرنامج كما في شكل (3)حيث تتكون من خمسة اجزاء رئيسية تتحكم بالمشروع وهذه الاجزاء هي :



1- Form وهو النموذج الذي سوف يتم العمل عليه.

2- Toolbox وهي الادوات التي نستعين بها لاداء مهام المشروع.

3- Properties خصائص العنصر الذي نتعامل معه.

4- Solution Explorer المكان الذي تستعرض في ملفات العمل المختلفة.

5- Code editor محرر الشفرة وهي النافذة التي يتم بها كتابة الشفرة وتظهر بالنقر المزدوج على العنصر المراد كتابة الشفرة فيه.

عملي 1- التعامل مع اجزاء واجهة البرنامج واضافة مجموعة من الادوات وتغيير خصائصها عن طريق نافذة الخصائص.

الادوات Tools

الأدوات تعرّف بأنها : " أجزاء برامج جاهزة للاستخدام " ، أي أنها أعدت مسبقاً من قبل مبرمجين لتوفر على المبرمج الوقت والجهد، فما فائدة الأدوات؟ إن الفائدة الرئيسية للأدوات أنه عند تصميم البرنامج لا يبدأ المبرمج كتابته من الصفر، وإنما هناك مجموعة من الأدوات الموجودة في "فيجول بيسك.نت" والتي يمكنك استخدامها في البرنامج، فمثلاً عندما تريد أن تقوم بعمل نافذة، فأنت لا تكتب أي تعليمات خاصة بإنشائها، وإنما تقوم بالتالي:

1. تصدر أمراً إلى "فيجول بيسك.نت" بإنشاء نافذة جديدة.
2. تبدأ بالتحكم في شكل وطريقة عمل هذه النافذة.
3. عندما تريد من المستخدم إدخال بيانات فبدلاً من كتابة جزء من برنامج يقوم بهذا العمل ، فإنك تقوم بوضع أداة النص على نافذة البرنامج التي تمكن المستخدم من إدخال البيانات.

ولذلك تعد عملية تصميم واجهات البرنامج واختيار الأدوات المناسبة جزءاً مهماً من مراحل كتابة البرامج بواسطة فيجول بيسك.نت.

في مايلي نستعرض بعض اهم الادوات في فيجوال بيسك واكثرها استخداما :

1- زر الأمـر Button : وظيفته تنفيذ أوامر المستخدم مثلا هذا الزر نعطيه أمر يعتبر كود نكتبه بداخله و أثناء الضغط عليه ينفذ الأمر.
2- اداة العنوان Label : نقوم بكتابة نصوص فيها و لا يمكن تعديل النصوص من قبل المستخدم لكونها مبرمجة من طرف المبرمج في حين يمكن للمبرمج تعديل هذه النصوص أو يجعلها ثابتة.
3- صندوق النص Textbox : نستخدمها في كتابة نص و إجراء التعديلات عليها و لدينا الحرية في استخدامها.
4- اداة الصورة Picture : وتتيح لنا عرض الصور من خلالها.
5- الاطار Frame : توضع بداخله أدوات لكن هذه الأدوات لا تتحرك في حالة تحرك الإطار في تبقى ثابتة و لا يتغير مكانها داخل الإطار.
6- الاداة CheckBox لعرض اختيارات للمستخدم ويمكن تختار اكثر من خيار.
7- الاداة RadioOption لعرض اختيارات للمستخدم ولا يمكن اختيار اكثر من خيار.
8- الاداة Timer يكرر عمل معين كل مدة زمنية وهذه الأداة لا تظهر وقت التنفيذ.

9- الاداة ComboBox لعرض قائمة من الخيارات للمستخدم حيث يمكن ان يختار منها دون ان يكتب شيء.

10- الاداة ListBox لعرض قائمة من الخيارات للمستخدم حيث يمكن ان يختار منها و يمكن ان يكتب فيه.

لاضافة الاداة الى Form يتم النقر المزدوج على الاداة بشريط الادوات او عند السحب والافلات.

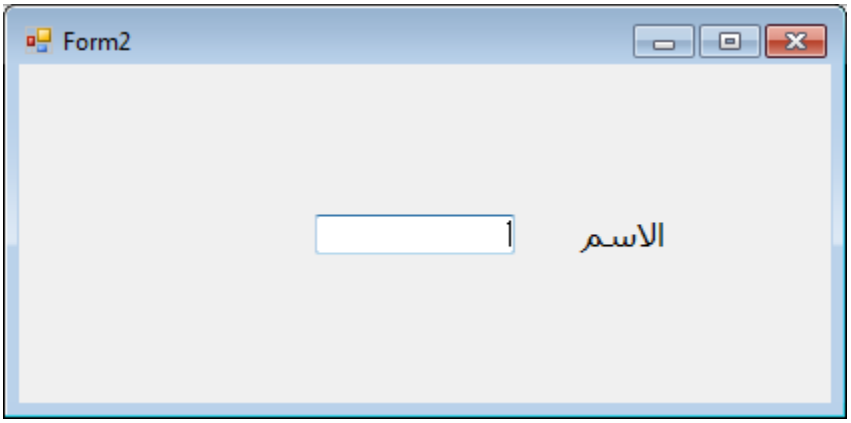
الخصائص Properties

لكل أداة من أدوات البرمجة في فيجول بيسك خصائص تحدد شكلها ، مثل : اللون ، ونوع الخط ، وحجمه ، وغيرها من الخصائص التي يحددها المستخدم خلال التصميم. وهناك خصائص عامة مشتركة بين الأدوات ، وأخرى خاصة بكل أداة .

من ابرز الخصائص التي يمكن للمبرمج التحكم بها قبل البدء بالبرمجة واكثرها استخداما هي

1. name (الاسم البرمجي) تحدد من هنا اسم الكائن الذي ستستخدمه في كتابة أكواد هذا الكائن.
2. Text النص الذي سيظهر على الأداة (اسمها الظاهري).
3. Alignment تحدد من هنا موقع الكتابة من الاداة.
4. Appearance تحدد من هنا شكل الأداة.
5. BackColor تحدد لون خلفية العنصر.
6. ControlBox اذا كان خيارك False فسوف تختفي أيقونة البرنامج والأزرار (اغلاق-تصغير-تكبير) اما ان كان True فسوف تظهر.
7. Font من خلالها نستطيع التحكم بالخط ومواصفاته.
8. PasswordChar هذه الخاصية تستعمل في كتابة كلمات السر حيث تظهر كلمة السر على شكل نجوم مثلا , فاذا اردت استخدامها اكتب داخل هذه الخاصية (*).
9. Visible اخفاء الأداة عن المستخدم (False)واظهارها(True) .

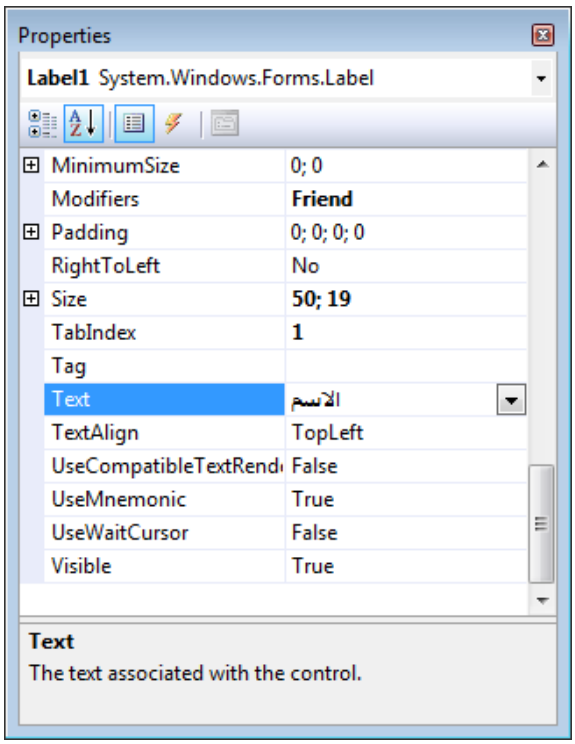
مثال 1 :
لتصميم واجهة من خلالها يقوم المستخدم بإدخال الاسم كما في الصورة التالية :



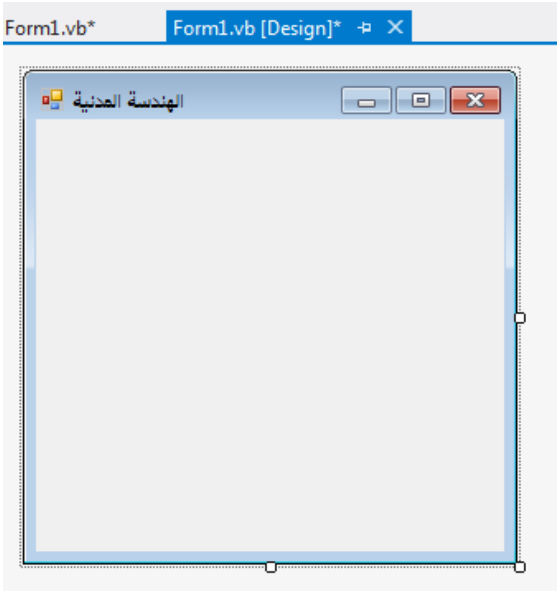
نتبع الخطوات التالية :

1. من مربع الأدوات نختار مربع نص TextBox ونضعه داخل الشاشة.
2. نضيف أداة عنوان Lable ونغير من خصائصها خاصية النص (text) الى (الاسم) .

كما في الصورة ادناه :



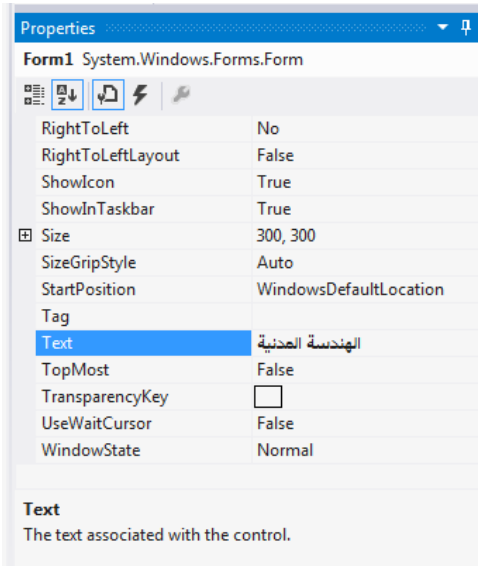
مثال 2 : لتغيير عنوان Form الى (الهندسة المدنية) كما في الصورة التالية :



نتبع الخطوات التالية :

1- النقر على Form

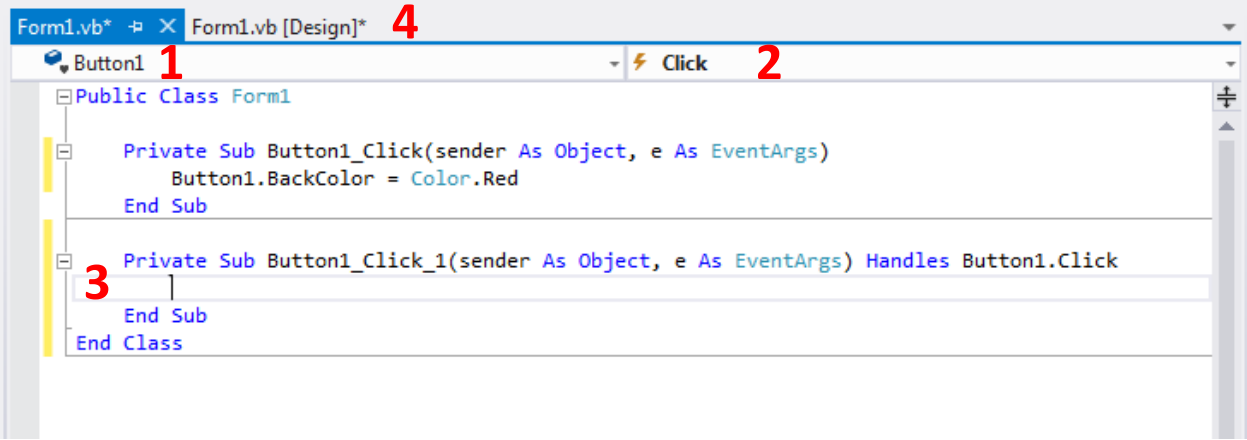
2- الذهاب الى الخصائص Properties واختيار text وتغييرها الى القيمة الجديدة بمجرد الضغط على قيمتها القديمة وكتابة القيمة الجديدة كما في الصورة التالية :



ملاحظة / يقوم برنامج فيجول بيسك بإعطاء أسماء تلقائية لكل أداة تقوم برسمها، فعندما ترسم أداة تسمية لأول مرة فإن فيجول بيسك يعطيها اسم (Label1) ، وعندما ترسم أداة التسمية مرة أخرى في نفس النموذج فإن فيجول بيسك يعطيها اسم : (Label2) وهكذا لبقية الأدوات.

كتابة الكود :

تطرقنا في ما سبق الى التحكم بخصائص الادوات بصورة مباشرة من تبويب الخصائص (Properties) وبحسب الخصائص المتاحة , لكن هذا التحكم لايفي بالغرض في كل الاوقات واحيانا يكون عبئا على البرنامج ويصيبه بالجمود , لذلك يجب علينا الانتقال الى التحكم بالخصائص عن طريق اكواد برمجية خاصة , وهي الانطلاقة الحقيقية في عالم البرمجة , وتتم كتابة الاكواد عن طريق النقر المزدوج Double click على الاداة المراد منها تنفيذ الامر لتنقلنا الى نافذة تحرير الاكواد الموضحة في الصورة التالية :



وتتكون من الاجزاء التالية :

1- الاسم البرمجي (Name) للاداة التي سيتم كتابة الكود فيها.
2- الطريقة التي من خلالها يتم تنفيذ الامر , كأن تكون Click or Double click او غيرها.
3- المساحة التي يتم كتابة الكود فيها .

عند الانتقال الى نافذة تحرير الاكواد قد يرتبك المبتدئ بالبرمجة ويحتار بكيفية العودة الى واجهة النموذج التصميمي , لا تقلق يمكنك ذلك بمجرد الضغط على الرقم (4) في الصورة السابقة .

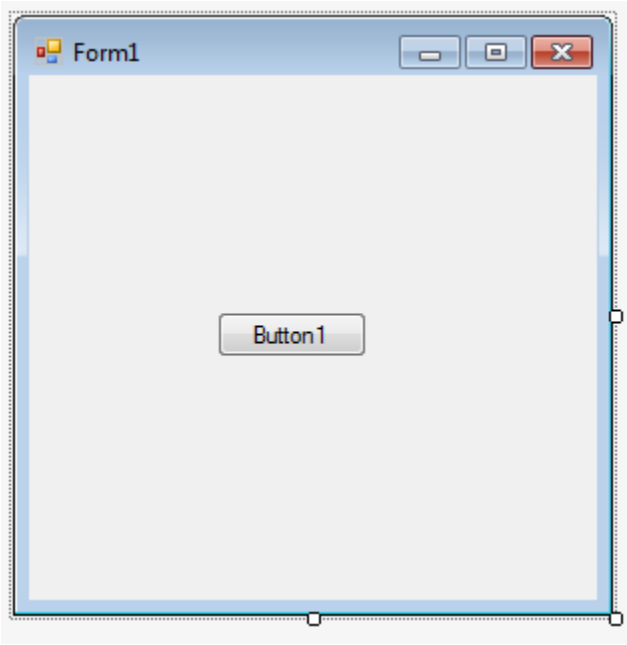
ولنبرهن كلامنا بمثال :

مثال 3 :

صمم برنامجا يغير عنوان الـ (Form) الى (الهندسة المدنية) عند النقر على (Button).

الحل :

بداية نصمم الواجهة كما في الصورة التالية :



ثم ننقر على الاداة التي نريد منها تنفيذ الامر , وهنا الاداة Button1 لتنقلنا الى نافذة محرر الاكواد , ليكون الكود بالصيغة التالية :

القيمة الجديدة لخاصية = الخاصية المراد تغييرها . الاداة المراد اجراء الامر او التغيير عليها

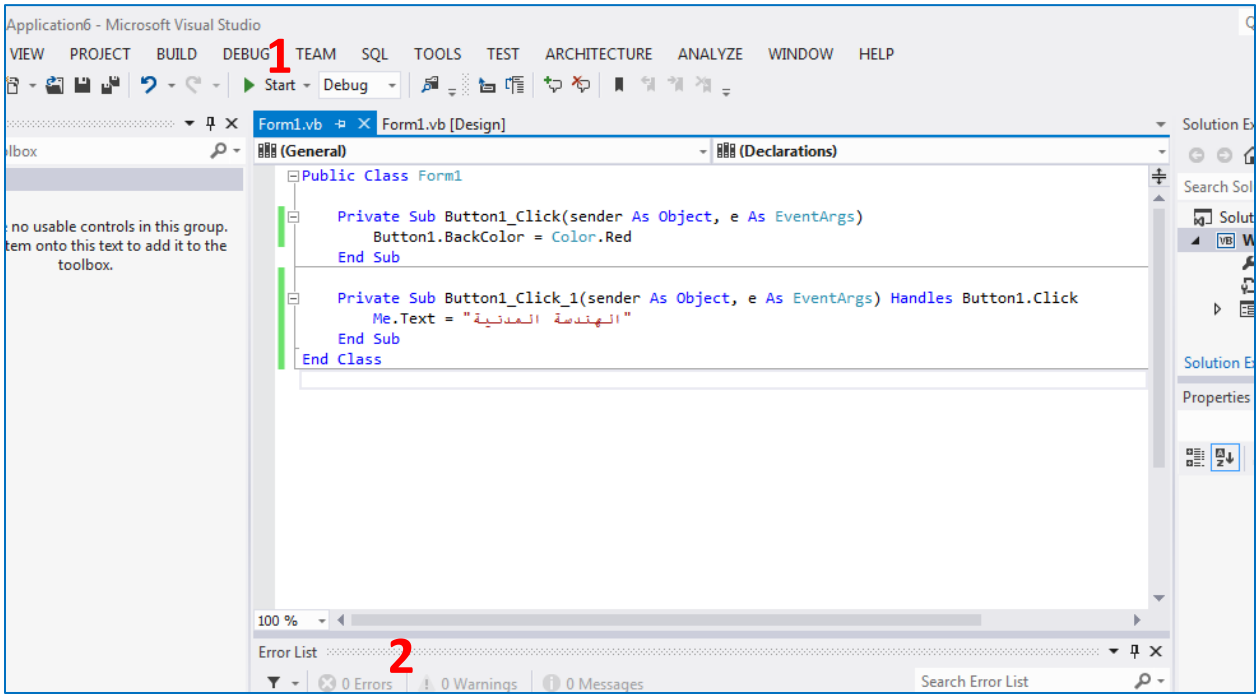
وفي مثالنا اعلاه يكون الكود :

```
Me . text = " الهندسة المدنية "
```

ملاحظة1 / Form1 يعبر عنه بالكود بكلمة me

ملاحظة2 / اي قيمة نصية يجب ان نضعها بين علامتي تنصيص " ".

ليكون الشكل النهائي للكود في نافذة تحرير الاكواد :



وبعد التاكد من سلامة الكود الخاص بك تنفذ البرنامج عن طريق الضغط على رقم 1 start .

ملاحظة : في حال وجود خطا في الكود الخاص بك سوف يتم تنبيهك قبل تنفيذ البرنامج وذلك بوضع خط ملون متعرج اسفل موقع الخطا مع تغيير عدد الاخطاء Errors اسفل الشاشة الموضح بالرقم 2.

عملي : تنفيذ تغيير خصائص لادوات متعددة عن طريق الاكواد.

روابط المقارنة comparison operators

هي روابط تستخدم من اجل مقارنة قيمتين وتحديد نوع العلاقة بينهما (اكبر من , اصغر من , تساوي ... الخ) , ونتيجة المقارنة تكون منطقية Boolean اي تقبل فقط قيمتين : اما True او False وفي الجدول ادناه ابرز الروابط المستخدمة :

الرابط	وظيفته
>	اكبر من
<	اصغر من
=	يساوي
<>	لايساوي
>=	اكبر من او يساوي
<=	اصغر من او يساوي

وسنستعرض امثلة على الاكواد الخاصة بالروابط اعلاه:

`Dim x As Boolean = 6 > 7`	النتيجة خاطئة
`Dim y As Boolean = 6 < 7`	النتيجة صحيحة
`Dim z As Boolean = 6 <> 7`	النتيجة صحيحة
`Dim n As Boolean = 6 >= 7`	النتيجة خاطئة
`Dim m As Boolean = 6 <= 7`	النتيجة صحيحة
`Dim r As Boolean = 6 = 7`	النتيجة خاطئة

روابط اسناد القيمة Assignment operators

رموز تستخدم من اجل تخزين واسناد قيمة الى متغير ما , ولعل ابرز روابط اسناد القيمة هو = .

وسندرج في ما يلي قائمة لبعض الرموز المستخدمة من اجل تخزين القيم وحسابها في نفس الوقت:

الرابط	وظيفته
=	الجمع
^=	اسناد القيمة بعد حساب القوة (الاس)
*=	اسناد القيمة بعد حساب الضرب
/=	اسناد القيمة بعد حساب القسمة
+=	اسناد القيمة بعد حساب الجمع
-=	اسناد القيمة بعد حساب الطرح
&=	اسناد القيمة بعد دمجها بقيمة معينة

ولتوضيح عمل الروابط اعلاه ناخذ الامثلة التالية:

`Dim mywork As Integer = 6`	تخزين القيمة 6 في المتغير Mywork
`Mywork ^=2`	حساب قيمة المتغير للاس اثنين
`Mywork *=2`	حساب قيمة المتغير مضروبا في 2
`Mywork /= 2`	حساب قيمة المتغير مقسوما على 2
`Mywork += 2`	حسابا قيمة المتغير مجموعا مع 2
`Mywork -= 2`	حساب قيمة المتغير مطروحا منه 2
`Mywork &=7`	سيتم دمج قيمة المتغير مع القيمة 7 ليصبح الناتج67

العبارات الشرطية Conditional Statements

وتدعى أيضا جمل السيطرة Control Statements الهدف من هذه الجمل هو إجراء عملية التحكم بطريقة سير وتنفيذ الايعازات في البرنامج , ومن هذه الجمل:

1. (If then) statement .
2. (If – else) statement .
3. (If – else If) statement .
4. (Select Case) statement .

تستخدم الجمل الشرطية معاملات المقارنة (comparison Operators) والمعاملات المنطقية (Logical Operators) في تكوين التعابير التي تحتوي على شروط.

المعاملات المنطقية :

- المعامل OR (أو) : وهو المعامل المنطقي الذي يستخدم للربط بين شرطين او اكثر ويكفي تحقق احد الشروط لتتحقق الهدف .
- المعامل AND (و) : وهو المعامل المنطقي الذي يستخدم للربط بين شرطين او اكثر ويشترط تحقق جميع الشروط لتحقق الهدف.

X	Y	X OR Y	X AND Y
FALSE	FALSE	FALSE	FALSE
FALSE	TRUE	TRUE	FALSE
TRUE	FALSE	TRUE	FALSE
TRUE	TRUE	TRUE	TRUE

غالبا مايستخدم في جمل اجابة الشرط مربع الرسالة Message Box الذي يعرض رسالة لها عنوان معين ومحتوى وزر محدد او اكثر حسب المعنى المطلوب ايصاله . وتكون الصيغة العامة لمربع النص :

MsgBox ("عنوان الرسالة" , نوع الرسالة , "محتوى الرسالة")

العبارات الشرطية

1- عبارة If then وتكون الصيغة العامة لها :

If الشرط then

الايعاز المراد تنفيذه عند تحقق الشرط (جملة برمجية او اكثر)

End if

2- عبارة If - Else وصيغتها العامة:

If الشرط then

الايعاز المراد تنفيذه عند تحقق الشرط (جملة برمجية او اكثر)

Else

الايعاز المراد تنفيذه عند عدم تحقق الهدف (جملة برمجية او اكثر)

End if

3- عبارة If - Else if وصيغتها العامة:

If الشرط then

الايعاز المراد تنفيذه عند تحقق الشرط السابق (جملة برمجية او اكثر)

Else If الشرط then

الايعاز المراد تنفيذه عند تحقق الهدف السابق (جملة برمجية او اكثر)

Else

الايعاز المراد تنفيذه عند عدم تحقق احد الاهداف السابقة (جملة برمجية او اكثر)

End if

مثال : في المثال التالي سنتناول النوع الاول من انواع If كما في برنامج الرقم السري فاذا كان الرقم السري المدخل مطابق الذي حددناه مسبقا فيجب تغيير لون خلفية مربع النص الى الاحمر وكذلك اظهار الصورة المخفية , كما في الشكل التالي:

ولتصميم البرنامج نتبع الجدول الاتي :

الاداة	الخاصية	القيمة الجديدة
Form1	Text	اختبار الرقم السري
Label1	Text	رجاءا ادخل الرقم السري
Button1	Text	ابدا الاختبار
TextBox1	PasswordChar	*
PicturBox1	Image	تحدد صورة من الجهاز
PicturBox1	SizeMode	StretchImage
PicturBox1	Visible	False

الكود :

```
Public Class Form1

    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles
Button1.Click
        If TextBox1.Text = "123" Then
            TextBox1.BackColor = Color.Red
            PictureBox1.Visible = True
        End If
    End Sub
End Class
```

تمرين : اعادة البرنامج السابق (الرقم السري) باستخدام :

1- **If-Else .**
2- **If-Else if .**
3- **استخدام المعاملات المنطقية (OR , AND) مع كل حالات من الحالات السابقة وذلك باضافة مربع نص اخر وشرط اخر .**

الاحداث

تعتمد فكرة البرمجة في Visual Basic على الأحداث، وكما ذكرنا، الحدث هو فعل يقوم به المستخدم على البرنامج مثل ضغط زر أو تحريك الماوس , لذلك ستجد أن شفرة Visual Basic مقسمة إلى إجرائيات (أحداث).

تملك Visual Basic نافذة خاصة لتحرير الشيفرة وكتابة التعليمات وهذه النافذة تعتبر بمثابة محرر نصوص بسيط، فهي توفر عمليات النسخ والقص واللصق، وتساعد في تحرير الشيفرة وإكمالها من خلال عرض قوائم الخيارات وما شابه.

وأهم ما تقوم به هذه النافذة هي تقسيم الشيفرة إلى أجزاء (برامج جزئية) ويتم ذلك اعتماداً على الأحداث، فمقابل كل حدث يوجد برنامج جزئي(إجراء) يُستدعى تلقائياً عند وقوع الحدث.

وللوصول إلى أحد أحداث أداة ما، نقوم بالنقر المزدوج فوق هذه الأداة فتظهر نافذة الشيفرة الخاصة بهذه الأداة وبحدثها الأكثر استخداماً، ويمكن التنقل بين الأدوات والأحداث الخاصة بها باستخدام القوائم الموجود في أعلى نافذة الشيفرة .

فمثلاً: لو أضفت زر أمر Button ثم ضغطت فوقه ضغطتين مزدوجتين ستظهر لك نافذة الشفرة كما في الشكل التالي:

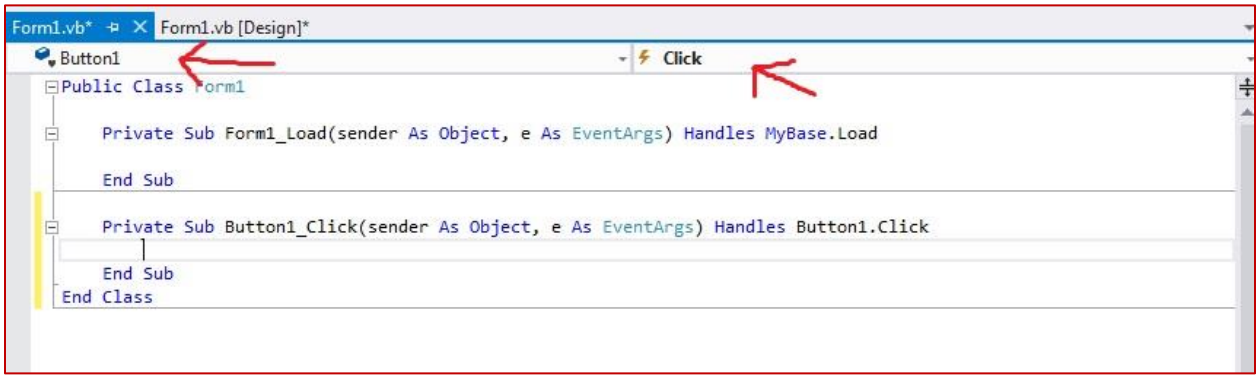
كما نلاحظ تقوم Visual Basic بتوليد إجرائية خاصة بكل حدث، اسم هذه الإجرائية مكون من اسم الأداة واسم الحدث على الشكل:

اسم الأداة_اسم الحدث

فما نلاحظه هو اسم الإجرائية السابقة هو: Command1_Click

تحوي القائمة الموجودة في أعلى يسار نافذة الشفرة على اسماء جميع الأدوات الموجودة على النافذة، بالإضافة إلى اسم النافذة Form وقسم التصريح General.

تحوي القائمة الموجودة في أعلى يمين نافذة الشفرة على أسماء الأحداث التابعة للأداة المختارة من القائمة اليسرى.



عند اختيار الأداة والحدث تتولد الإجرائية المناسبة كما يلي:

```
Private Sub اسم الأداة_اسم الحدث()

End Sub
```

مثلا : `Private Sub Button1_Click`

بعد ظهور سطري البداية والنهاية الخاصين بالإجرائية يمكنك كتابة الشفرة التي تريد، ولكن تذكر أنه يجب أن تبقى الشفرة ضمن السطرين السابقين، وأن هذه الشفرة ستنفذ عند وقوع الحدث.

ادوات الاختيار (Radio Button & CheckBox)

تعتبر هاتان الاداتان من اهم الادوات واكثرها استخداما :

- **CheckBox** تستخدم هذه الاداة لاختبار حالة ما هل هي صحيحة أم لا مثل اختبار حالة شخص هل هو متزوج أم لا وبالتالي يمكن إستعمال مجموعة أدوات Check لتحديد مواصفات شخص بحيث تأخذ كل أداة صفة مثل متزوج أم لا, يملك بيت أم لا، يملك تليفون أم لا ، يملك سيارة أم لا وهكذا , ومن هذه يمكن تحقق أكثر من صفة فى نفس الوقت.

- **Radio Button** تستخدم هذه الاداة لتحديد إختيار من مجموعة اختيارات بحيث لايمكن للاختيارات جميعها ان تتحقق ولايمكن لاكثر من اختيار أن يتحقق فى نفس الوقت ومثال لذلك اختبار تقدير طالب فنسخدم أداة لكل تقدير (مقبول ، جيد ، جيدجدا ، امتياز)فلايمكن أن يأخذ الطالب أكثر من تقدير فى نفس الوقت ,

ملاحظة : من اهم الخصائص التي تتعامل معها الاداتان هي خاصية الاختيار checked وتاخذ اما true او false.

المثال التالي يوضح عمل الاداتين سابقتي الذكر عن طريق تصميم آلة حاسبة كما في الشكل التالي:

فعند ادخال العددين المطلوب اجراء العملية الرياضية عليهما في مربعي النص الاول والثاني يتم تحديد العملية ثم اجراءها وهنا يمكن ملاحظة عدم امكانية اختيار عمليتين بنفس الوقت وهذه من مميزات Radio Button .

اما على الجانب الايمن فيمكن ملاحظة امكانية اختيار الخيارين بتغيير لون خلفية Form وكذلك لون خلفية Label او عدم اختيارهما. كما يوضحها الكود الاتي :

```
Public Class Form1
```

```
Public Class Form1
    Private Sub Button1_Click(sender As Object, e As EventArgs)
Handles Button1.Click
        If RadioButton1.Checked = True Then
            Label1.Text = Val(TextBox1.Text) + Val(TextBox2.Text)
        ElseIf RadioButton2.Checked = True Then
            Label1.Text = Val(TextBox1.Text) - Val(TextBox2.Text)
        ElseIf RadioButton3.Checked = True Then
            Label1.Text = Val(TextBox1.Text) * Val(TextBox2.Text)
        ElseIf RadioButton4.Checked = True Then
            Label1.Text = Val(TextBox1.Text) / Val(TextBox2.Text)
        Else
            End
        End If
    End Sub

    Private Sub CheckBox1_CheckedChanged(sender As Object, e As
EventArgs) Handles CheckBox1.CheckedChanged
        If CheckBox1.Checked = True Then
```

```
            Label1.BackColor = Color.Yellow
        Else
            Label1.BackColor = Color.Wheat
        End If
    End Sub

    Private Sub CheckBox2_CheckedChanged(sender As Object, e As
EventArgs) Handles CheckBox2.CheckedChanged
        If CheckBox2.Checked = True Then
            Me.BackColor = Color.Red
        Else
            Me.BackColor = Color.Gray
        End If

    End Sub
```