

الدرس السادس: الحلقة التكرارية for.

رأينا في الدرس السابق كيفية استخدام `while` ، وفي هذا الدرس سوف نتعلم كيفية استخدام `for` لتكرار جمل معينة .

على العموم لا يوجد فرق بينها من ناحية الفعلية ، ولكن الذي يمتاز به `for` أنها أكثر اختصارا من `while` ، خذ المثال التالي :

```
using System ;
class whileloop
{
    static void Main ()
    {
        int i = 1 ;
        while ( i <= 5)
        {
            Console.WriteLine (i);
            i ++;
        }
    }
}
```

يمكنك أن تكتبه بواسطة `for` كالتالي :

```
using System ;
class forloop
{
    static void Main ()
    {
        for ( int i =1; i <= 5; i++)
            Console.WriteLine (i);
    }
}
```

أليس هذه أكثر اختصارا !!!

والآن دعني أشرح لك هذا السطر :

```
for ( int i =1; i <= 5; i++)
```

فهذا السطر يحوي على ثلاث جمل مستقلة و مفصولة بواسطة ;

و تفصيلها كالتالي :

```
for( الجملة الثالثة ; الجملة الثانية ; الجملة الأولى )
```

الجملة الأولى : تقوم هذه الجملة بتحديد المتغير الذي سوف يتحكم في عدد التكرار (المتحكم) ، فإذا لم

يكن موجودا ، فيمكنك الإعلان عنه في هذه الجملة ، وإسناد قيمة إليه مباشرة .

ولكن لاحظ أنك إذا قمت بالإعلان عنه في حلقة التكرارية `for` فإنك لن تستطيع أن تصل إليه بعد الانتهاء من تنفيذ `for` ، وهذا يسمى بالمدى `scope` .

الشفرة التالية تبين ذلك

```
using System;
class checkfor
{
    static void Main()
    {
        for (int i=0; i<= 5; i++);
        Console.WriteLine(i);
    }
}
```

عند محاولتك ترجمتها فإن المترجم سوف يعطي خطأ ، والسبب أن `for` قد كرر جملة فارغة خمس مرات ثم أتت إلى التي تليها ، والذي يريد أن يصل إلى المتغير `i` والذي يعتبر خارج نطاق البرنامج فتولد الخطأ .
الجملة الثانية : وهي الجملة التي يوضع فيها الشرط الذي يجب أن يكون صحيحا حتى تعمل الحلقة التكرارية .

الجملة الثالثة : وهي الجملة التي تتحكم بالزيادة أو النقصان لقيمة المتحكم الذي يتحكم بدورة `for` .

ملاحظات عامة :

1- إن هذه الجمل الثلاث هي جمل اختيارية ، أي يمكنك أن تحذفها ، ولكن انتبه أن لا تجعل الحلقة التكرارية `for` تكرر إلى ما لا نهاية .

2- يمكنك أن تكتب أكثر من عبارة في الجملة رقم واحد و الجملة رقم ثلاثة ، ولكن يجب عليك أن تفصلها بـ (,)

كالتالي :

```
for ( int x=1, int y=4; x<5 ; y--,x++)
```

3- تنفيذ الحلقة التكرارية كالتالي : أولاً تنفيذ الجملة الأولى ثم يتم فحص الجملة الثانية فإذا وجدت أنها صحيحة يتم تنفيذ الجمل التي تحويها `for` ثم بعد ذلك تنفذ الجملة الثالثة وهكذا ..

4- تستطيع أن تكتب أكثر من جملة في `for` ولكن بشرط أن تحتويها بـ { }

كالتالي :

```
for ( int i=1;i<=5;i++)
{
    Console.WriteLine("Hello"+i);
    Console.WriteLine("Just Examle");
}
```

و الآن دعنا نتمرن على كتابة الحلقة التكرارية **for** :

أ- اكتب **for** بحيث يزيد المتحكم من 1 إلى 100 بمقدار 1:

```
for (int i=1; i<=100; i++)
```

ب- اجعل المتحكم يتغير من 100 إلى 1 بحيث ينقص بمقدار 1:

```
for(int i=100; i>=1; i--)
```

ج- اجعل المتحكم يتغير من 7 إلى 77 بمقدار 7 في كل مرة:

```
for(int i=7; i<=77; i+=7)
```

د- اجعل المتحكم يتغير من 20 إلى 2 بمقدار 2 في كل مرة:

```
for(int i=20; i>=2; i -=2)
```

هـ- اجعل المتحكم يتبع السلسلة التالية : 2-5-8-11-14-17-20

```
for(int i=2; i<= 20; i += 3)
```

و- اجعل المتحكم يتبع السلسلة التالية : 0-11-22-33-44-55-66-77-88-99

```
for(int i=99; i <= 0; i -= 11)
```

و الآن لنطبق هذا عمليا ، خذ المثال التالي :

اكتب برنامجا يقوم بجمع الأعداد الزوجية من 2 إلى 100 ، كتالي $2+4+8+10+12+.....$

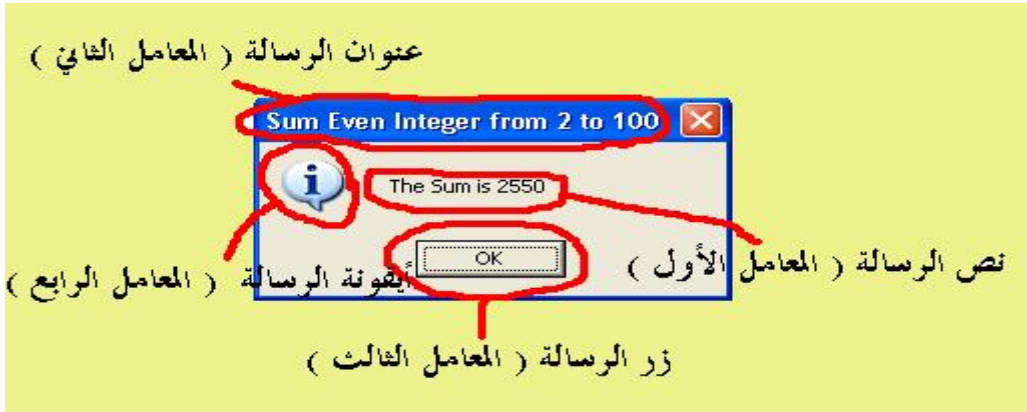
ثم أظهر النتيجة على شكل رسالة ، فكر و حلها بنفسك ؟

هذا هو حلي أنا :

```
// برنامج الأعداد الزوجية
using System;
using System.Windows.Forms; // لا تنسى أن تضع مرجع لهذه اسم الفضاء

class Sum
{
    static void Main()
    {
        int sum = 0; // إعلان عن متغير يحمل قيمة الجمع
        for ( int number = 2; number <= 100; number += 2)
            sum += number;
        // الآن سوف نظهر النتيجة
        MessageBox.Show( " The Sum is " + sum,
            "Sum Even Integer from 2 to 100",
            MessageBoxButtons.OK,
            MessageBoxIcon.Information);
    }
}
```

و النتيجة هكذا :





قد ترغب في المزيد من الشرح عن معاملات الوسيلة **Show** ، فأقول إن المعامل الأول وهو أساسي يحمل نص الرسالة ، أما المعامل الثاني و هو اختياري فيحمل عنوان الرسالة ، أما المعامل الثالث فيحدد نوع الزر الذي سوف يظهر ، و أما المعامل الرابع فيحدد نوع أيقونة الرسالة ، و الجدول التالي يشرح عدة أنواع من الأزرار يمكنك أن تضع واحدة منها في الرسالة :

شكل الزر	عنوان الرز
	MessageBoxButtons.OK
	MessageBoxButtons.OKCancel
	MessageBoxButtons.YesNoCancel
	MessageBoxButtons.RetryCancel
	MessageBoxButtons.AbortRetryIgnore

و الشكل التالي يوضح بعض الأيقونات التي يمكنك استعمالها :

شكلها	عنوان الأيقونة
	MessageBoxIcon.Information
	MessageBoxIcon.Exclamation

	MessageBoxIcon.Question
	MessageBoxIcon.Error

وتفصيل كيفية استعمال **MessageBox** يكون خارج نطاق هذه الدروس .

والآن ما رأيك لو أن مسألة التالية واجهتك ، والمسألة تقول : شخص عنده 1000 سهم ، وكل سهم قيمته دولار واحد ، والشركة التي يستثمر فيها لها عوائد ثابتة تقريبا كل سنة ، و مقدارها 5٪ من قيمة السهم وهذا الشخص يقوم بشراء أسهم جديدة كل سنة بالمبالغ التي يحصل عليها من الشركة ، و الآن احسب قيمة الأسهم مقدرة بالدولار مع نهاية كل سنة ، ولمدة عشر سنين [اعتبر أن كل سهم يساوي دولار لمدة عشر سنوات] ...

استخدم المعادلة التالية لتسهل عليك الحل :

قيمة الأسهم مع نهاية السنة = القيمة الأصلية \times (معدل الدخل كل سنة + 1) عدد السنين

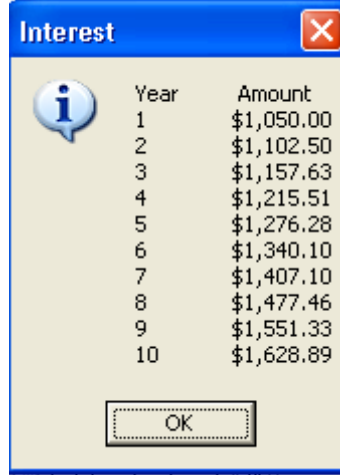
وهذا هو الحل :

```
using System;
using System.Windows.Forms;

class Interest
{
    static void Main()
    {
        decimal amount,principal=(decimal)1000.00;
        //قمنا بتعريف متغيران أحدهما القيمة الكلية و الآخر للقيمة الأصلية//
        double rate= 0.05; // دخل كل سنة
        string output; // متغير يحوي الجمل التي نريد إخراجها
        output = "Year \t Amount \n";
        for (int year =1; year <= 10; year ++)
        {
            amount = principal * (decimal)Math.Pow(1.0 + rate, year);

            output += year + "\t" + String.Format("{0 :c}",amount)+ "\n";
        }
        MessageBox.Show(output, "Interest",
            MessageBoxButtons.OK,
            MessageBoxIcon.Information);
    }
}
```

و النتيجة كالتالي :



Year	Amount
1	\$1,050.00
2	\$1,102.50
3	\$1,157.63
4	\$1,215.51
5	\$1,276.28
6	\$1,340.10
7	\$1,407.10
8	\$1,477.46
9	\$1,551.33
10	\$1,628.89

أتوقع سؤالاً منك يقول : لماذا استخدمنا النوع **decimal** بدلاً من **double** ؟؟؟
سأقول إن **decimal** عادة ما يستخدم للمعاملات المالية حيث أن له دقة عرض تصل إلى 27 خانة بينما **double** 16 خانة فقط ، و هنا نتعامل مع أموال فالأفضل أن نستخدم **decimal** .
و **decimal** أقل مرتبة من **double** فعندما نريد أن نحول **double** إلى **decimal** فيجب عليك استخدام معامل التشكيل ، وقد استخدمناها مرتين : مرة عندما أردنا أن نحول 1000.00 الذي تعتبره لغة السي شارب أنه من نوع **double** إلى **principal** ، ومرة عندما أردنا أن نحول راجع الوسيلة **Pow** و الذي هو من نوع **double** إلى **decimal** كي تتم العملية الحسابية .

هناك مكتبة في الدوت نيت للعمليات الحسابية و تسمى **System.Math** و **Pow** هي إحدى

وسائلها ، و تكتب على الشكل التالي :

<code>Math.Pow(x, 3)</code>	←	x^3
<code>Math.Pow(x, 7)</code>	←	x^7
<code>Math.Pow(x + 7, y)</code>	←	$(x + 7)^y$

و هذه الوسيلة تأخذ معاملات من النوع **double** ، فلذا قمنا بجعل **rate** من نوع **double** حتى نوفر على المترجم عناء تحويل **rate** إلى **double** ثم يجمعها مع 1.0 و الذي هو بدوره من

نوع **double** ، بعد ذلك قمنا بإضافة الناتج إلى **output** لكل سنة ، و تلاحظ أننا استخدمنا **String.Format** لتهيئة الخرج بالشكل المطلوب .

و يمكن أن نتعامل مع هذه الوسيلة كما نتعامل مع وسيلة **WriteLine** في **Console** ، لكن الذي يميزها أنك تستطيع أن تنسق المخرجات بعدة تنسيقات منها تنسيق العملة **{0:c}** و التنسيق على الطريقة العلمية **{0:E}** و غيرها من الطرق ... و الجدول التالي يوضح لك بعض هذه التنسيقات :

شفرة التهيئة	الوصف
C أو c	لتهيئة الرقم ليكون على شكل الذي تكتب به العملة المحلية ، (استخدمناها في البرنامج السابق) ، و يكون تنسيق العملة حسب إعدادات المحلية لجهازك.
D أو d	لتنسيق الأرقام على أساس عدد الخانات و هو مختص بالأعداد الصحيحة فقط.
N أو n	لتنسيق الأرقام بواسطة الفاصلة للألف و خانتين عشريتين بعد النقطة .
E أو e	لتهيئة الأرقام لتصبح على الهيئة العملية بست خانات .
F أو f	لتنسيق الأرقام بعدد ثابت من الخانات العشرية ، عادة 2 .
G أو g	التهيئة عامة ، إما F أو E أيهم أكثر اختصاراً.
X أو x	لتهيئة الأرقام على أساس أنها من النظام الست عشري.

و أفضل طريقة لتوضيح الفرق بين هذه التهيئات هو تجربتها ، قم بتغيير شفرة التهيئة في المثال السابق و اكتشف الفرق بنفسك . الجدول التالي يوضح بعض الفروق ...

الرمز	الوصف	مثال	المخرجات
C أو c	عملة	<code>Console.WriteLine("{0:C}", 2.5);</code>	\$2.50
D أو d	عشري	<code>Console.WriteLine("{0:D5}", 25);</code>	00025
E أو e	علمي	<code>Console.WriteLine("{0:E}", 250000);</code>	2.500000E+005
F أو f	النقطة	<code>Console.WriteLine("{0:F2}", 25);</code>	25.00
G أو g	العائمة	<code>Console.WriteLine("{0:F0}", 25);</code>	25
G أو g	عام	<code>Console.WriteLine("{0:G}", 2.5);</code>	2.5
N أو n	رقمي	<code>Console.WriteLine("{0:N}", 2500000);</code>	2,500,000.00

FA	Console.Write("{0:X}", 250);	ست عشري	x أو X
FFFF	Console.Write("{0:X}", 0xffff);		

الواجب :

- 1- اكتب برنامج يقوم باستقبال خمسة أرقام ثم يحدد أيهم أكبر قيمة؟.
- 2- اكتب برنامج يقوم بطباعة مضروب الأعداد من 1 إلى 20 [مضروب العدد $n = 1 * 2 * 3 * 4 * \dots * n$] بحيث يكون العمود الأول للعدد و الثاني للمضروب ، و لكن بواسطة متغير `int` ، و الثالث للمضروب و لكن بواسطة المتغير `long` ، بحيث تستخدم متغير من نوع `int` لحساب المضروب ، مثل الشكل التالي :

number	n!	n!
1	1	1
2	2	2
3	6	6

ثم أظهر المضروب بواسطة التهيئة العلمية ؟

