الفصل السابع
جمل الدوران والتكرار
Loop Statements

1.7 مقدمة

أن أمكانية تكرار عبارة أو مجموعة من العبارات هو مبدأ أساسي للعديد من الأمور المهمة في عملية البرمجة، وهنا يبرز دور جمل الدوران (التكرار) فبدون هذه الإمكانية فأننا نضطر لكتابة العبارة أو التعبير عدة مرات بالبرنامج مما يصعب التعامل معها بهذه الطريقة مما تؤدي إلى ازدياد عدد المتغيرات كذلك. توفر لغة C++ عددا من جمل التكرار سيتم توضيحها بالفقرات القادمة من هذا الفصل.

2.7 جمل التكرار Loop Statements

1.2.7 جملة التكرار for statement

يمتلك أسلوب التكرار باستخدام for قوة ومرونة ، لا تتوفران في غيرها من اللغات ويمكن كتابة صيغة جملة for كما بالصيغة التالية:

```
for (initial-value;condition;increment) statement;
```

حيث:

initial-value: القيمة البدائية لعداد حلقة for

condition : شرط يقيد حركة for وغالبا ما يحتوي قيمه نهائية

increment: الزيادة المنتظمة في العداد

statement: الجملة المراد تكرارها

عبارة for عبارة ذات إمكانيات كبيرة ومفيدة ومرنة لدرجة عالية ويمكن أن نوجز تنفيذها بثلاث خطوات:

1) تنفيذ العبارة الأولى في رأس الأمر for والتي هي إسناد قيمة ابتدائية للمتغير الذي سيعمل كعداد.

2) تقييم الشرط (حسب قيمته) (true or false) بحيث أذا كانت قيمة الشرط (true) فيتم تنفيذ العبارة/العبارات (statement/s) والتي تمثل جسم الأمر for ، إذا كان جسم التكرار يتكون من أكثر من عبارة واحده ، عند ذلك يجب أن تحدد ككتله بين قوس البداية وقوس النهاية {}. إما أذا كان الشرط خاطئ (false) فسيتم إهمال العبارة / العبارات في جسم أمر التكرار والانتقال إلى تنفيذ الأوامر التي بعده أن وجدت.

3) إما الخطوة الثالثة فهي تنفيذ الجزء الثالث من أمر التكرار for والتي تمثل عداد يعد عدد مرات التكرار التي حدثت سواء كان العداد للزيادة أو للنقصان حيث في كل مره يتم فيها تنفيذ العبارات في جسم حلقة التكرار يتم زيادة أو إنقاص العداد حسب طبيعة الأمر وحسب كمية الزيادة أو النقصان المحدد لكل مرة. وبعد كل عملية تنفيذ لجسم حلقة التكرار يتم العودة إلى الخطوة (2).

مثال : أطبع الإعداد من الرقم 1 إلى 6 باستخدام صيغة التكرار for؟
الحل:

```
for (count=1; count=6;count++) cout<<count<<"\n";
```

تنفذ جملة الطباعة cout<<count ست مرات حيث يأخذا العداد القيم 1,2,3,4,5,6على التوالي ، ثم تطبع قيمة كل منها بسطر مستقل كالتالي:

```
1
2
3
4
5
6
```

مثال: اكتب برنامج لجمع الأعداد الواردة في المثال السابق؟
الحل: نفرض قيمه أولية للمجموع sum تساوي صفرا ثم نضيف أعداد count إلى المجموع نفسه واحدا تلو الأخر مع كل دوره من دورات العداد كالتالي:

```
#include <iostream>
using namespace std;
main ( )
{
        int count,sum;
        sum=0;
  for (count=1;count<=6;count++) sum=sum+count;
  cout<<"sum="<<sum<<endl;
        return 0;
}
```

في هذا البرنامج جمعت الأرقام كالتالي:

sum=0	قبل البداية	
sum=0+1	عندما يكون	count=1
sum=0+1+2	عندما يكون	count=2
sum=0+1+2+3	عندما يكون	count=3
sum=0+1+2+3+4	عندما يكون	count=4
sum=0+1+2+3+4+5	عندما يكون	count=5
sum=0+1+2+3+4+5+6	عندما يكون	count=6

مثال: أكتب برنامج بلغة C++ يقرأ عشرة أعداد ثم يوجد مجموعها؟

الحل:

```
#include <iostream>
using namespace std;
main( )
{
        int count,sum,a;
        sum=0;
  for (count=1;count<=10;count++)
{
          cout<<"enter number\n";
          cin>>a;
          sum=sum+a;
 }
          cout<<"sum="<<sum<<endl;
        return 0;
}
```

ملاحظة: في المثال السابق نلاحظ وجود أكثر من جملة بعد عبارة التكرار for يراد تنفيذها لذلك تم استخدام الأقواس الكبيرة {} للإحاطة بالجمل المراد تكرارها بعد عبارة (for) أذ بالدوران الأول يقوم البرنامج بقراءة العدد الأول (a) ثم يضيفها إلى قيمة المجموع (sum) وكما يلي:

sum=0

sum=0+a_1

sum=0+a_1+a_2

sum=0+a_1+a_2+a_3

……..

……..

sum=0+a_1+a_2+a_3+……+a_{10}

مثال: أكتب برنامج يحسب قيمة x المعطاة بالعلاقة التالية :

x=3*n-8

ولقيم n من 2 ولغاية 4- ؟

الحل:

```
#include <iostream>
using namespace std;
main( )
{
        int n,x;
   for (n=2;n>=- 4;n--)
 {
          x=3*n-8;
          cout<<"n="<<n<<"  x="<<x<<endl;
 }
        return 0;
}
```

ملاحظات:

1) يجب أن يعرف العداد المستخدم مع (for) على أنه عددا صحيحا (integer).

2) الأمر (for) لا ينفذ أكثر من أيعاز أو عبارة واحده تأتي بعدها مباشرة، فإذا كان هناك أكثر من أيعاز يجب أن يكرر ضمن الأمر (for) فيجب أن يحدد بين ({ }) ليكون كتلة.

3) لا يجوز استعمال أسم العداد كاسم متغير بعد الانتهاء من أنجاز عمليات التكرار في البرنامج.

4) في لغة C++ من الممكن أن يكون مكان أي تعبير (expression) في عبارة (for) فراغ كما بالمثال التالي:

```
for  (e1;e2;  )
for (  ; e2  ;  )
```

5) من الممكن تعريف القيمة الابتدائية لعداد حلقة التكرار (for) قبل (أو خارج) الحلقة كما بالمثال التالي.

مثال :اكتب برنامج لجمع ألإعداد من 1-10 ؟

```
int I = 1,sum = 0 ;
for (  ; I <= 5 ; ++I )
sum + = I ;
```

وممكن إعادة كتابة ذات الخطوات أعلاه بطريقة أخرى:

```
int I = 1, sum = 0 ;
for (  ; I <= 5 ;  )
sum + = I ++ ;
```

2.2.7 عبارة التكرار do..while

يستخدم هذا الأمر لتكرار عبارة أو أكثر لعدد من المرات وفقا لمتطلبات البرنامج والتي يحددها المبرمج، في هذا الأمر فان البرنامج سينفذ العبارات بين (do) و (while) على الأقل مرة واحدة ويكون توقف البرنامج اعتمادا على شرط يوضع بعد (while).

التكرار يبدأ بالأمر(اعمل أو كرر) (do) ثم مجموعة من الايعازات المطلوب تكرارها وتنتهي بالأمر(طالما) (while) الذي يكون بعده شرط (أي لغاية عدم تحقق هذا الشرط)، المترجم حين يجد العبارة (do) فأنه سيقوم بإعادة تنفيذ العبارات المحصورة بين هذا الأمر والأمر (while) في كل مرة يصل المترجم إلى الأمر (while)

يفحص الشرط الذي بعده فإذا كان الشرط متحقق فأن المترجم سيعود إلى الأمر (do) ويبدأ بالتنفيذ من الأمر (do) نزولا من جديد إلى الأمر (while) هذه العملية تستمر لغاية عدم تحقق الشرط الصيغة القواعدية لهذا الإيعاز هي:

```
do
{
Instruction 1 ;
Instruction 2 ;
…….
…….
}
while (condition is true) ;
```

مثال : أكتب برنامج يطبع أي حرف يدخل له على أن يتوقف البرنامج عند إدخال الحرف (Y)؟
الحل:

```
include<iostream>
using namespace std;
main( )
{
char YN;
cout << "enter character?";
do
{
//repeat the code for at least one time
cin>> YN;
cout<< YN;
}
while (YN!= 'Y');
return 0;}
```

3.2.7 عبارة التكرار while

وهي أيضا من عبارات التكرار وهي تشابه إلى درجة كبيرة الإيعاز (do..while) إذ أن واجب الأيعازين هو التكرار لمرات غير محددة ابتداء، وإنما يعتمدان على تحقق شرط معين لإيقاف التكرار، الصيغة القواعدية لهذا التكرار :

```
while (condition is true)
{
instruction 1;
instruction 2;
instruction 3;
etc...
}
```

ماذا يعني هذا الأمر (عندما يتحقق الشرط نفذ العبارات التي تلي الأمر while) وفي كل مرة سينفذ الإيعاز أو الايعازات التي بعدة مباشرة والمتعلقة بالأمر (while) ليفحص الشرط هل هو متحقق أم لا فإذا كان متحققا ينفذ وأن كان غير متحقق سيهمل الإيعاز الذي بعد (while) وينفذ ما بعده.

كما هو الحال في (for,if and else) فان الأمر (while) ينفذ عبارة واحدة فقط والتي تأتي بعده مباشرة، إما أذا كان هنا أكثر من عبارة واحدة مطلوب تكرارها ضمن الامر (while) فيجب أن تحدد بين قوسي البداية والنهاية ({ }) لتكون كتله تنفذ جميعها.

خطوات تنفيذ عبارة (while) كما يأتي:

1. التحقق من الشرط بين القوسين .
2. إذا كان الشرط غير متحقق فسوف لا ينفذ المترجم ما موجود في جسم (while) أي لا تكون هناك عملية تكرار ويستمر تنفيذ العبارات التي تلي جسم (while).
3. أما أذا كان الشرط متحققا فسيتم تنفيذ كل العبارات داخل جسم (while) أي كل العبارات المحددة بين قوسي البداية والنهاية للأمر (while) بعدها العودة إلى الخطوة (1) أعلاه.

هذه العملية تسمى تكرار لان الخطوة (3) تعود وتكرر الخطوات (1..3)

ملاحظات:

1) يتم اختيار الشرط بعد الأمر (while) بحيث يساعد حلقة التكرار أن تستمر طالما كان هذا الشرط متحقق، وأن تتوقف الحلقة عن التكرار عندما لا يتحقق هذا الشرط.

2) عند استخدام الأمر (while) فيجب ملاحظة ان المتغير الذي يستخدم معها في الشرط يجب ان تكون له قيمة قبل الدخول الى حلقة (while) وهذه القيمة هي بطاقة الدخول الى حلقة التكرار (while) وبعد الدخول إلى حلقة التكرار يجب ان تتغير قيمة هذا المتغير داخل الحلقة (حلقة التكرار) بما يساعد على إنهاء التكرار.

3) في حالة الأمر (do..while) فأن الشرط يأتي بعد (while) في نهاية حلقة التكرار لذا يجب أن يتم اختياره بحيث عندما يتم فحصة تكون النتيجة (true) أي متحقق، لكي يستمر التكرار بالعمل ومتى ما أصبحت نتيجة فحص الشرط (false) فأن التكرار يتوقف.

4) من السهل كتابة حلقة تكرار بشكل عفوي ، شرطها يصبح متحققا دائما ، هذا سيؤدي إلى برنامج مقفل أو مغلق ويستمر بالتنفيذ إلى مالا نهاية.

4.2.7 مقارنه بين الأمر do..while والأمر while

الجدول أدناه يوضح مقارنه بين الأمر do..while والأمر while :

While	do-while
الشرط في بداية التكرار	الشرط في نهاية التكرار
لا ينفذ أي أيعاز مالم يتم فحص الشرط والتأكد من تحققه	سيتم تنفيذ الإيعاز او الإيعازات المشمولة بالتكرار على الأقل مرة واحدة قبل أن يتم فحص الشرط
تعيد تنفيذ الايعازات المشمولة بالتكرار عند تحقق الشرط	تعيد تنفيذ الايعازات المشمولة بالتكرار عند تحقق الشرط
غالبا ما يستخدم مع طلبات التكرار غير المحددة بعدد ثابت من التكرارات مسبقا	غالبا ما يستخدم مع طلبات التكرار غير المحددة بعدد ثابت من التكرارات مسبقا
يعتمد استمرار التنفيذ على تحقق الشرط ويتوقف التنفيذ عند عدم تحقق الشرط	يعتمد أستمرار التنفيذ على تحقق الشرط ويتوقف التنفيذ عند عدم تحقق الشرط

مثال : أكتب برنامج لإدخال مجموعة أرقام وطباعتها بشرط يتم توقف البرنامج عند إدخال الرقم (0)؟
الحل:

```
#include<iostream>
using namespace std;
main()
{
int x;
cout<< " Enter number";
cin>> x ;
while (x != 0)
{
cout<< x ;
cin >> x ;
}
return 0;
}
```

5.2.7 جمل التكرار المتداخلة Nested Loop Statements

ممكن استخدام جمل التكرار (for,do..while,while) بشكل متداخل لكل واحده منها ولأكثر من مرة وبهذه الحالة فان حلقة تكرر كاملة تتكرر بعدد مرات التكرار المحددة في جملة التكرار الخارجي. من الممكن أن يكون التداخل بين عبارات التكرار جميعا سواء المتشابهات أو المختلفات، مثلا بين (for & for) ، (for & while) (for & do while) ، (while & while do).....الخ.

مثال : أكتب برنامج يحسب معدل اعمار 200 طالب موزعين بالتساوي على خمس شعب بحيث يطبع رقم كل شعبة ومعدل اعمار الطلاب فيها؟

الحل:

```
#include <iostream>
using namespace std;
main( )
{
	int cls,count;
	float age,sum,mean;
	for (cls=1;cls<=5;cls++)
	{
	sum=0;
	for (count=1;count<=40;count ++)
	{
		cout<<"Enter Student No."<<count<<"  age?\n";
		cin>>age;
		sum=sum+age;
	}	//second for
	mean=sum/40;
	cout<<"cls="<<cls<<"    mean="<<mean<<endl;
	}	//first for
	return 0;}
```

ملاحظة:

يستخدم الأمر (break) والأمر (continue) مع حلقات (for) وكافة حلقات التكرار الأخرى مثل (while ,do..while) وكما يلي:

1) الأمر (break) ويستخدم للسيطرة على تدفق تكرار العبارات وهي تؤدي إلى إنهاء أو توقف التكرار عند تحقق شرط معين كما بالمثال التالي:

```
for ( i=1 ; i <= 10 ; i++)
{
cin >> x ;
if  (x < 0)
break ;
else
cout << sqrt ( x ) ;
}
```

في هذه الحالة يتوقف التنفيذ عند ورود عدد سالب لعدم أمكانية أيجاد الجذر التربيعي للعدد السالب

2) الأمر (continue) ويستخدم أيضا مع حلقات التكرار وهو يعني تجاوز تنفيذ بقية الجمل في التكرار خلال الدورة الحالية والانتقال إلى الدورة التالية (أي أستمر مع حلقة تكرار جديدة مع أهمال تنفيذ الأوامر التي بعد الأمر (continue) عند تحقق شرط معين حيث سيعيد المؤشر إلى عبارة (for) كما بالمثال:

```
for ( i=1 ; i <= 10 ; i++)
{
cin >> x ;
if  (x < 0)
continue ;
cout << sqrt (x ) ;
}
```

في هذه الحالة عند ورود عدد سالب فان الامر (continue) سيمنع تنفيذ العبارات الأخرى في حلقة التكرار والمتمثلة بأمر الطباعة في هذا المثال ويعيد المؤشر إلى الأمر (for) ليبدأ بتكرار جديد.

مثال : أكتب برنامج لطباعة الأعداد الزوجية بين 20-2؟

الحل :

```
#include <iostream>
using namespace std;
main ( )
{
        int count;
   for (count=4;count<20;count++) if (count%2==0)
  cout<<count<<endl;
        return 0;
}
```

مثال : أكتب برنامج لطباعة الأعداد الفردية المحددة بالرقمين (35-55)؟

```
#include<iostream>
using namespace std;
main()
{
int i ;
for ( i=35 ; i<=55 ; i++ )
if ( i % 2 == 0)
continue;
cout << i ;
return 0;
}
```

مثال : أكتب برنامج يحسب ويطبع مجموع الحدود في المتسلسلة التالية باستخدام صيغة do-while؟

sum = 1/3 + 1/6 + …….. + 1/15

الحل :

```
#include <iostream>
using namespace std;
main( )
{
	int count;
	float sum,j;
	sum=0;
	count=1;
do
	{
		j=3*count;
		sum=sum+1/j;
		count=count++;
	}
while (count<=5);
cout<<"sum="<<sum<<endl;
	return 0;
}
```

مثال : أكتب برنامج يوجد مجموع الحدود في المتسلسلة التالية :

$$\frac{1}{3}+\frac{3}{5}+\frac{5}{7}+.....+\frac{97}{99}$$

الحل :

```
#include <iostream>
using namespace std;
main ( )
{
	int i;
	float j,k,sum;
	sum=0;
		for (i=1;i<=49;i++)
	{
		j=2*i-1;
		k=j+2;
		sum=sum+j/k;
	}
	cout<<"sum="<<sum<<endl;
	return 0;
}
```

مثال: أكتب برنامج لحل المعادلة

$$\mathbf{y = x^2 + x + 1/x}$$

حيث إن قيمة x تتغير بين (2-4) مع زيادة مقدارها 0.2 في كل خطوة ثم رتب النتائج عند الطباعة بحيث تطبع جميع قيم x وقيم y المقابلة لها؟

الحل:

```
#include<iostream>
using namespace std;
main ( )
{
int I;
double x,y;
cout.setf(ios::fixed);     //floatfield set
cout <<"x value" << "      " << "f(x)\n\n";
for (I = 20 ; I <= 40 ; I+=2)
{
x = I / 10.0;
y = x*x + x + 1/x;
cout.width(7);
cout.precision(1);
cout<<x;
cout.width(16);
cout.precision(10);
cout<<y<<endl;
}
return 0;
}
```

مثال: اكتب برنامج بلغة C++ يحسب مضروب n من المعادلة التالية:

n!=(1) (2) (3) …….(n)

الحل:

```
#include<iostream>
using namespace std;
main ( )
{
int count,n,fact;
fact=1;
cin>>n;
for (count=1;count<=n;count++) fact=fact*count;
cout<<"factorial="<<fact<<endl;
return 0;
}
```

حل نفس السؤال بصيغة while :

```
#include<iostream>
using namespace std;
main ( )
{
int count,n,fact;
fact=1;
count=1;
cout<<"Enter n value\n";
cin>>n;
while (count<=n)
{
fact=fact*count;
count=count++;
}
cout<<"factorial="<<fact<<endl;
return 0;
}
```

مثال: أكتب برنامج C++ لحساب الاقتران e^x والمعرف بالمعادلة التالية:

$$e^x = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \ldots\ldots \frac{x^n}{n!}$$

نفذ البرنامج لقيمة x تساوي 2 وبحيث يصبح الحد الأخير في السلسلة أقل من 10^{-5}؟

الحل:

```
#include<iostream>
#include<math.h>
using namespace std;
main ( )
{
int n,i=1,fact=1;
double x=2,sum=1;
cin>>n;
while (i<=n)
{
        fact=fact*i;
        sum=sum+pow(x,i)/fact;
        if ((pow(x,i)/fact)<=1e-005) break;
        i=i++;
}
cout<<"sum="<<sum<<endl;
}
```

أسئلة حول جمل التكرار

س1/اكتب برنامج يقوم بقراءة درجات 100 طالب في مادة الحاسبات ثم يطبع نسبة النجاح إذا علمت أن درجة النجاح من 50.

س2/اكتب برنامجاً عاماً لحساب مضروب الحدود فيما يأتي بحيث يكون N عدداً صحيحاً موجباً :

$$e^{-1} \bullet e^{-2} \bullet e^{-3} \bullet ..... \bullet e^{-N}$$

س3/ أكتب برنامج لإيجاد معدل مجموعة من الأرقام أخر رقم فيها هو (12).

س4/ اكتب برنامج لإيجاد اكبر وأصغر عدد من بين (10) إعداد ؟

س5/ اكتب برنامج لطبع مجموع المتسلسلة التالية :

$$Y = 2A1 + 6A2 + 12A3 + 20A4 + ... + N$$

س6/ أكتب برنامج يقرأ عددين A,B ويقوم بطباعة الأعداد المحصورة بينهما بطريقة تنازلية.

س7/ أكتب برنامج لإيجاد مجموع الأرقام التي تقبل القسمة على (7) وأخر رقم فيها يساوي (0).

س8/ أكتب برنامج لإيجاد مجموع الحدود في المتسلسلة التالية:

$$\mathbf{Sum} = 1/3 - 1/5 + 1/7 - .... + 1/33$$

س9/ أكتب برنامج لإيجاد مجموع الحدود في المتسلسلة التالية:

$$\mathbf{Sum} = \left(\mathbf{x} - 1/\mathbf{x}\right) + 1/2\left(\mathbf{x} - 1/\mathbf{x}\right)^2 + 1/3\left(\mathbf{x} - 1/\mathbf{x}\right)^3 + ...1/\mathbf{n}\left(\mathbf{x} - 1/\mathbf{x}\right)^{\mathbf{n}}$$

س10/ أكتب برنامج لإيجاد عدد القيم الموجبة في مجموعة من القيم تنتهي بالرقم (0).