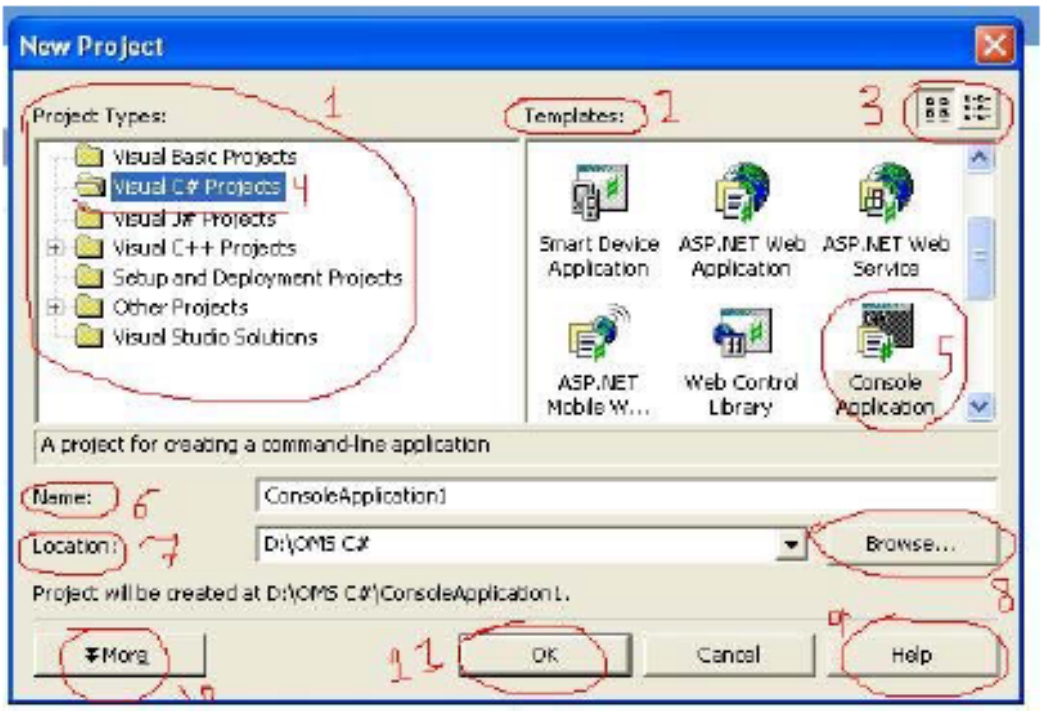
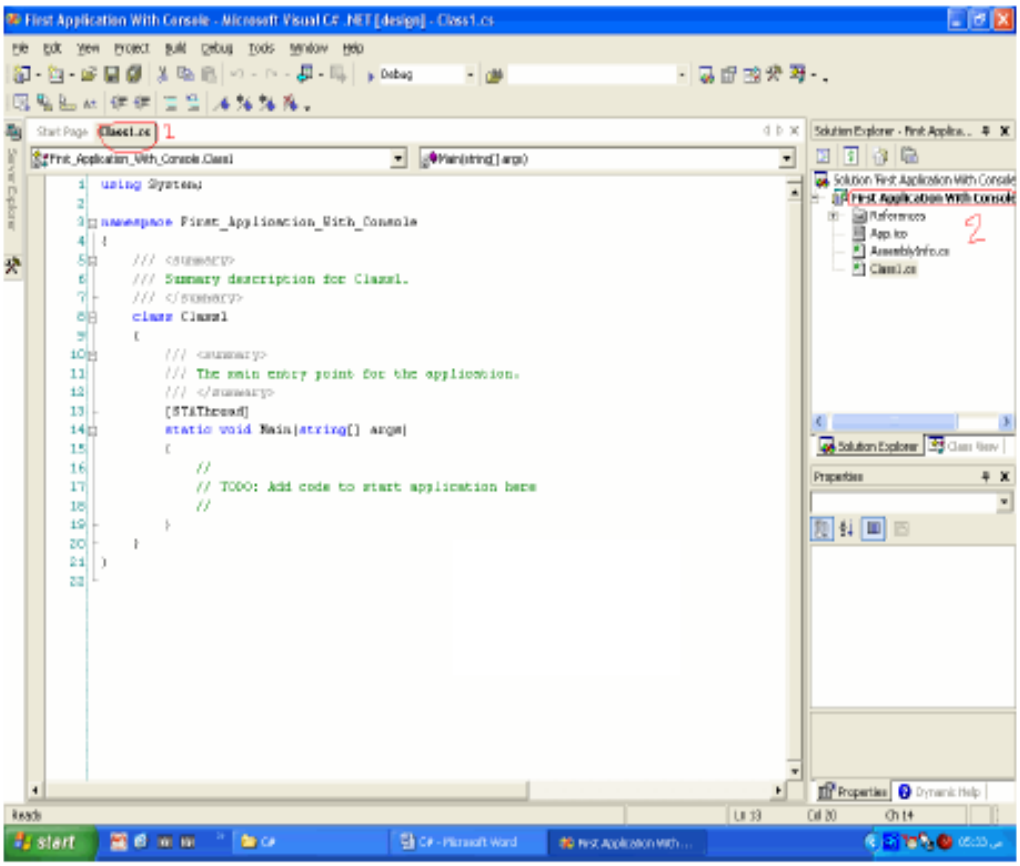
المحاضرة الأولى

إذهب الآن إلى File → New → Project .
أو قم بالضغط على Ctrl + Shift + N .
أو قم بالنقر على زر المشروع الجديد من شريط الإختصارات .
إذا نجحت في ذلك ستظهر لك النافذة التالية :



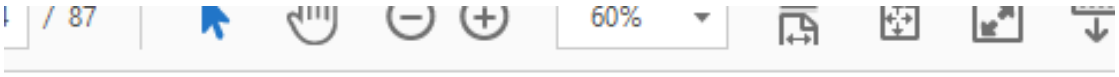
لنقف قليلاً عند هذه النافذة لكي نبين أجزائها :
1 – أنظر هنا في هذا القسم ((1)) ستلاحظ أسماء جميع اللغات المستعملة في بيئة الدوت نيت وبهذا تكون شركة مايكروسوفت قد نجحت في تجميع جميع اللغات في إطار واحد .
2 – تجد جميع الأقسام ((القوالب)) التي يمكنك تصميمها مثل تطبيقات الويندوز وتطبيقات الكونسول وتطبيقات السمارت ديفايس ((Pocket PC)) وذلك في القسم ((2)) .
3 – يمكنك أن تصغر وتكبر الأيقونات الموجودة في القسم ((2)) بواسطة الأزرار في القسم ((3)) .
4 – إذن الآن لنقم بفتح مشروع سي شارب فقم بالضغط على ((4)) .
5 – ثم بعدها قم بالضغط على ((5)) ولاحظ أن تطبيقات الـ Console سبق لنا تعريفها وهي التطبيقات التي تفتح شاشة سوداء كنظام الـ DOS .
6 – يمكنك تسمية المشروع من الرقم ((6)) وتذكر أنه سيسمي الملفات كلها بهذا الإسم ((ملفات العمل)) .
7 – من إسمها ((7)) تعرف أنها مكان منطقة العمل Directory . وبإمكانك تحديد مكان معين بالضغط على الزر الذي يحمل الرقم ((8)) .
8 – إذا ضغط هذا المفتاح ((9))سينتج لك نافذة مساعدة عن هذه الصفحة فقط .
9 – بإمكانك زيادة الخيارات المستخدمة عندك بإضافة عملك في مجلد جديد على المسار الذي حددته في الخطوة رقم ((7)) وذلك بالضغط على الرقم ((10)) والذي يحمل كلمة More .
10 - وأخيراً قم بعمل OK ((موافق)) لنبدأ مشروعنا اليوم .



أنظر إلى النافذة السابقة ولاحظ معي :
1 – قام بفتح نافذة جديدة ((1)) وسماها Class1.cs .
يتكون الإسم من قسمين كالتالي :
القسم الأول يحمل إسم Class1 وهذه تعتبر الكائن الرئيسي في المشروع لأنه سبق لنا أن قلنا أننا نسعى للغة تدعم البرمجة الكائنية OOP ولأننا قلنا أنه يجب أن يكون هناك على الأقل كائن واحد وهو يحمل الدالة الرئيسية Main Function وبهذه الخطوة إرتقت مايكروسوفت إلى مستوى البرمجة بالكائنات والتي لم تكن موجودة بالإصدارات السابقة .

والقسم الثاني يحمل إسم الإمتداد cs ومعناه C Sharp أي إسم اللغة

2 –في القسم ((2)) ماذا تلاحظ ؟
المشروع يحتوي على 3 ملفات فقط . ملف للأيقونة الناتجة في المشروع وملف التحويل للغة الأسمبلي والملف الثالث المحتوي على الكود الذي نقوم بكتابته .
ملاحظة سريعة هنا :
حاول و إفتح مشاريع غير السي شارب ((ماذا تلاحظ)) ؟
ستقول لي كذلك تنتج 3 ملفات . وفي هذه الخطوة إستطاعت شركة مايكروسوفت توحيد أنماط جميع لغاتها ضمن باقة دوت نيت .



- لنأتي الآن إلى منطقة العمل ونشرحها بالتفصيل :
لاحظ معي النافذة كالتالي :

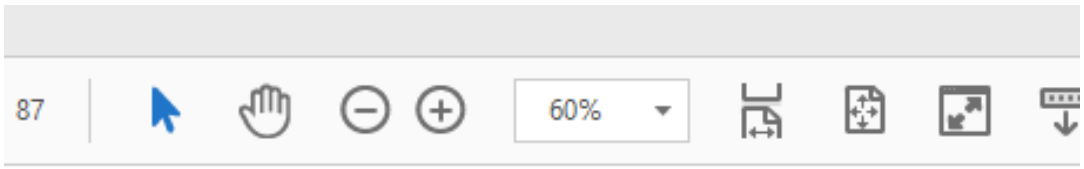
```
1  using System;
2
3  namespace First_Application_With_Console
4  {
5      /// <summary>
6      /// Summary description for Class1.
7      /// </summary>
8      class Class1
9      {
10         /// <summary>
11         /// The main entry point for the application.
12         /// </summary>
13         [STAThread]
14         static void Main(string[] args)
15         {
16             //
17             // TODO: Add code to start application here
18             //
19         }
20     }
21 }
22
```

((إقرأ الملاحظة بعد الموضوع مباشرة))

في السطر رقم (1) جملة using System; نستطيع إستنتاج التالي :
* جميع الكلمات الموجودة في منطقة العمل والتي تحمل اللون الأزرق هي كلمات محجوزة Keyword لا نستطيع إستعمالها كمتغيرات .
* تعتبر لغة السي شارب لغة حساسة Case Sensitive يعني أن المتغيرات (Age, AGE, aGE , AgE) كلها متغيرات لا يشبه بعضها بعضاً وتعاملها هذه اللغة كل واحدة على حده .
* نهاية كل جملة تحتوي على فاصلة منقوطة ; وهي تعبر عن نهاية السطر .

ووظيفة السطر الأول هي إستدعاء مكتبة ((سنتفق على تسميتها namespace)) للتعامل مع المشروع بشكل جيد مثل جمل الإدخال والإخراج وتعادل هذه الجملة بكلمة #include في لغة السي هنا إستدعى مكتبة الـ System ولاحظ أن أول حرف كبير وهذه المكتبة مختصة بالدوال الرئيسية التي تستخدم بكثرة كجمل الإدخال والإخراج وتعادل هذه المكتبة مكتبة الـ iostream.h المستخدمة في لغة السي .

في السطر رقم (3) جملة namespace First_Application_With_Console نستطيع إستنتاج التالي :
* قام بإنشاء مكتبة تحتوي على المشروع الذي نكتبه الآن .
* قام بوضع علامة تحت السطر (_) بدلاً من الفراغات والتي أصلاً إسم مشروعنا الحالي .
* تلاحظ أنه يوجد مربع صغير يحتوي على إشارة ناقص (-) ووظيفته إخفاء تفاصيل الكلاس أو الدالة المشار إليها وبعد الضغط عليه يظهر لنا مستطيل يحتوي على ثلاث نقاط (...) إذا حركت الماوس عليه يعطيك محتوى الكلاس أو الدالة المشار إليها كاملاً كشكل ملاحظة Tag بمستطيل أصفر اللون فيعرض لك محتوياتها مهما بلغت من الطول .
والهدف من هذا المربع هو إخفاء دالة أو كلاس سبق لنا أن كتبناها ولا نريد إظهارها .



ومعنى هذه الجملة أنه قام بإنشاء مكتبة خاصة والتي تحتوي على الـ Classes الموجودة في مشروعنا الحالي فمثلاً إذا أردنا إستدعاء دالة من الدوال في مشروع اخر ما علينا سوى كتابة إسم المشروع الحالي ثم إتباعه بنقطة ثم إسم الكلاس أو الدالة التي نريد إستعمالها .

في السطر رقم (4) القوس المشهور ({) والذي يدل على بداية الدالة أو الكلاس وطبعاً نغلقها بالمثل بإستخدام القوس المثيل (}) كما في السطر رقم (21) .

في السطر رقم (5) `/// <summary>` جملة تعيق ولكنها للغة الـ XML دعها جانباً لن تفيدنا الآن في الوقت الحالي فلها وقتها . لاحظ أنها تحتوي على ثلاث أقواس .

في السطر رقم (7) لاحظ وجود إشارة (-) على العمود وهذا يعني بداية الجملة الأولى في الكلاس أو الدالة Function .

في السطر رقم (8) تلاحظ وجود إسم الكلاس المستعملة في مشروعنا الحالي .

في السطر رقم (13) `[STAThread]` أي كلمة موجودة بين قوسين ((مربعين)) تسمى خاصية Attribute وسنقوم بشرحها لاحقاً .

في السطر رقم (14) `static void Main(string[] args)` هنا توجد الدالة الرئيسية لمشروعنا .

والجملة تتكون من :

* `static void` تحديد نوع الدالة الرئيسية فهي من نوع Void التي تعني أن الدالة لا ترجع أي قيمة وهي من القسم Static من النوع الإستاتيكي .

والنوع الإستاتيكي يمكن شرحه كالتالي : لو أنك عرّفت متغير ما بالنوع الإستاتيكي في دالة معينة في داخل كلاس معين ثم إستدعيت الدالة وأجريت تعديلات على هذا المتغير وخرجت من الدالة فإن الأصل أن يحذف المتغير من الذاكرة ولكن المتغير الإستاتيكي يقوم بتسجيل نسخه في الذاكرة ما دام البرنامج أو المشروع الذي صممته في وقت التنفيذ . فمثلاً لو عرفت في دالة معينة المتغير X من نوع Int من النوع الإستاتيكي وقمت في سطر تالي بزيادة هذا المتغير بقيمة واحد فإنه كلما قمت بإستدعاء الدالة سينفذ السطر الثاني فقط ويقفز عن السطر الأول لأنه موجود في الذاكرة.
دعه الآن له وقت سنشرحه بالتفصيل .

* `Main` لاحظ أن أول حرف كبير .
* `(string[] args)` وهي هنا تعني أننا بإمكاننا أن نستخدم الوسائط ((البارامترات))

فمثلاً لو أنشأنا مشروع لجمع عددين وقمنا بتسميته Sum طبعاً ستقول لي بعد تنفيذ المشروع نكتب الجمل اللازمة لكي يقوم بالحساب في ما بينهما وذلك بالطلب من المستخدم أن يدخل رقمين مباشراً من طريق جمل الإدخال . حسناً هنا بإمكانك قبل تنفيذ المشروع أن تدخل العددين وتقوم بالتعامل معهما فمثلاً نذهب إلى محرر الدوس ونقوم بكتابة الجملة التالي :

C:\> Sum.exe 152 965

فنستطيع مباشرة وبأول جملة في المشروع أن نعطيه الناتج .

في السطر رقم (16) `//` لاحظ وجود قوسين هنا وهما لجمل التعليقات ((لاحظ الفرق في السطر الخامس)) . أي أنها جمل لا معنى لها تضع التوضيح أو التعليق على الجمل في هذا السطر

ملاحظة :
إذا واجهتك أي مشكلة في أي سطر وتريد معرفة المزيد قم بالنقر مرتين مزدوجتين على الجملة ثم قم بالضغط على F1 لظهور نافذة المساعدة بالجملة التي تريد فقط ((يجب أن تمتلك MSDN))

مبادئ أساسية في C#

بعد تنصيب Visual studio 2008 نختار من شريط القوائم File → New → Project

بعدها نختار نوع التطبيق Console Application ضمن صندوق النص Name نكتب الاسم التطبيق وضمن location نحدد موقع التطبيق ثم نضغط على زر OK

عندها يتم بناء التطبيق حيث نكتب البرنامج ضمن ال ()main

```
static void Main(string[] args)
{
  }
```

يعتبر الاقواس { } أقواس بداية ونهاية البرنامج.

أنماط المعطيات Data Type :

نتعامل مع الكثير من أنماط المعطيات فمنها أنماط عددية ومنها أنماط نصية والجدول التالية يوضح أهم المعطيات المستخدمة:

نمط المعطيات	الاستخدام
١ - الأنماط العددية	
byte	للأعداد التي تقع ضمن مجال [٢٥٥ - ٠]
int	لتمثيل الأعداد الصحيحة الموجبة والسالبة بحجم أعظمي (٣٢ بت)
short	لتمثيل الأعداد الصحيحة الموجبة والسالبة صغيرة الحجم (١٦ بت)
long	لتمثيل الأعداد الصحيحة الموجبة والسالبة التي يتجاوز حجمها (٣٢ بت)والتي تصل الى (٦٤ بت)
double	لتخزين قيم كبيرة (٦٤ بت)وتمثل الأعداد التي تحوي فاصلة عشرية
float	تمثل الأعداد التي تحوي فاصلة عشرية بحجم اعظمي (٣٢ بت)
٢ – الأنماط النصية	
char	لتمثيل محرف واحد يشمل (أعداد – رموز – احرف)
string	لتمثل سلسلة من المحارف
٣ - الأنماط بولانية	
bool	لتصريح عن متحولات تحمل قيم بوليانية منطقية (True , False)

يمكن التعريف عن اكثر من متحول ونفصل بين المتحولات بإشارة "," وعند الانتهاء من التعريف ننهي التعليمة بإشارة " ; " مثال :

```
int x, y, z;
string name;
char ch;
```

عمليات الإدخال والإخراج:

- الإدخال:

يتم ادخال عن طريق التابع ReadLine()

مثال

حيث يتم قراءة النص المدخل على كامل السطر

```
static void Main(string[] args)
        {
            string name, lname;
            name = Console.ReadLine();
            lname = Console.ReadLine();
        }
```

عند تعريف متحول من نمط معطيات غير نمط معطيات ال string

نحتاج الى تحويل قسري مثال:

```
  static void Main(string[] args)
        {
            int num1;
            double num2;
            num1 = int.Parse(Console.ReadLine());
            num2 = double.Parse(Console.Readline());
        }
```

- **الإخراج:**

يتم عرض المعطيات والنتائج على الشاشة السوداء باستخدام التابع (WriteLine) او (Write) حيث يستخدم (Write) للكتابة على نفس السطر اما (WriteLine) يكتب ما داخل الأقواس ثم ينزل سطر جديد وذلك كما يلي:

```
static void Main(string[] args)
        {
            Console.WriteLine("Welcome to ");
            Console.WriteLine("C# Programming!");
            Console.WriteLine("The End");
            int num1;
            num1 = int.Parse(Console.ReadLine());
            Console.WriteLine("Number = " + num1);
        }
```

نتيجة التنفيذ:

```
Welcome to
C# Programming!
The End
4
Number = 4
```

ملاحظة : نستفيد من اشارة + ضمن WriteLine بدمج المعطيات وإظهارها على الشاشة السوداء حيث التعليمة:

```
num1 = 4;
Console.WriteLine(num1);
```

تظهر قيمة المتحول num1 مثل 4

اما التعليمة

```
Console.WriteLine("C# Programming!");
```

يطبع مابين اشارتي التنصيص (" ") . C# Programming!

العمليات الحسابية

- اشارة(=): نستخدم اشارة = للإسناد قيمة الى قيمة

```
int num1;
num1 = 4;
int num2 = 7;
int num3;
num3 = Int.Parse(Console.ReadLine());
```

- اشارة (==): تستخدم للمقارنة

```
int num1;
num1 = 4;
int num2 = 4;
bool t = num1 == num2; // t= true
```

- اشارة (+):تستخدم لعملية الجمع
- اشارة (-):تستخدم لعملية الطرح
- اشارة (*):تستخدم لعملية الضرب
- اشارة (/):تستخدم لعملية القسمة
- اشارة (%):تستخدم لعملية باقي القسمة

y = m x + b÷5(a+ bc);

C#: y = m * x + /5 *(a+ b*c);

أولوية العمليات:

١. بين الاقواس [] و ()
٢. عمليات الضرب والقسمة وباقي القسمة
٣. عمليات الجمع والطرح
٤. واذا العمليات متشابهة من اليسار الى اليمين

Algebra: $z = pr\%q + w/x - y$

C#: `z = p * r % q + w / x - y;`

6 1 2 4 3 5

العمليات المنطقية

اكبر , اكبر او يساوي , أصغر , أصغر او يساوي , يساوي , لا يساوي , النفي

{ < , =< , > , => , == , =! , ! }

Standard algebraic equality and relational operators	**C# equality or relational operator**	**Sample C# condition**	**Meaning of C# condition**
Equality operators			
=	==	x == y	x تساوي y
	!=	x != y	x لا تساوي y
Relational operators			
>	>	x > y	xأكبر من y
<	<	x < y	x أصغر من y
	>=	x >= y	x أكبر أو يساوي y
	<=	x <= y	x أصغر أويساوي y

سلاسل الهروب

\n لإدراج سطر

\t فراغ بقدر Tab أي مقدار ٥ محارف

\r للعودة الى بداية السطر

مثال:

```
Console.WriteLine("Welcome \n to\n C#\n Programming!");
```

```
Welcome
to
C#
Programming!
```

```
Console.WriteLine("Welcome \r  to \t C#\n Programming!");
```

```
To      C#
programming!
```

أمثلة برامج :

١ اكتب برنامج يظهر ناتج ضرب عدد مجهول بالعدد ٥٧

```
static void Main(string[] args)
        {
            int num;
            num = Int.Parse(Console.ReadLine());
            Console.WriteLine("the result " + num * 57);
            Console.ReadLine();
        }
```

٢ اكتب برنامج يقوم به المستخدم بإدخال عددين ويطبع نتائج جمع وطرح وضرب

```
static void Main(string[] args)
        {
            int num1, num2;
            num1 = int.Parse(Console.ReadLine());
            num2 = int.Parse(Console.ReadLine());
            Console.WriteLine("the result + =" + (num1 + num2));
            Console.WriteLine("the result - = " + (num1 - num2));
            Console.WriteLine("the result * = " +( num1 * num2));
            Console.ReadLine();

         }
```

٣ اكتب برنامج يقوم به المستخدم بإدخال نصف القطر الدائرة وبرنامج يطبع مساحة الدائرة(area)ومحيط الدائرة(ocean)

```
static void Main(string[] args)
{
    Console.WriteLine("Enter r");
    double r;
    r = double.Parse(Console.ReadLine());
    Console.WriteLine("area =" + (2 * 3.14 * r));
    Console.WriteLine("ocean= " + (3.14 * r * r));
    Console.ReadLine();
}
```

وظيفة :

١- اكتب برنامج يقوم به المستخدم بإدخال أبعاد مستطيل يطبع مساحة و محيط مستطيل.

٢- اكتب برنامج يقوم به المستخدم بإدخال بعد مربع يطبع مساحة و محيط المربع.

بنى التحكم:

Control Structure

- ✶ بنى شرطية لا تنتهي بفاصلة منقوطة
- ✶ بنى تكرارية

بنى الشرطية:if*

النوع الأول:

if(conditional)
Statement 1;
---------- 	
if(conditional)
{Statement 1;
Statement 2;
…….
}
	اذا كان الشرط محقق ينفذ تعليمة بعد if
	اذا كان الشرط يتطلب تنفيذ عملية واحدة لا نستخدم
	اقواس مجموعة { }
	----- 	
	اذا كان الشرط يتطلب تنفيذ أكثر من تعليمة نستخدم
	اقواس مجموعة { }
	في حال تحقيق الشرط يتم تنفيذ التعليمات المحددة بين
	الاقواس { } وإلا يتم الانتقال الى خارج if

النوع الثاني:

if(conditional)
Statement 1;
else
Statement 1;
---------- 	
if(conditional)
{Statement 1;
Statement 2;
…….}
else
	اذا كان الشرط محقق ينفذ تعليمة if وإذا كان الشرط
	غير محقق ينتقل لينفذ تعليمة داخل
	else
	يتطلب تنفيذ عملية واحدة لا نستخدم اقواس مجموعة {
	{
	----- 	
	اذا كان الشرط يتطلب تنفيذ أكثر من تعليمة نستخدم
	اقواس مجموعة { }.

{Statement 1; Statement 2; …….}	

النوع الثالث:

if(conditional)
Statement 1;
Else if(conditional)
{Statement 1;
Statement 2;
…….}
else
{Statement 1;
Statement 2;
…….}
	اذا كان الشرط محقق ينفذ تعليمة if وإذا كان الشرط غير محقق ينتقل لينفذ تعليمة داخل else If وإذا كان الشرط غير محقق ينتقل لينفذ تعليمة داخل else
	
	اذا كان الشرط يتطلب تنفيذ عملية واحدة لا نستخدم اقواس مجموعة { }
	--- 	
	اذا كان الشرط يتطلب تنفيذ أكثر من تعليمة نستخدم اقواس مجموعة { }.

العمليات المنطقية يستخدم فى حلقات الشرطية

And:يعني ان الشرطين محققين ويرمز لها && مثال:

```
if (x > 1) && (x<10)
```

أي ان العدد يجب ان يكون اكبر من العدد ١ وأصغر من العدد ١٠

OR: تعني ان احد الشرطين محقق ويرمز لها ll (حرف L على لوحة المفاتيح) مثال:

```
if (x > 1) || (x<10)
```

أي ان العدد اما ان يكون اكبر من العدد ١ أو أصغر من العدد ١٠

أمثلة:

١. اكتب برنامج يقوم بإدخال عددين وقسمة العدد الأول على العدد الثاني.

ملاحظة:هذا البرنامج يتطلب اختبار العدد الثاني اذا كان يساوي الصفر أو لا

```
static void Main(string[] args)
      {
          double num1;
          double num2;
          Console.WriteLine("Input the First number");
          num1 = double.Parse(Console.ReadLine());
          Console.WriteLine("Input the Second  number");
          num2 = double.Parse(Console.ReadLine());
          if (num2 != 0)
              Console.WriteLine("the result " + num1 / num2);
          else
              Console.WriteLine("Error because num2=0 ");
          Console.ReadLine();
      }
```

٢. اكتب برنامج يقوم باختبار العدد زوجي أو فردي

```
static void Main(string[] args)
      {
          int x;
          Console.WriteLine(" Input The number");
          x = int.parse(Console.ReadLine());
          if (x % 2 == 0)
              Console.WriteLine(" Even");
          else
              Console.WriteLine(" odd");
          Console.ReadLine();

      }
```

٣. اكتب برنامج يقوم بإدخال معدل طالب

```
static void Main(string[] args)
        {
            int x;
            Console.WriteLine(" Input The mark");
            x = int.parse(Console.ReadLine());
            if (x > 90)
                Console.WriteLine(" Very good");
            else if (x >= 50)
                Console.WriteLine(" good");
            else
                Console.WriteLine(" filled");
            Console.ReadLine();

        }
```

٤. اكتب برنامج يقوم بإدخال ثلاث اعداد صحيحة واختبار أكبر عدد وأصغر عدد

```
static void Main(string[] args)
        {
            int  num1, num2, num3;
            Console.WriteLine("Input the First number" );
            num1 = int.Parse(Console.ReadLine());
            Console.WriteLine("Input the Second  number" );
            num2 = int.Parse(Console.ReadLine());
            Console.WriteLine("Input the Third  number" );
            num3 = int.Parse(Console.ReadLine());
            if ( num1 > num2 &&  num1>num3 )
                Console.WriteLine( "the max number " + num1);
            else if ( num2 > num1 &&  num2>num3 )
                Console.WriteLine( "the max number " + num2);
            else if  ( num3 > num1 &&  num3>num2  )
                Console.WriteLine( "the max number " + num3);
```

```
            // اختبار العدد الأصغر
        if ( num1 < num2 &&  num1<num3 )
              Console.WriteLine( "the min number " + num1);
        else if  ( num2 < num1 &&  num2 < num3  )
              Console.WriteLine( "the min number " + num2);
        else if  ( num3 < num1 &&  num3<num2  )
               Console.WriteLine( "the min number " + num3);
        else
                   Console.WriteLine( "the same number ");
        Console.ReadLine();
    }
```

طريقة ثانية:

```
static void Main(string[] args)
    {
        int  num1, num2, num3;
        Console.WriteLine("Input the First number" );
        num1 = int.Parse(Console.ReadLine( ));
        Console.WriteLine("Input the Second  number" );
        num2 = int.Parse(Console.ReadLine( ));
        Console.WriteLine("Input the Third  number" );
        num3 = int.Parse(Console.ReadLine( ));
        int max = num1;
        if ( num2 > max )
            max=num2;
        if  ( num3 > max  )
            max=num3;
        Console.WriteLine( "the max number " + max);
//      اختبار العدد الأصغر
        int min = num1;
        if  ( min> num2 )
            min=num2;
        if  (min> num3)
            min=num3;
        Console.WriteLine( "the min number " + min);
        Console.ReadLine( );
}
```

بنى الشرطية بنية الاختيار المتعدد :Switch()

تدعى بنية الاختيار المتعدد تستخدم من أجل التعامل مع تنفيذ خيار ما من مجموعة خيارات محتملة وتكون بالشكل التالي:

نلاحظ ان هذه البنية تتألف من مجموعة من الحالات كل حالة تبدأ بالكلمة المفتاحية (case) ثم قيمة المفترض ان تتوافق مع قيمة المتحول أو التعبير حتى يتم تنفيذ التعليمات التي تلي(:) وفي نهاية مجموعة التعليمات الخاصة بحالة ما نكتب االتعليمة(break) حيث ان كلمة break تعني الخروج من كتلة البنية Switch نلاحظ ايضا ان البنية تحوي (default)وهي حالة افتراضية التي يتم تنفيذها في حال لم تتطابق قيمة المتحول مع قيمة أي حالة من حالات ال(case) ملاحظة القيمة التي تلي case يمكن ان تكون أي نوع من انواع المعطيات (رقم ,سلسلة,محرف)	Switch(variable) {case value1:statement1; Statment2: …… break; case value2:statement1; Statment2: …… break; case value3:statement1; Statment2: …… break; …… default:statement1; …… break;}

مثال : اكتب برنامج يطبع اسم يوم من أيام الأسبوع بحسب رقم هذا اليوم من قبل المستخدم

```
static void Main(string[] args)
```

```
{
    int  num;
    Console.WriteLine("Input number" );
    num = int.Parse(Console.ReadLine( ));
    switch (num)
    {
        case 1: Console.WriteLine("Sunday ");
            break;
        case 2: Console.WriteLine("Monday ");
            break;
        case 3: Console.WriteLine("Tuesday ");
            break;
        case 4: Console.WriteLine("Wednesday ");
            break;
        case 5: Console.WriteLine("Thursday ");
            break;
        case 6: Console.WriteLine("Friday ");
            break;
        case 7: Console.WriteLine("Saturday ");
            break;
        default: Console.WriteLine("Out of day ");
            break;
    }
    Console.ReadLine( );

}
```

مثال : اكتب برنامج يقوم بإدخال عددين حقيقين وإدخال العملية الحسابية وتنفيذ العملية على العددين.

```
static void Main(string[] args)
        {
            int num1, num2;
            Console.WriteLine("Input the First number");
            num1 = int.Parse(Console.ReadLine());
            Console.WriteLine("Input the second number");
            num2 = int.Parse(Console.ReadLine());
            char c = char.parse(Console.ReadLine());
            switch (c)
            {
                case '+': Console.WriteLine(" num1+num2 = " + num1 + num2);
                    break;
                case '-': Console.WriteLine(" num1- num2 = " + num1 - num2);
                    break;
                case '*': Console.WriteLine(" num1* num2= " + num1 * num2);
                    break;
                case '/':
                    if (num2 != 0)
                        Console.WriteLine(" num1/num2= " + num1 / num2);
                    else
                        Console.WriteLine("  false");
                    break;
                default: Console.WriteLine("Out of operation ");
                    break;
            }
            Console.ReadLine();
        }
```

اذا كنا تنفيذ أكثر من حلقة ضمن نفس البيئة للقيام بذلك نستبدل تعليمة break بكلمة

goto case value();

حيث value هي قيمة الحالة المراد الانتقال الها ويجب الانتباه الى ان تعليمة caseلا تحوي تعليمة break مثلا لو أردنا طباعة أيام الأسبوع في حال اخترنا يوم من الأيام يطبع كذا يوم تصبح البنية كالتالي:

```
static void Main(string[] args)
        {
            int num;
            Console.WriteLine("Input number");
            num = int.Parse(Console.ReadLine());
            switch (num)
            {
                case 1: Console.WriteLine("Sunday ");
                    break;
                case 2: Console.WriteLine("Monday ");
                    goto case 6;
                case 3: Console.WriteLine("Tuesday ");
                    break;
                case 4: Console.WriteLine("Wednesday ");
                    goto case 7;
                case 5: Console.WriteLine("Thursday ");
                    break;
                case 6: Console.WriteLine("Friday ");
                    break;
                case 7: Console.WriteLine("Saturday ");
                    break;
                default: Console.WriteLine("Out of day ");
                    break;
            }
            Console.ReadLine();
           }
```

وظيفة :

١- اكتب برنامج حل معادلة درجة أولى.

البنى تكرارية

تسمح البنى التكرارية بتكرار مجموعة من الأفعال طالما الشرط المرفق للبنية محقق ويوجد عدة بنى تكرارية

While() , do{}while() , for() ,foreach()

- **While:**

while(conditional) Statement 1; ---------- while(conditional) {Statement 1; Statement 2; ……}	طالما الشرط محقق يتم تنفيذ التعليمة أو كتلة التعليمات التي تلي While تحتاج الى عداد

مثال : اكتب برنامج يطبع الأعداد من ١ الى ١٠

int count=1;
while (count <=10){
Console.WriteLine("Number =
"+count);
count ++;}
	نلاحظ في هذا البرنامج أن الشرط محقق طالما أن قيمة المتحول أصغر أو تساوي العدد ١٠ وبالتالي يتم تنفيذ تعليمات الحلقة وهي طباعة العدد الحالي وعندما تصبح قيمة المتحول ١١ يصبح الشرط غير وبالتالي يتم الخروج من جسم الحلقة.محقق

ملاحظات :

ملاحظة ١:عدم وضع تعليمة في جسم الحلقة يسبب عدم تحقق الشرط المرافق للبنية While ينتج عنه الدخول في حلقة لانهائية من التكرارات في المثال السابق قمنا بزيادة count لتجنب هذه المشكلة.

ملاحظة ٢: في حال استخدام عداد أو متحول يحمل قيمة من الشرط يفضل اعطاؤه قيمة ابتدائية كما في المثال السابق.

ملاحظة ٣: عند الوصول الى بنيةwhile اختبار الشرط ممكن عندها أن يكون غير محقق وبالتالي فإن تعليمات الحلقة يتم تنفيذ ٠ مرة على الأقل.

مثال :

اكتب برنامج يقوم بطباعة الأعداد الأولية من ١ الى ١٠٠

```
static void Main(string[] args)
        {
            int num, i = 1;
            Console.WriteLine("Input the number");
            num = int.Parse(Console.ReadLine());
            while (i <= num)
            {
                if (num % i == 0)
                    Console.WriteLine("Number =  " + i);
                i++;
            }
            Console.ReadLine();        }
```

بنى تكرارية For :

تدعى هذه البنية بالبنية ذات العداد و تتطلب: ١. قيمة بدائية ٢. الشرط ٣. عداد للحلقة (مقدار زيادة او مقدار نقصان) مثلا for(int i=0 ; i<10 ; i++)	for(st1 ; st2 ; st3) Statement 1; for(st1 ; st2 ; st3) {Statement 1; Statement 2; ……}

مثال :

برنامج يقوم بجمع الأعداد من ١ الى ١٠ :

```
static void Main(string[] args)
        {   int sum = 1;
            for (int i = 1; i <= 10; i++)
                sum = sum + 1;
            Console.WriteLine("result sum =  " + sum);
            Console.ReadLine();         }
```

مثال :

برنامج يقوم بطابعة (yes) اذا كان العدد أولي و (NO) اذا غير أولي:

```
        static void Main(string[] args)
        {    int num;
            int count = 0;
            Console.WriteLine("Input the number");
            num = int.Parse(Console.ReadLine());
            for (int i = 1; i <= num; i++)
                if (num % i == 0)
                    count++;
            if (count == 2)
                Console.WriteLine("yes");
            else
                Console.WriteLine("No");         }
```

المصفوفات:

المصفوفة هي عبارة عن سلسلة من البيانات من نفس النوع مثال مصفوفة اعداد صحيحة , مصفوفة محارف .....الخ . ويوجد نوعين من المصفوفات مصفوفة احادية او نسميها ذات بعد واحد ومصفوفة ثنائية أي مصفوفة ذات بعدين

مصفوفة احادية(ذات بعد واحد):

ويمكن التصريح عن مصفوفة كما يلي:

```
int [] array ;
array=new int [size];
```

او نكتب مباشرة:

```
int[] array = new int[size];
```

مثلا تصريح عن مصفوفة عددية اسمها a تحمل ٥ عناصر

```
int[] a = new int[5];
```

او يمكن التعريف عن مصفوفة وإدخال قيم ابتدائية لها

```
int[] a = { 3, 4, 8, 10 };
int[] a = new int[] { 3, 4, 8, 10 };
```

نلاحظ في كلا الحالتين لم يتم ذكر حجم مصفوفة

او يمكن التعريف عن مصفوفة وإدخال قيم ابتدائية لها

```
int[] a = new int[4];
                a[0] = 4;
                a[1] = 5;
                a[2] = 2;
                a[3] = 0;
```

للتصريح عن مصفوفة و ملء خاناتها عبر لوحة المفاتيح كما يلي:

```
static void Main(string[] args)
        {
            int[] a = new int[5];
            for (int i = 0; i < 5; i++)
             a[i] = Int32.Parse(Console.ReadLine());        }
```

نلاحظ أن العداد يبدأ من رقم ٠ لأنه يمثل الدليل العنصر الحالي في المصفوفة ويجب ان يكون الدليل اصغر تماما من حجم المصفوفة

ويمكن طباعة المصفوفة

```
static void Main(string[] args)
        {
            int[] a = new int[5];
            for (int i = 0; i < 5; i++)
                a[i] = Int32.parse(Console.ReadLine());
          for (int i = 0; i < 5; i++)
                Console.WriteLine(a[i]);
        }
```

مثال اكتب مصفوفة ذات بعد واحد وبحجم ٥ عناصر و بإدخال ثابت ثم اطبع المصفوفة بشكل التالي:Array[0]=3 الى نهايتها:

```
        static void Main(string[] args)
        {
            int[] a = new int[5];
            a[0] = 3;
            a[1] = 6;
            a[2] = 8;
            a[3] = 20;
            a[4] = 31;
            for (int i = 0; i < 5; i++)
           Console.WriteLine("Array[" + i + "]= " + a[i]);
        }
```

مثال :

اكتب برنامج يقوم بإدخال مصفوفة أعداد وطباعة مجموع عناصرها وطباعة أكبر عدد واصغر عدد:

// لطباعة مجموع عناصر المصفوفة نحتاج متحول نضع فيه المجموع ومتحول لاكبر قيمة ومتحول لأصغر قيمة:

```
static void Main(string[] args)
```

```
        {
             int[] a = new int[5];
            for (int i = 0; i < 5; i++)
                a[i] = Int32.Parse(Console.ReadLine());
            int max, min, sum = 0;
            max = a[0];
            min = a[0];
            for (int i = 0; i < 5; i++)
            {
                sum = sum + a[i];
                if (a[i] > max)
                    max = a[i];
                if (a[i] < min)
                    min = a[i];
            }
            Console.WriteLine(" Sum = " + sum);
            Console.WriteLine(" Max = " + max);
            Console.WriteLine(" Min = " + min);
        }
```

مثال :

اكتب برنامج يقوم بإدخال مصفوفة أحادية محرفيه وطباعة عناصرها بالشكل التالي

مثال ('A' 'r' 'r' 'a' 'y'):

```
static void Main(string[] args)
        {
            char[] a = new char[5];
            for (int i = 0; i < 5; i++)
                a[i] = char.Parse(Console.ReadLine());
            Console.Write("( ");
            for (int i = 0; i < 5; i++)
                Console.Write("'" + a[i] + "'  ");
            Console.Write(")");
        }
```

المصفوفات:

المصفوفة هي عبارة عن سلسلة من البيانات من نفس النوع مثال مصفوفة اعداد صحيحة , مصفوفة محارف .....الخ . ويوجد نوعين من المصفوفات مصفوفة احادية او نسميها ذات بعد واحد ومصفوفة ثنائية أي مصفوفة ذات بعدين

مصفوفة ثنائية(ذات بعددين):

هذه المصفوفة يميزها بوجد عدد اسطر وعدد أعمدة ويمكن التصريح عن مصفوفة كما يلي:

```
int[,] array;
array = new int[4, 5];
```

او نكتب مباشرة:

```
int[,] array = new int[3, 3];
```

هذه المصفوفة ببعدين تحدد ب ٣ اسطر و ٣ اعمدة نسمي هذه النوع من المصفوفة بمصفوفة مربعة أي ان عدد الأعمدة يساوي عدد الأسطر

اما المصفوفة

```
array = new int[4, 5];
```

فهذه المصفوفة مصفوفة ثنائية

أو يمكن التعريف عن مصفوفة وإدخال قيم ابتدائية لها

1)
```
int[,] a = { { 3, 4, 8 }, { 2, 3, 7 }, { 6, 8, 2 } };
```

2)
```
int [ , ] a =new int [3,3 ];
        a[0,0]=1;
        a[0,1]=5;
        a[0,2]=6; --->>>  a[2,2]=9;
```

للتصريح عن مصفوفة و ملء خاناتها عبر لوحة المفاتيح كما يلي:

```
static void Main(string[] args)
      {
            int[,] a = new int[3, 3];
            for (int i = 0; i < 3; i++)
                for (int j = 0; j < 3; j++)
                    a[i, j] = Int32.Parse(Console.ReadLine());
```

```
    }
```

لطباعة المصفوفة:

```
static void Main(string[] args)
    {
        int[,] a = new int[3, 3];
       for (int i = 0; i < 3; i++)
            for (int j = 0; j < 3; j++)
                Console.WriteLine(a[i, j]);
    }
```

اكتب برنامج يقوم فيه المستخدم بادخال عناصر مصفوفة ثنائية البعد(٣,٣) والبرنامج يقوم بطباعة المصفوفة بالشكل الرياضي .

مثال :

٨ ٧ ٣

١ ٨ ٢

٥ ٩ ٤

// طباعة المصفوفة بشكل رياضي:

```
static void Main(string[] args)
        {
            int[,] a = new int[3, 3];
            for (int r = 0; r < 3; r++)
                for (int c = 0; c < 3; c++)
                    a[r, c] =
Int32.Parse(Console.ReadLine());

for (int r = 0; r < 3; r++)
{
                for (int c = 0; c < 3; c++)

                    Console.Write(a[r, c]);
                Console.WriteLine();
}

        }
```

اكتب برنامج يقوم فيه المستخدم بإدخال عناصر مصفوفة ثنائية البعد(٣,٣) والبرنامج يقوم بجمع المصفوفتين وطباعة النتيجة بمصفوفة ثالثة بشكل رياضي.

```
static void Main(string[] args)
        {
            int[,] a = new int[3, 3];
            int[,] b = new int[3, 3];
            int[,] sum = new int[3, 3];
            for (int r = 0; r < 3; r++)
                for (int c = 0; c < 3; c++)
                    a[r, c] = Int32.Parse(Console.ReadLine());
            for (int r = 0; r < 3; r++)
                for (int c = 0; c < 3; c++)
                    b[r, c] = Int32.Parse(Console.ReadLine());
            for (int r = 0; r < 3; r++)
                for (int c = 0; c < 3; c++)
                    sum[r, c] = a[r, c] + b[r, c];
            for (int r = 0; r < 3; r++)
              {
                    for (int c = 0; c < 3; c++)
                         Console.Write(sum[r, c]);
                    Console.WriteLine();
              }
        }
```

اكتب برنامج يقوم

١- بإدخال مصفوفة مربعة (٣,٣)

٢- حساب مجموع القطر الرئيسي

٣- حاصل ضرب القطر الثانوي

٤- البحث عن عدد مدخل من قبل المستخدم داخل المصفوفة

```
static void Main(string[] args)
        {
            const int n = 3;
            int[,] a = new int[n, n];
            // مصفوفة ادخال
            for (int i = 0; i < n; i++)
                for (int j = 0; j < n; j++)
                    a[i, j] = Int32.Parse(Console.ReadLine());
```

```
        // الرئيسي القطر مجموع حساب
        int sum = 0;
        for (int i = 0; i < n; i++)
            for (int j = 0; j < n; j++)
                if (i == j)
                    sum = sum + a[i, j];
        Console.WriteLine("Sum  " + sum);
        // الثانوي القطر ضرب حساب
        int mid = 1;
        for (int i = 0; i < n; i++)
            for (int j = n-1; j >= 0; j--)
                if (j == (n-1)- i)  // if (i+j==(n-1))  او
                {
                    mid = mid * a[i, j];
                }
        // المصفوفة ضمن عنصر عن البحث
        int find;
        find = Int32.Parse(Console.ReadLine());
        for (int i = 0; i < 3; i++)
            for (int j = 0; j < n; j++)
                if (find == a[i, j])
                {
                    Console.WriteLine("find the number ");

                }
    }
```

المحاضرة الثانية

التوابع:

تستخدم التوابع لمنع تكرار الأكواد

أنواع التوابع : يوجد نوعين من التوابع وهي :

١- توابع لا تعيد قيمة (Void).

٢- توابع تعيد قيمة أما(عددية أو نصية أو بوليانية أو مصفوفة.....)

أولاً : توابع لا تعيد قيمة (Void):

هي توابع لا تحوي نمط معطيات و لا تحوي على تعليمة return و صيغتها الشكل العام:

Public Static void اسم التابع (الوسطاء)

Public Static void fact (int m , int n)

تستخدم معظم هذه التوابع عندما يكون التابع يحوي تعليمات الطباعة

احيانا هذا النوع لا يحتاج الى وسطاء وعند الاستدعاء لا يحتاج الى عبارة الطباعة أو اضافته الى متحول فقط يذكر اسم التابع والوسطاء إذا كان يحوي وسطاء

مثال١:

اكتب تابع يقوم بطباعة جملة Welcome to Program C# خمس مرات.

```
public static void print ( )
            {
            for (int i=0 ; i<5 ; i++)

              Console.WriteLine("Welcome to Program C#");
         }
```

جسم البرنامج

```
static void Main(string[] args)
        {   print();
            }
```

في هذا المثال التابع لا يحتاج الى وسطاء.

مثال٢:

اكتب تابع يقوم بطباعة جملة مدخلة من قبل المستخدم خمس مرات.

```
public static void print (string x )
            {
            for (int i=0 ; i<5 ; i++)

              Console.WriteLine(x);
        }
```

جسم البرنامج

```
static void Main(string[] args)
        {   string n;
Console.WriteLine("input the state ??");
n= Console.ReadLine();
print(n);
Console.WriteLine("input the state anther ??");
n= Console.ReadLine();
print(n);
}
```

مثال٣:

اكتب تابع يقوم بإدخال جملة وادخال عدد مرات طباعة هذه الجملة.

```
public static void print ( string l,int m)
{
            for (int i = 0; i < m; i++)

                Console.WriteLine(l);
        }
```

جسم البرنامج

```
 static void Main(string[] args)
        {
Console.WriteLine("input the word ");
            string m = Console.ReadLine();
Console.WriteLine("input the number ");
            int x = Int32.Parse(Console.ReadLine());
            print(m,x);
           }
```

مثال٤:

اكتب تابع يقوم باختبار إذا كان العدد زوجي أو فردي وطباعة .

```
 public static void chek (int c)
{      if (c % 2 == 0)
           Console.WriteLine (" num "+ c + " is even");
       else
Console.WriteLine (" num "+ c + " is odd");
       }
```

جسم البرنامج

```
      static void Main(string[] args)
        { Console.WriteLine("input the number ");
          int m = Int32.Parse(Console.ReadLine());
          chek(m);
Console.WriteLine("input the number ");
          int n = Int32.Parse(Console.ReadLine());
          chek(n);
          chek(5);
          chek(4);
        }
```

ثانيا : التابع الذي يعيد قيمة

الشكل العام للتوابع:

Public Static نمط المعطيات اسم التابع (الوسطاء)

Public Static int fact (int m , int n)

مثال١:

اكتب تابع يقوم بحساب وطباعة ناتج جمع عددين

```
public static int sum1(int a ,int b)
        { int s = a + b;
            return s;        }
```

جسم البرنامج

```
static void Main(string[] args)
        {
         Console.WriteLine("input the number ");
        int x = Int32.Parse(Console.ReadLine());
        Console.WriteLine("input the number ");
        int y = Int32.Parse(Console.ReadLine());
        int f = sum1(x, y);
        Console.WriteLine("sum number ="+f);
              }
```

مثال٢:

اكتب تابع يقوم بحساب عاملة لعدد معين

```
public static int fact(int a)
{
int f=1;
for(int i=a;i>=1;i--)
  f=f*i
  return f;              }
```

جسم البرنامج

```
static void Main(string[] args)
        {
         Console.WriteLine("input the number ");
        int x = Int32.Parse(Console.ReadLine());
        int f = fact (x);
        Console.WriteLine("fact number " +x+" ="+f);
        Console.WriteLine("input the number ");
```

```
        int y = Int32.Parse(Console.ReadLine());
       f = fact (y);
        Console.WriteLine("fact number " +y+" ="+f);
       f = fact (5);
        Console.WriteLine("fact number " +5+" ="+f);
                    }
```

مثال٣:

اكتب تابع يعيد قوة عدد

```
public static int power ( int n,int m)
        {int f=1;
        for (int i = 0; i < m; i++)

           f=f * n;
       return f;
         }
```

جسم البرنامج

```
        static void Main(string[] args)
        {
            Console.WriteLine("input the number  ");
            int x = Int32.Parse(Console.ReadLine());
            Console.WriteLine("input the number ");
            int y = Int32.Parse(Console.ReadLine());
           Console.WriteLine("result="+ power ( x,y));
       }
```