الفصل السابع

مفاهيم البرمجة بواسطة الأهداف OOP

فى هذا الفصل Programming Concepts

7

تعتبر مفاهيم البرمجة بواسطة الأهداف OOP من أهم مفاهيم البرمجة الحديثة حيث أصبحت معظم لغات البرمجة تطبقه وبناء الفصائل classes هو أساس البرمجة بواسطة الأهداف OOP وفى هذا الفصل نشرح الخطوات العملية لبناء الفصائل وذلك من خلال النقاط التالية :

- مفاهيم البرمجة بواسطة الأهداف OOP
- انشـاء واستعمال فصائل Classes
- أضافة دوال أعضاء adding methods to class
- فصائل بدون دوال بناء classes without constructors
- دوال البناء في الفصيلة constructors
- استعمال أداة designer في التعامل مع الفصائل classes
- إنشاء خصائص للفصيلة properties
- إضافة خصائص قراءة أو كتابة فقط read-only and write- only

مفاهيم البرمجة بواسطة الأهداف Object Oriented Programming

تعتبر البرمجة بواسطة الأهداف (object oriented programming)

من الاتجاهات الحديثة للبرمجة ومن أول اللغات التي طبقت هذا المفهوم لغة c++ وكذلك من اللغات التي تلتزم بمفهوم OOP لغة JAVA ولغة VB.NET زادت من اهتمامها بمفهوم OOP بل وتعتبر ذلك من التعديلات المهمة والتي زادت من قوة VB.NET لذلك قبل ان نشرح كيفية تحقيق ذلك في VB.NET نشرح أولاً مفاهيم البرمجة بواسطة الأهداف (OOP)بشكل مختصر لذلك إذا كنت على علم بها انتقل إلي الفقرة التالية وهي استعمال VB.NET في إنشاء الفصائل والا تابع معنا شرح هذه الفقرة حيث نتناول النقاط التالية :

- معنى البرمجة بواسطة الأهداف Object Orient Programming
- معنى الفصيلة class
- ما هي دالة البناء و دوال الهدم constructors & destructors
- ما هي خاصية التوريث inheritence
- ما هي خاصية over loading

معنى البرمجة بواسطة الأهداف OOP(Object oriented programming

تبني فكرة البرمجة بواسطة الأهداف (OOP) على استعمال الهدف (Object) كوحدة برمجة بمعنى انه بدلاً من استعمال الدوال والأوامر لبناء البرنامج مما يضطر المبرمج لإعادة كتابة الأوامر كل مرة لتحقيق فكرة معينة وهذه كانت فكرة البرمجة التقليدية .

ولكن أتت البرمجة بواسطة الأهداف (OOP)لتجعل وحدة بناء البرمجة كبيرة وهي هدف Object أو فصيلة class وبالتالي يتم إعداد مجموعة من الفصائل العامة classes التي تلبي معظم متطلبات إعداد برنامج والتي يحتاجها المبرمج حتى ان بعض المبرمجين يشبه البرمجة بواسطة الأهداف بالبناء باستعمال المباني الجاهزة والبرمجة بالطريقة التقليدية القديمة تشابه البناء باستعمال الأدوات الأولية وبالتالي الفرق بينهما في السرعة كبيرة جدًا.

معنى الفصيلة class

الفصيلة class هي أساس البرمجة بواسطة الأهداف (OOP)وهي التي يبنى عليها

البرنامج وأخذت فكرة الفصيلة CLASS من الواقع فكل عنصر من عناصر الحياة عبارة عن فصيلة class ،فأنت تستطيع أن تطلق علي جميع السيارات إنها من فصيلة CAR مع بعض الاختلافات ،ويمكنك ان تطلق على الطيور فصيلة Birdأى طائر وهكذا تنتمى جميع العناصر الى فصائل classes ،وكل فصيلة تستطيع تمثيلها بعنصرين هما البيانات والدوال (methods , data)، فمثلاً فصيلة الموظف Employee بياناتها هي بيانات الموظف العامة مثل كود الموظف ، اسم الموظف ، عنوان الموظف ،تليفون الموظف ، وباقي بياناته ، وكذلك الدوال (methods) هي دوال تحقيق العمليات التي يمكن أن تتم على الموظف مثل :عملية إضافة موظف جديد ، وحذف موظف موجود ، تعديل بيانات موظف وجميع العناصر يمكن تمثيلها بهذه الطريقة .

إذن الفصيلة هي مجموعة من السطور التي تمثل عنصراً تمثيل تاماً من حيث بيانات العنصر والتي تسمى خصائص proprties، وكذلك دوال العنصر التي تسمى methods وبتقسيم البرنامج إلي فصائل يصبح أكثر نظاماً وأسرع في الإعداد ، حيث قامت ميكروسوفت في منتجها vb.net بإعداد مجموعة كبيرة من الفصائل التي تلبي متطلبات المبرمج،وما عليك عند إعداد البرامج إلا أن تدرس مكتبة الفصائل class Lib الموجودة لتعرف المتوفر منها، وكذلك تتعرف على بيانات (خصائص) proprtie ودوال الفصيلة Methods وما شاهدناه في الفصول الأولي من الأوامر التي يكتبها vb.net عند إنشائك لبرنامج جديد هو استعمال فصائل كل عنصر تستعمله ، فمثلاً عند إضافتك لنموذج (form) تلاحظ انه يستورث الفصيلة form وبالمثل عند إضافتك لزر الأمر button1 يتم استوراث الفصيلة button وهكذا .

وهذا الأسلوب لم يكن واضحاً في الإصدار السابق vb6 حيث كنت تستعمل ذلك بدون الشعور بالفصائل ولا التعامل معها ولا التعديل فيها ولكن أتاح لك vb.net التعامل مع الفصائل للاستفادة من مفهوم OOP .

والسؤال :كيف أستطيع إنشاء فصيلة CLASS والتعامل معها وتطبيق مفاهيم OOP ؟

الإجابة: هذا ما سوف نوضحه في هذا الفصل بعد الانتهاء.

أولاً : توضيح المفاهيم الأساسية لمفهوم OOP

دوال تنفذ عند البداية Construction

دوال البناء (constructors): هي دوال تنفذ تلقائياً عند استعمالك للفصيلة class.

(عند تعريف عنصر object) وهي تشبه حدث form-load الموجود في الـ form ،تستعمل في تسجيل أي قيم ابتدائية أو تعريف أي متغيرات أو أي شروط ابتدائية.

دوال تنفذ عند النهاية destructors : هي دالة أو دوال تنفذ تلقائياً عند الانتهاء من استعمال الفصيلة class ويصبح هدف الفصيلة يشير إلى nothing .

وهي تشبه الحدث form-unload الموجود في الform وتستعمل للإنهاء أو التخلص من تعريف متغيرات أو إجراء أي عمليات قبل الخروج من البرنامج.

ما هي خاصية التوريث inheritance

معنى خاصية التوريث (inheritance) هو توريث فصيلة Base class قديمة موجودة بالفعل لفصيلة class جديدة عند إنشائها بحيث يضاف تركيب الفصيلة Base class القديمة من بيانات و دوال إلى الفصيلة الجديدة new class ثم تكمل عليها في الفصيلة الجديدة

وهذه الخاصية من أهم خصائص مفهوم OOP فعلى أساسه تبنى مكتبات الفصائل Class Lib حيث يتم بناء فصيلة أساس Base class ثم تستورثها الفصيلة الثانية والثالثة وتأخذ الفصائل من بعضها البعض حتى تكون مكتبة فصائل عبارة عن شجرة فصائل Class Tree وهذا هو الحال في مكتبة الفصائل الشهيرة MFC الخاصة بميكروسوفت في لغة Visual C++ وكذلك مكتبة فصائل لغة Java المعروفة بالاسم JFC.

ما هي الخاصية Overloading

معنى خاصية **overloading** : هو إمكانية إنشاء اكثر من دالة Method بنفس الاسم مع تغيير عدد المعاملات Parameters مثلاً ، وهذا يفيد بإنشاء أكثر من دالة لنفس الوظيفة

بمعاملات مختلفة بنفس الاسم ، مثل إنشاء دالتين بالاسم ( )find cust إحداهما تأخذ معامل رقمي هو كود الموظف ، والثانية تأخذ عبارة حرفية هي اسم الموظف وفي الحالتين تبحث الدالة ، و بالتالي يشعر المبرمج كأنها دالة واحدة ولكنها دالتين يتم استدعاء كل واحدة تلقائياً حسب المعامل المرسل لها .

فى VB.NET

إنشاء واستعمال الفصائل creating & Use class

بعد شرح بعض المفاهيم المهمة لمفهوم OOP تعال معنا نشرح عملياً كيفية إنشاء الفصائل CLASSES وتحقيق هذه المفاهيم :

الفصيلة class هى أساس مفهوم OOP لذلك أول ما تعلمه فى هذا المفهوم هو كيفية إنشاء واستعمال المصفوفة class ولتحقيق ذلك تابع معى الخطوات التالية:

1- قم بإنشاء تطبيق جديد من النوع ConsoleApplication بالخطوات المعتادة كما سبق.

2- أسفل الدالة الرئيسية اكتب سطور إنشاء فصيلة جديدة بالاسم class1 كما فى الشكل (7-1).

الشكل (1-7)

فى لغة C#

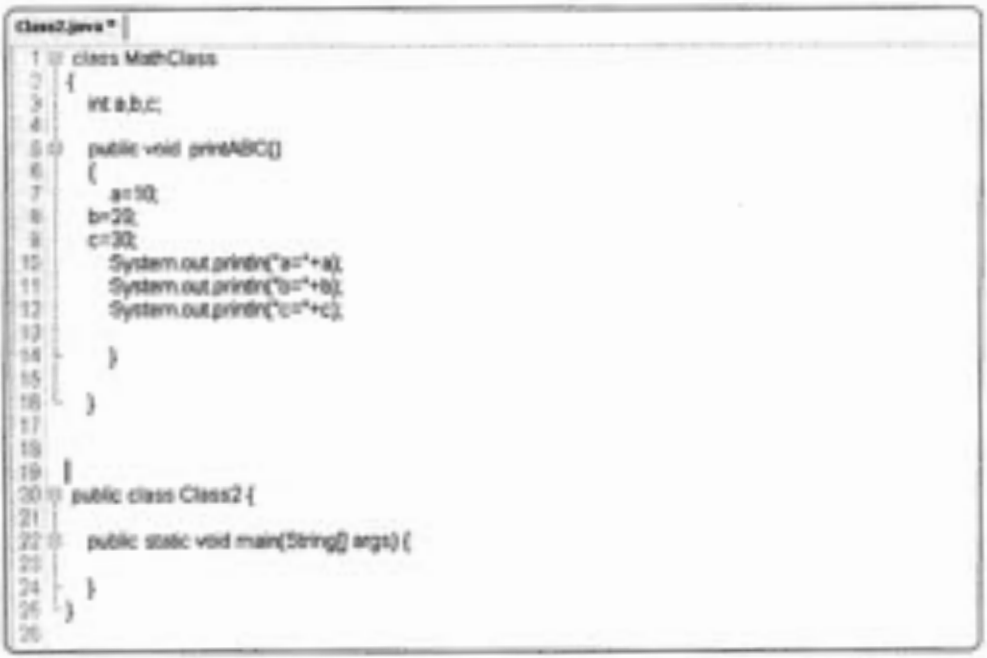
```
using System;
namespace ConsoleApplication3
{
    class Class1
    {
        [STAThread]
        static void Main(string[] args)
        {

        }
    }
    public class MyClass
    {
    public  int a,b,c;
    public  void printabc()
        {
            a=10;
            b=20;
            c=30;
            Console.WriteLine ("a="+a);
            Console.WriteLine ("b="+b);
            Console.WriteLine ("c="+c);

        }

    }
}
```

فى لغة Java



```
class MathClass
{
  int a,b,c;

  public void printABC()
  {
    a=10;
  b=20;
  c=30;
    System.out.println("a="+a);
    System.out.println("b="+b);
    System.out.println("c="+c);

    }

  }


public class Class2 {

  public static void main(String[] args) {

  }
}
```

1- في هذا الشكل في السطر رقم 7 تم إنشاء فصيلة بالاسم class1 وذلك باستعمال الكلمة المحجوزة class مع وضع كلمة publi قبلها حتى يمكن التعامل مع الفصيلة class1.

2- في السطر رقم 8 تم الاعلان عن المتغيرات a,b,c من النوع Integer مع وضع كلمةpublic قبلها حتى يمكن التعامل مع هذه المتغيرات مباشرة من خارج الفصيلة class1 وسوف يتم توضيح هذه النقطة.

3- في السطر رقم 9 تم إنشاء Method من النوع Sub إجراء بالاسم ()Printabc.

- مع وضع كلمة public قبلها حتى يمكن التعامل مع هذا الاجراء مباشرة من خارج الفصيلة class1 وسوف يتم توضيح هذه النقطة.
- وداخل سطور هذا الاجراء تم استعمال الدالة ()WriteIn ثلاث مرات لطباعة قيم المتغيرات a,b,c كما في السطور 10,11,12.
- بهذا تم إنشاء فصيلة جديدة Class بالاسم class1 مع تعريف متغيرات data (a,b,c) ودالة ()Printabc كأعضاء لهذة الفصيلة class1.
- من فضلك قم بمراجعة سطور إنشاء الفصيلة class1 جيدا وحاول استيعاب كيفية إنشاء وكتابة سطور الفصيلة class.

والخطوة التالية هي كيفية استعمال الفصيلة الجديدة class1 لتوضيح ذلك تابع الخطوات التالية:

1- عد الى البرامج وداخل سطور الدالة الرئيسية()Sub Main اكتب سطور استعمال الفصيلة class1 كما في الشكل (2-7).

الشكل (2-7)

فى لغة C#

```
using System;
namespace ConsoleApplication1
{
    class Class1
    {
        [STAThread]
        static void Main(string[] args)
        {
Myclass MyObj=new Myclass();
MyObj.printabc ();
        }
    }
    public class Myclass
    {
    public  int a,b,c;
    public  void printabc()
        {
            a=10;
            b=20;
            c=30;
            Console.WriteLine ("a="+a);
            Console.WriteLine ("b="+b);
            Console.WriteLine ("c="+c);

        }

    }
}
```

فى لغة Java

```
Class2.java *
1  class MathClass
2  {
3     int a,b,c;
4
5     public void printABC()
6     {
7        a=10;
8     b=20;
9     c=30;
10       System.out.println("a="+a);
11       System.out.println("b="+b);
12       System.out.println("c="+c);
13
14       }
15
16    }
17
18
19 public class Class2 {
20
21    public static void main(String[] args) {
22       MathClass obj1=new MathClass();
23       obj1.printABC();
24
25    }
26 }
```

فى هذا الشكل تم استعمال المصفوفة كما يلى:

- فى السطر رقم 4 تم الاعلان عن متغير بالاسم nc من نوع الفصيلة class1 فهو فى هذه الحالة لا يسمى متغير بل يسمى هدف Object مع استعمال الكلمة new لإنشاء هدف object فعلى بالذاكرة
- وبالتالى تكون أول خطوة لاستعمال الفصيلة class هى تعريف هدف object من الفصيلةclass كما فى السطر التالى:

```
Dim nc As New class1()
```

- فى السطر رقم 5 تم وضع القيمة 20 فى المتغير a التابع للفصلة class1 ولكن مع الهدف nc.
- بالمثل السطر 6 و7 تم وضع قيم فى المتغيرات.

- وبالتالى أى تعامل مع الفصيلة لا يتم مع اسمها بل يتم مع اسم الهدف المعرف منها مع ملاحظة امكانية تعريف أكثر من هدف وبالتالى تغيير القيم فى كل هدف .
- فى السطر رقم 8 تم استدعاء الدالة Printabc() ولكن مع اسم الهدف من الفصيلة وهو nc حيث لا يصلح التعامل مع أى عضو داخل الفصيلة الا عن طريق متغير الهدف المعرف من الفصيلة.

3- قم بتنفيذ البرنامج تلاحظ استدعاء السطور المكتوبة داخل الدالة الرئيسية وهى تعريف هدف من الفصيلة واعطاء قيم لمتغيرات الفصيلة ثم طباعة هذه القيم باستدعاء الدالة Printabc() كما فى الشكل (3-7) .

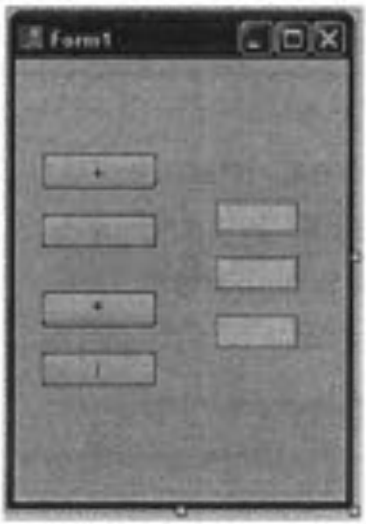
الشكل (3-7)

صديقى حاول التدرب كثيرا على أعداد الفصائل classes بما تحتويه من بيانات Data Members ودوال اعضاء Methods وكذلك التدريب على استعمال هذه الفصائل classes بتعريف متغير object ثم استدعاء الدوال الاعضاء Methods

ولزيادة التوضيح تابع معنا المثال التالى:

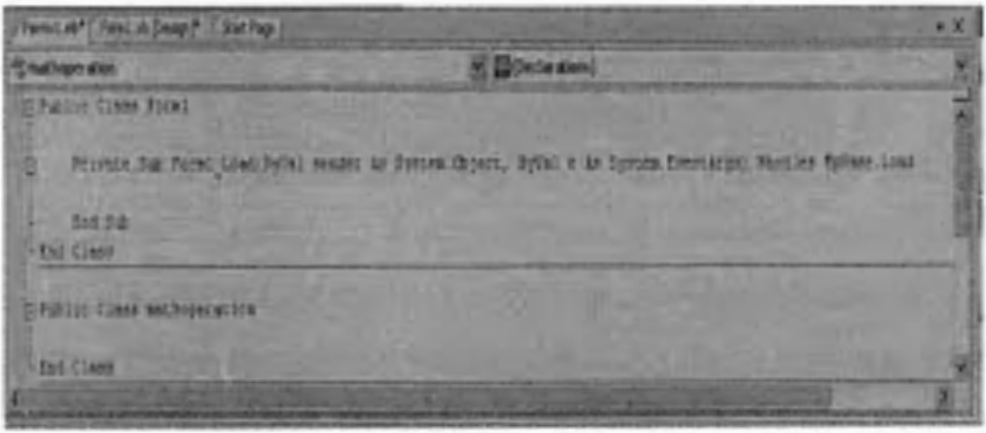
1- قم بإنشاء تطبيق جديد من النوع Windows Application بالخطوات المعتادة كما سبق

2- قم بتصميم برنامج العمليات الحسابية الذى تم اعداده فى الفصل الأول كما فى الشكل (4-7) .



الشكل (4-7)

3- اضغط بالماوس مرتين فوق خلفية البرنامج Form1 للحصول على صفحة الأوامر في نهاية الصفحة قم بكتابة سطور إنشاء فصيلة class جديدة كما في الشكل (5-7) .



الشكل (5-7)

في لغة C#

```
150         static void Main()
151         {
152             Application.Run(new Form1());
153         }
154
155         private void Form1_Load(object sender, System.EventArgs e)
156         {
157
158         }
159     }
160
161     public class mathoperations
162     {
163
164     }
165
166 }
167
```

فى لغة Java

Mathoperations.java *

```
1  class operations
2  {
3
4  }
5
6
7  public class Mathoperations {
8
9      public static void main(String[] args) {
10
11     }
12 }
```

- فى هذا الشكل فى السطر رقم 122 تم إنشاء فصيلة جديدة بالاسم mathoperations وذلك بالاستعمال الكلمة المحجوزة class وتلقائيا تم إغلاق هذه الفصيلة بالعبارة End class.

4- داخل سطور الفصيلة الجديدة mathoperations قم بتعريف متغيرات جديدة كما فى السطر التالى Public a, b, c As Double.

5- داخل سطور الفصيلة الجديدة mathoperations قم بإنشاء دوال العمليات الحسابية فقم بتعريف دالة لعملية الجمع واخرى لعملية الطرح وعملية الضرب وعملية القسمة ، لتصبح الفصيلة الجديدة mathoperations كما فى السطور التالية:

```
Public Class mathoperations
    Public a, b, c As Double
    Public Function sum(ByVal v1 As Double, ByVal v2 As Double)
        Dim res As Double
        res = v1 + v2
        Return res
    End Function

    Public Function subs(ByVal v1 As Double, ByVal v2 As Double)
        Dim res As Double
        res = v1 - v2
        Return res
    End Function
```

```
    Public Function mul(ByVal v1 As Double, ByVal v2 As Double)
        Dim res As Double
        res = v1 * v2
        Return res
    End Function

    Public Function divs(ByVal v1 As Double, ByVal v2 As Double)
        Dim res As Double
        res = v1 / v2
        Return res
    End Function

End Class
```

فى لغة C#

```
    public class mathoperations
    {
    public double a, b, c;
  public double sum( double v1 ,  double v2  )
      {
    double res;
    res = v1 + v2;
        return res;
        }
public double sub( double v1 ,  double v2  )
      {
    double res;
    res = v1 - v2;
        return res;
        }

        public double mul( double v1 ,  double v2  )
        {
            double res;
            res = v1 * v2;
            return res;
        }

public double div( double v1 ,  double v2  )
      {
    double res;
    res = v1 / v2;
        return res;
        }

    }
```

في لغة Java

```
class operations
{
  public double a,b,c;

  public double sum(double v1,double v2)
  {
    double res;
    res=v1+v2;
    return res;
      }
  public double sub(double v1,double v2)
  {
    double res;
    res=v1-v2;
    return res;
      }
  public double mul(double v1,double v2)
  {
    double res;
    res=v1*v2;
    return res;
      }
   public double div(double v1,double v2)
  {
    double res;
    res=v1/v2;
    return res;
      }
   }


public class Mathoperations {

  public static void main(String[] args) {

  }
}
```

راجع هذه السطور تلاحظ أننا أنشأنا فصيلة جديدة وبها مجموعة دوال العمليات الحسابية التي أشرنا إليها.

6- قم باستعمال الفصية الجديدة كما يلي:

- اضغط زر أمر الجمع واكتب به السطور التالية:

```
1. Dim mathobj As New mathoperations()
2. Dim a, b, c As Double
3. a = Val(TextBox1.Text)
4. b = Val(TextBox2.Text)
5. c = mathobj.sum(a, b)
6. TextBox3.Text = c
```

فى لغة C#

```
161 private void button1_Click(object sender, System.EventArgs e)
162         {
163 mathoperations mathobj=new mathoperations ();
164 double a, b, c ;
165 a = double.Parse (textBox1.Text);
166 b = double.Parse(textBox2.Text);
167 c = mathobj.sum(a, b);
168 textBox3.Text = c+"";
169         }
```

ق هذه السطور:

- ق السطر رقم 1 تم تعريف هدف object بالاسم mathobj من الفصيلة الجديدة mathoperations وذلك باستعمال كلمتى dim وNew كما سبق وشرحنا.
- ق السطر رقم 2 تم الاعلان عن ثلاثة متغيرات a,b,c من النوع Double.
- ق السطر رقم 3 و4 تم وضع قيمة محتوى مربعى النص TextBox1,TextBox2 ق المتغيرين a,b وذلك باستعمال الدالة ()val.
- ق السطر رقم 5 تم استدعاء الدالة ()sum مع هدف الفصيلة mathobj مع إرسال معاملين هما a,b لجمع المتغيرين a,b ووضع النتيجة ق المتغير c.
- ق السطر رقم 6 تم وضع قيمة المتغير c وهو نتيجة الجمع ق مربع النص TextBox3 بهذا تم استعمال الدالة ()sum المعرفة داخل الفصيلة mathoperations
- اضغط زر أمر الطرح واكتب به السطور التالية:

```
7.  Dim mathobj As New mathoperations()
8.  Dim a, b, c As Double
9.  a = Val(TextBox1.Text)
10. b = Val(TextBox2.Text)
11. c = mathobj.subs(a, b)
12. TextBox3.Text = c
13.
```

فى لغة C#

```
161 private void button1_Click(object sender, System.EventArgs e)
162         {
163 mathoperations mathobj=new mathoperations ();
164 double a, b, c ;
165 a = double.Parse (textBox1.Text);
166 b = double.Parse(textBox2.Text);
167 c = mathobj.sub(a, b);
168 textBox3.Text = c+"";
169         }
```

فى هذه السطور:

تلاحظ أنها نفس السطور السابقة مع استدعاء دالة الطرح.

- اضغط زر أمر الضرب واكتب به السطور التالية:

```
14. Dim mathobj As New mathoperations()
15. Dim a, b, c As Double
16. a = Val(TextBox1.Text)
17. b = Val(TextBox2.Text)
18. c = mathobj.mul(a, b)
19. TextBox3.Text = c
```

فى لغة C#

```
161 private void button1_Click(object sender, System.EventArgs e)
162         {
163 mathoperations mathobj=new mathoperations ();
164 double a, b, c ;
165 a = double.Parse (textBox1.Text);
166 b = double.Parse(textBox2.Text);
167 c = mathobj.mul(a, b);
168 textBox3.Text = c+"";
169         }
```

فى هذه السطور:

تلاحظ أنها نفس السطور مع استدعاء دالة الضرب.

- اضغط زر أمر القسمة واكتب به السطور التالية:

```
20. Dim mathobj As New mathoperations()
21. Dim a, b, c As Double
22. a = Val(TextBox1.Text)
23. b = Val(TextBox2.Text)
24. c = mathobj.divs(a, b)
25. TextBox3.Text = c
```

فى لغة C#

```
private void button1_Click(object sender, System.EventArgs e)
        {
mathoperations mathobj=new mathoperations ();
double a, b, c ;
a = double.Parse (textBox1.Text);
b = double.Parse(textBox2.Text);
c = mathobj. div (a, b);
textBox3.Text = c+"";
        }
```

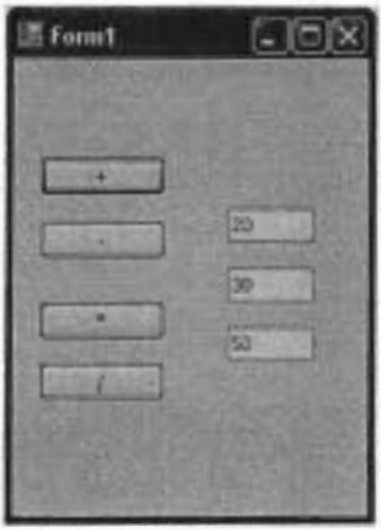
فى هذه السطور:

تلاحظ أنها نفس السطور مع استدعاء دالة القسمة.

7- قم بتشغيل البرنامج لتحصل على نتيجة التنفيذ جرب واكتب رقمين واضغط زر أمر الجمع و الطرح وغيره ولاحظ النتيجة كما فى الشكل (6-7).

فى لغة Java

جميع العمليات كما فى السطور.



الشكل (6-7)

الجديد فى هذا البرنامج:

لم يكن الغرض من هذا البرنامج هو تحقيق عمليات الجمع والطرح والضرب وإلا فقد حققناها فى أول الكتاب ببساطة وبدون استعمال فصائل وإنما الغرض من البرنامج هوتوضيح كيفية إنشاء فصيلة واضافة بيانات اعضاء ودوال أعضاء لها فأرجو أن تكون الصورة واضحة وكما لفت نظرك من قبل حاول التدرب على هذا الأسلوب الجديد وهو إنشاء الفصائل وإضافة دوال وبيانات استعمال هذة الفصائل.

اضافة فصيلة class على مستوى التطبيق:

فى المثال السابق تم إنشاء الفصيلة class داخل Form1 ولكن هناك أكثر من طريقة لإنشاء الفصيلة class على مستوى التطبيق كما يلى:

(1) اضافة ملف Module

يمكن اضافة ملف Module ثم إنشاء الفصيلة class داخلها وبالتالى تصبح هذه الفصيلةclass معرفة على مستوى التطبيق أويمكن اضافة ملف Module لاختيار Project من القائمة الرئيسية ثم اختيار Add Module كما سبق فتحصل على صفحة بيضاء يمكنك

الكتابة بها ويمكنك كتابة الفصيلة الجديدة داخلها كما فى الشكل (7-7).

```
Module Module1

    Public Class Ourclass

    End Class

End Module
```

الشكل (7-7)

(2) إضافة ملف فصيلة مباشرة

يمكن اضافة فصيلة class باضافة ملف للفصيلة وذلك باختيار Add Class من الاختيار Project من القائمة الرئيسية فتحصل على مربع حوار كما فى الشكل (8-7) .

الشكل (7-8)

وبعد تحديد اسم الفصيلة class والضغط على الزر Open تحصل على الفصيلة class الجديدة كما فى الشكل (9-7) .

```
Public Class Class1


End Class
```

الشكل (7-9)

فى هذا الشكل يمكنك كتابة محتويات الفصيلة class1 من بيانات Data ودوال أعضاء Methods.

ملحوظة:

لا تستخدم الكلمة MyClass كأسم للفصيلة class لانها كلمة محجوزة من كلمات Visual Basic.Net التى لها معنى خاص

دوال Constructors

من الخصائص المتوفرة فى مفاهيم البرمجة بالاهداف OOP وتدعمها Visual Basic.Net فكرة وجود دالة البناء Constructor داخل الفصيلة class وهى دالة Method مثل أى دالة ولكن الفرق فى أنها دالة تنفذ تلقائيا بدون استدعاء أى بمجرد تعريف هدف object من الفصيلة class ، والغرض من ذلك هو استعمال هذه الدالة فى تنفيذ أى عمليات أولية للفصيلة class مثل اعطاء قيم ابتدائية للمتغيرات.

وفى لغة Visual Basic.Net يتم تحديد دالة البناء Constructor داخل الفصلة بالاسم New أى تصبح بالصورة التالية:

```
Sub New()
End Sub
```

فى لغة C#

```
 8    public class ExtClass1
 9    {
10        public int v1,v2,v3;
11
12        public ExtClass1()
13        {
14            //
15            // TODO: Add constructor logic here
16            //
17
18        }
19    }
```

فى لغة Java

```
1  public class Constructor {
2      Constructor ()
3      {
4        System.out.println("this is the Constructor >>> it run without calling ");
5        }
6
7        public static void main(String[] args) {
8        Constructor obj=new Constructor();
9          }
10 }
11
```

ولتوضيح ذلك تابع الخطوات التالية:

1- قم باضافة فصيلة class بالاسم class1 كما شرحنا فى الفقرة السابقة.

2- داخل سطور الفصيلة الجديدة class1 قم بإنشاء دالة البناء Constructor بالاسم Sub New() كما فى الشكل (10-7).

```
Public Class Class1

    Public Sub New()
        MsgBox("This is My constructor")
    End Sub

End Class
```

الشكل (10-7)

فى لغة C#

```
using System;
using System.Windows.Forms;
namespace WindowsApplication11
{
    /// <summary>
    /// Summary description for constclass1.
    /// </summary>
    public class constclass1
    {
        public constclass1()
        {
            MessageBox.Show (" this message displayed without call");
        }
    }
}
```

فى لغة Java

```
Methodoperations.java  Constructor.java*
1 public class Constructor {
2     Constructor()
3     {
4       System.out.println("this is the Constructor >>> it run without calling ");
5     }
6
7     public static void main(String[] args) {
8     Constructor obj=new Constructor();
9       }
10 }
11
```

فى هذا الشكل (10-7) تم إنشاء دالة Sub جديدة بالاسم ()New وهى تحتوى على سطر واحد هو عرض رسالة باستعمال الدالة ()MsgBox.

3-قم بتوقيع زر أمر Button1 فوق خلفية البرنامج Form وغير عنوانه خاصية Text إلى Test Constructor.

4- اضغط زر الأمر Button1 مرتين واكتب به السطر التالي :.

```
Dim ConsObj As New Class1()
```

فى لغة C#

```
179         private void button3_Click(object sender, System.EventArgs e)
180         {
181             constclass1 obj1 =new constclass1 ();
182
183         }
```

- فى هذا السطر تم تعريف هدف object من الفصيلة class1 بالاسم ConsObj ولم يتم استدعاء أى دوال.

5- نفذ البرنامج F5 واضغط على زر الأمر تلاحظ ظهور رسالة دالة البناء Costructor بالرغم من عدم استدعائه ولكن تم الاستدعاء تلقائيا وتحصل على النتيجة كما فى الشكل (11-7).

الشكل (11-7)

إنشاء أكثر من دالة باسم واحد Method Overloading

من المفاهيم المشهورة فى البرمجة بالأهداف OOP مفهوم إنشاء أكثر من دالة باسم واحد ويطلق عليه Method Overloading حيث يسمح هذا المفهوم بإنشاء أكثر من دالة بنفس الاسم اذا دعت الحاجة لذلك مثل إنشاء مجموعة دوال رسم بالاسم draw() تستطيع رسم خط Line ومربع Box ودائرة circle وغيره ولكن يتوقف ذلك على معاملات الدالة.

ومثال أخر يمكن إنشاء أكثر من دالة بحث بالاسم Search() واحدة للبحث عن موظف بمعلومية رقم الموظف والثانية للبحث عن الموظف باسم الموظف ويتم استدعاء الأولى أو الثانية حسب المعامل الذى تم إرساله.

ولتوضيح فكرة إنشاء أكثر من دالة بنفس الاسم Mehod Overloading تابع المثال التالى:

1- قم بإنشاء تطبيق جديد من النوع ConsoleApplication بالخطوات المعتادة كما سبق.

2- عدل سطور البرنامج وقم بإنشاء فصيلة جديدة بالاسم Shapes كما فى السطور التالية:

```
26.   Public Class shapes
27.       Public Overloads Sub drawLine()
28.           Dim i As Integer
29.   Console.WriteLine("First Method")
30.           For i = 0 To 20
31.               Console.Write("*")
32.           Next
33.           Console.WriteLine()
34.
35.       End Sub
36. Public Overloads Sub drawLine(ByVal n As Integer)
37.           Dim i As Integer
38.        Console.WriteLine("Secand Method")
39.           For i = 0 To n
40.               Console.Write("*")
41.           Next
42.           Console.WriteLine()
43.       End Sub
44.
45.   Public Overloads Sub drawLine(ByVal n As Integer, ByVal ch As Char)
46.           Dim i As Integer
47.    Console.WriteLine("Thrid Method")
48.           For i = 0 To n
49.               Console.Write(ch)
50.           Next
51.           Console.WriteLine()
52.       End Sub
53.
54.   End Class
55.
```

فى لغة C#

```
24        public class shapes
25        {
26            public void drawline()
27            {
28            for (int i=0;i<10;i++)
29                Console.Write ("*");
30            Console.WriteLine  ("");
31        }
32            public void drawline(int n)
33            {
34                for (int i=0;i<n;i++)
35                    Console.Write ("*");
36                Console.WriteLine  ("");
37            }
38
39            public void drawline(int n,char ch)
40            {
41                for (int i=0;i<n;i++)
42                    Console.Write (ch+"");
43                Console.WriteLine  ("");
44            }
45        }
```

فى لغة Java

```
1  class shapes
2  {
3    public void drawline ()
4    {
5      for(int i=0;i<10;i++)
6      System.out.print("*");
7      System.out.println("");
8      }
9
10   public void drawline (int n)
11   {
12     for(int i=0;i<n;i++)
13     System.out.print("*");
14     System.out.println("");
15     }
16
17     public void drawline (int n,char ch)
18   {
19     for(int i=0;i<n;i++)
20     System.out.print(ch);
21     System.out.println("");
22     }
23
24   }
25
26 public class ShapesApp {
27
28   public static void main(String[] args) {
29     shapes obj=new shapes();
30     obj.drawline();
31     obj.drawline(20);
32     obj.drawline(30,'-');
33
34   }
35 }
```

- فى هذه السطور تم إنشاء فصيلة بالاسم Shapes وداخل هذه الفصيلة Class تم تعريف ثلاث دوال بالاسم ()drawLine كلها باسم واحد ولكن الملاحظ أن الفرق بينها هو المعاملات الأولى بدون معاملات وتقوم بطباعة الحرف *عدد 20 مرة الثانية بمعامل واحد هو n من نوع Integer ويستخدم لتحديد عدد مرات طباعة الحرف *
- الثالثة لها معاملين الأول هو n من نوع Integer ويستخدم لتحديد عدد مرات طباعة الحرف * والثانى ch لتحديد الحرف المطلوب طباعته.
- بعد إنشاء الدوال الثلاثة داخل الفصيلة Shapes قم بكتابة أوامر استعمال الفصيلة Shapes ودوالها الثلاثة داخل الدالة الرئيسية للبرنامج ()main sub كما فى السطور التالية:

```
1. Sub Main()
2. Dim shapobj As New shapes()
3. shapobj.drawLine()
4. shapobj.drawLine(10)
5. shapobj.drawLine(15, "-")
6. 5-End Sub
```

فى لغة C#

```
1  using System;
2
3  namespace ConsoleApplication4
4  {
5          class Class1
6      {
7              [STAThread]
8          static void Main(string[] args)
9          {
10             shapes shobj =new shapes();
11             shobj.drawLine();
12             shobj.drawLine(15);
13             shobj.drawLine(12,'-');
14         }
15 }
```

فى هذه السطور:

- فى السطر رقم 1 تم تعريف هدف Object من الفصيلة Shapes كما سبق.
- فى السطر رقم 2 تم استدعاء الدالة ()drawLine مع هدف الفصيلة ولكن الملاحظ

إننا لم نكتب معاملات للدالة لذلك فأننا نستدعى الدالة الأولى التى لا تأخذ معاملات.

- فى السطر رقم 3 تم استدعاء الدالة drawLine(10) مع هدف الفصيلة ولكن الملاحظ إننا كتبنا معامل للدالة لذلك فأننا نستدعى الدالة الثانية التى تأخذ معامل واحد.
- فى السطر رقم 4 تم استدعاء الدالة drawLine(15, "-") مع هدف الفصيلة ولكن الملاحظ إننا كتبنا معاملين للدالة لذلك فأننا نستدعى الدالة الثالثة التى تأخذ معاملين.
- قم بتنفيذ البرنامج تحصل على نتيجة التنفيذ التى تعبر عن البرامج كما فى الشكل التالى (12-7):

الشكل (12-7)

الدالة Finalize

من الأفكار المتاحة أيضا فى البرمجة بالأهداف OOP فكرة وجود دالة Method داخل الفصيلة class هذه الدالة تنفذ عند الانتهاء من استعمال أهداف الفصيلة مثل إنهاء البرنامج أى عكس عمل دالة البناء Constructor.

وتستخدم هذه الدالة فى كثير من الاستخدمات أشهرها هو التخلص من أى متغيرات تم الاعلان عنها فى أول الفصيلة حتى لا تزدحم الذاكرة بمتغيرات غير مرغوب فيها فكما ذكرنا أن دالة الهدم هذه تنفذ تلقائيا عند الانتهاء من استعمال أهداف الفصيلة وتحقق Visual Basic.Net هذه الفكرة عن طريق إنشاء دالة بالاسم Finalize.

ولتوضيح ذلك قم بإضافة سطور هذه الدالة إلى سطور الفصيلة shapes فى المثال السابق كما فى السطور التالية:

```
Protected Overrides Sub Finalize()
        MsgBox("go go  go  go  go  go  go")
    End Sub
```

فى لغة C#

```
16          public class shapes
19          {
20              ~shapes()
21              {
22  Console.WriteLine  ("this is the Destructor");
23              }
```

- نفذ البرنامج وعند الخروج تلاحظ تنفيذ هذه الدالة وظهور الرسالة تلقائيا كما فى الشكل (13-7).

الشكل (13-7)

قواعد يفضل الالتزام بها عند كتابة البرامج:

الشكل التالى يعرض التركيب المثال للبرنامج من حيث اسم البرنامج والتعليقات وصاحب البرنامج وتاريخ كتابة البرنامج ومكتبات البرنامج وبيئة التطوير ومتطلبات التطوير .

من فضلك حاول مراجعة هذه القواعد وحاول الاسترشاد بها عند كتابة البرامج.

```
//*************************************************************
//*************************************************************
                        //  Program name
                    //  Copyright (c) 200? by
                     //  ALL RIGHTS RESERVED
//*************************************************************
//*************************************************************
          // Date:                              Coded by:
         // Filename:                          Module name:
         //                                    Source file:
                    // Program description:
                              //
//*************************************************************
                // Libraries and software support:
                              //
//*************************************************************
                    // Development environment:
                              //
//*************************************************************
                     // System requirements:
                              //
//*************************************************************
                           // Start date:
                        // Update history:
          //              DATE              MODIFICATION
                              //
//*************************************************************
                        // Test history:
      //  TEST PROTOCOL              DATE        TEST RESULTS
                              //
//*************************************************************
                     // Programmer comments:
                              //
                              //
//*************************************************************
                                 ***
```

ملاحظات يجب مراعاتها عند تصميم الفصائل:

1- حاول أن تحافظ على أن تكون البيانات من النوع الـ private.

2- حاول ان تعطى قيم ابتدائية للمتغيرات initialize data.

3- لا تستعمل متغيرات منفردة كثيرا فبدلا من ذلك حاول تجميعها فى فصيلة class.

فمثلا بالنظر إلى المتغيرات التالية :

```
private String street;
private String city;
private String state;
private int zip;
```

يمكن تجميعا فى فصيلة بالاسم Address.

4- حاول استخدام الصورة القياسية لتعريف الفصيلة class كما فى التركيب التالى:

```
public features

package scope features

private features

Within each section, we list:

instance methods

static methods

instance fields

static fields
```

5- حاول تقسيم الفصائل حسب الوظائف .. وتلافى إنشاء فصيلة تتناول أكثر من موضوع ولتوضيح ذلك راجع هذا التصميم.

```
public class CardDeck // bad design
{
  public CardDeck() { . . . }
  public void shuffle() { . . . }
  public int getTopValue() { . . . }
  public int getTopSuit() { . . . }
  public void draw() { . . . }

  private int[] value;
  private int[] suit;
}
```

فى هذا التصميم تلاحظ احتوى على بيانات أكثر من موضوع وهذا التصميم سيئ بل يجب فصل الموضوعات بحيث لا تحتوى الفصيلة إلا على موضوع واحد .

وبالتالى يصبح التصميم الجيد كما فى السطور التالية:

```
public class CardDeck
{
   public CardDeck() { . . . }
   public void shuffle() { . . . }
   public Card getTop() { . . . }
   public void draw() { . . . }

   private Card[] cards;
}

public class Card
{
   public Card(int aValue, int aSuit) { . . . }
   public int getValue() { . . . }
   public int getSuit() { . . . }

   private int value;
   private int suit;
}
```

6- حاول اعطاء أسماء للفصائل والمتغيرات معبرة عن الغرض منها كمات يجب أن تلاحظ انه من المشهور إنشاء الدوال METHODS بالأسماء ()_SET و ()GET حيث أن مجموعة دوال ()Set_ تقوم باعطاء قيم لمتغيرات الفصيلة ومجموعة دوال ()Get_ تقوم باعادة قيم متغيرات الفصيلة.