كلية العلوم

القسم : **الفيزياء**

السنة : **الثانية**

المادة : لغات البرمجة١

المحاضرة : **الخامسة والسادسة/نظري/**

{{ مكتبة A to Z }}

Facebook Group : مكتبة A to Z

2025

كلية العلوم ، كلية الصيدلة ، الهندسة التقنية

يمكنكم طلب المحاضرات برسالة نصية (SMS) أو عبر(What's app-Telegram)على الرقم 0931497960

الفصل الرابع

بنى التحكم التكرارية (for)

يهدف هذا الفصل إلى تعريف الطالب ببنى التحكم بلغة C++ الحلقية بشكل مفصل من خلال التعرف على كافة اشكال هذه البنى والممثلة بثلاث حلقات ويصبح قادر على استخدام هذه الحلقات التي ستصادفة خلال حياته العملية.

البنية التكرارية for :

The for repetition structure

تسمى بالبنية ذات العدد المعروف من مرات التكرار.

تملك الصيغة العامة التالية :

```
for( initialization ; continuation condition ;update )
statement ;
```

يتم التحكم في هذه البنية التكرارية بثلاثة أجزاء منفصلة:

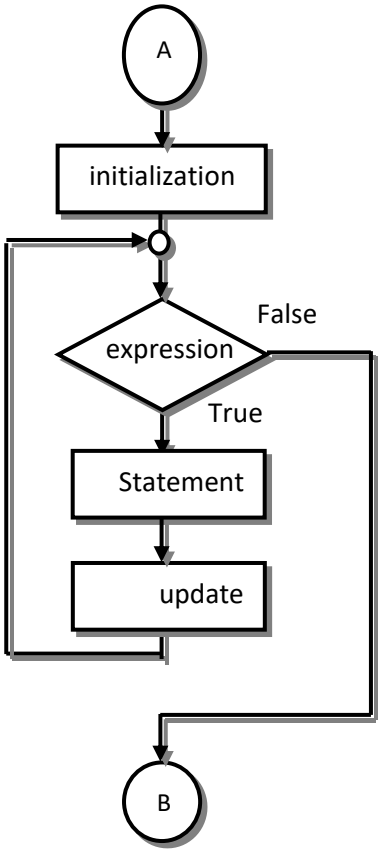
- القيمة الابتدائية initialization.
- شرط الاستمرار continuation condition.
- القيمة الجديدة update.

آلية تنفيذ الحلقة :

يأخذ متحول الحلقة قيمة هي القيمة الابتدائية initialization ويتم اختبار شرط استمرار الحلقة، فإن كان محققاً تنفذ التعليمة statement مفردة كانت أم مركبة، ثم يعطى متحول الحلقة قيمة جديدة ويعاد اختبار شرط استمرار الحلقة فإن كان محققاً تنفذ statement، ثم يعاد إعطاء متحول الحلقة قيمة جديدة وهكذا يستمر تنفيذ statement حتى يختل شرط استمرار الحلقة.

القيمة الابتدائية وشرط استمرار الحلقة والقيمة الجديدة يمكن أن يكونوا فارغين بدون أي قيم.

يبين الشكل (4-7) المخطط الصندوقي للبنية for.



الشكل (4-7) المخطط الصندوقي للبنية for.

مثال 4-13 :

```
#include<iostream.h>
void main(){
     for(int i=1 ; i<3  ; i++ )
          cout<<"*" ;
     cout<<endl;
     cout<<"++"<<endl;}
```

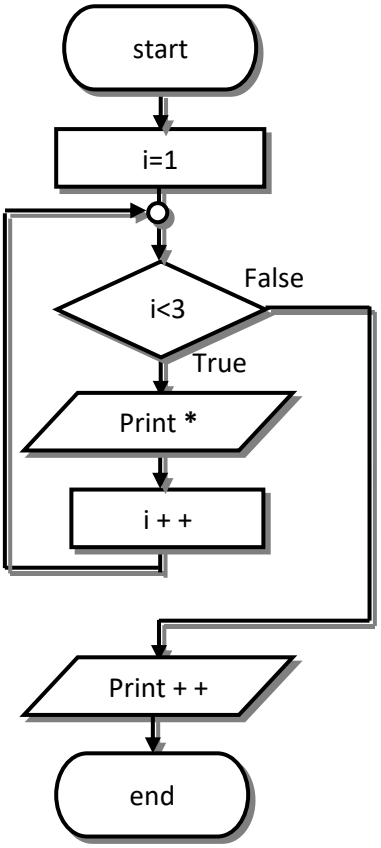
في هذا المقطع البرمجي يأخذ المتحول الصحيح للحلقة قيمة ابتدائية مساوية للعدد 1 وتكون نتيجة اختبار شرط استمرار الحلقة (1<3) true لذلك ستنفذ

```
**

++
```

التعليمة ;'*'<<cout ويتم طباعة المحرف * ، ثم تزاد بعد ذلك قيمة المتحول i وقيمة الزيادة هنا خطوة واحدة (يمكن أن تزاد بأي قيمة صحيحة) ومن ثم يعاد اختبار شرط استمرار الحلقة (3 > 2) والذي تكون قيمته true ويطبع * ، ومن ثم تزاد قيمة المتحول i فتصبح 3 وهنا يختل شرط استمرار الحلقة ويتم الخروج من for. ويكون خرج البرنامج:

ويبين المخطط(4-8) خوارزمية العمل:



المخطط الانسيابي(4-8)

مثال 4-14 :

```
#include<iostream.h>
void main(){
     for(int i=2;i>0;i--)
          cout<<"*";
     cout<<endl<<"++"<<endl;}
```

في هذا المثال القيمة الابتدائية للحلقة أكبر من القيمة النهائية والعداد i يتم إنقاصه حتى يصل للقيمة النهائية.

سيعطي هذا البرنامج خرجاً كالبرنامج السابق.

مثال 4-15:

البرنامج التالي يدخل متتالية من الأعداد الصحيحة تنتهي بالعدد الصحيح صفر ويطبع أكبر وأصغر الأعداد في المتتالية وذلك بالاعتماد على البنية for.

وشرط استمرار حلقة for في البرنامج الثاني أن يدخل عدد مغاير للصفر.

لايوجد عداد متغير في حلقة البرنامج الأول ولا يوجد قيمة جديده في حلقة البرنامج الثاني.

```
#include<iostream.h>
void main(){
      int n،min،max;

      cout<<"enter positive integers،";
      cout<<" terminate input with 0:\n";
      cin>>n;
      for(min=max=n;n!=0;)
            {
                  if(n<min)
                        min=n;
                  else
                        if(n>max)
                              max=n;
                  cin>>n;}
      cout<<"min="<<min;
      cout<<"  "<<"max="<<max<<endl;
}
```

مثال 4-16:

أكتب برنامجاً يطبع مكعبات الأعداد المدخلة إلى أن يدخل العدد صفر بالاعتماد على البنية for.

```
#include<iostream.h>
void main(){
       int n;
      cout<<"Enter a positive integer،";
      cout<<"0 to end:";
      cin>>n;
```

```
for(  ; n!=0 ; )
{     cout<<n<<" cubed is:";
      cout<<n*n*n<<endl;

      cout<<"Enter a positive integer،";

      cout<<" 0 to end:";
      cin>>n;}}
```

خرج البرنامج:

```
Enter a positive integer,0 to end :5

5 cubed is:125

Enter a positive integer,0 to end :6

6 cubed is:216

Enter a positive integer,0 to end :2

2 cubed is:8

Enter a positive integer,0 to end :3

3 cubed is:27
```

مثال 3-24 :

أكتب برنامج يطبع القيم بين العدد المدخل والعدد 10 سواء كان ذلك العدد أكبر أو أصغر من العدد 10، وذلك بالاعتماد على البنية for.

```
#include<iostream.h>
void main(){
      int counter;
      cin >> counter;
      if (counter <= 10)
            for(counter=counter;counter<=10;counter++)
                  cout<<counter<<endl;
      else
            for(counter=counter;counter>=10;counter--)
                  cout << counter << endl;}
```

مثال 4-17:

أكتب برنامج يقوم بما يلي :

1-طباعة مجموع الأعداد المحصورة بين العدد 1 والعدد 100.

2-طباعة مجموع الأعداد الزوجية المحصورة بين العدد 1 والعدد100.

3-طباعة مجموع الأعداد الزوجية التي تقبل القسمة على العدد 5 والمحصورة بين 1و100.

4-طباعة مجموع الأعداد المحصورة بين عدد مدخل والعدد 10.

```
#include <iostream.h>
void main(){

      int sum=0،sum1=0، sum2=0،sum3=0;

      int sum0=0،sum4=0;
      cout<<"the sum of the number between 1 and 100 is:";
      for(int number=1;number<=100;number +=1)
            sum += number;
      cout <<sum<<endl;

      cout<<"the sum of the even number between 1 and 100
      is:";
      for(int number2=2;number2<=100; number2 +=2)
            sum2 += number2;
      cout<<sum2<<endl;
      cout<<"the sum of the even number"
            <<" between 1 and 100 that divide 5 is:";
      for(int number1=2;number1<=100;number1 +=2)
            if(number1%5 == 0)
                  sum1+=number1;
      cout<<sum1<<endl;
      cout<<" enter number: ";
      cin >>sum0;
      cout<<" the sum of the numbers between "
              <<sum0<<"and 10 is:";
      if(sum0>=10){
      for( number2=sum0;number2>=10;number2 -=1)
            sum4 += number2;
            cout<<sum4<<endl; }
      else {
            for(number2=sum0;number2<=10;number2
            +=1)sum4 += number2; cout<<sum4<<endl; } }
```

1-الحلقات المتداخلة :The nested loops

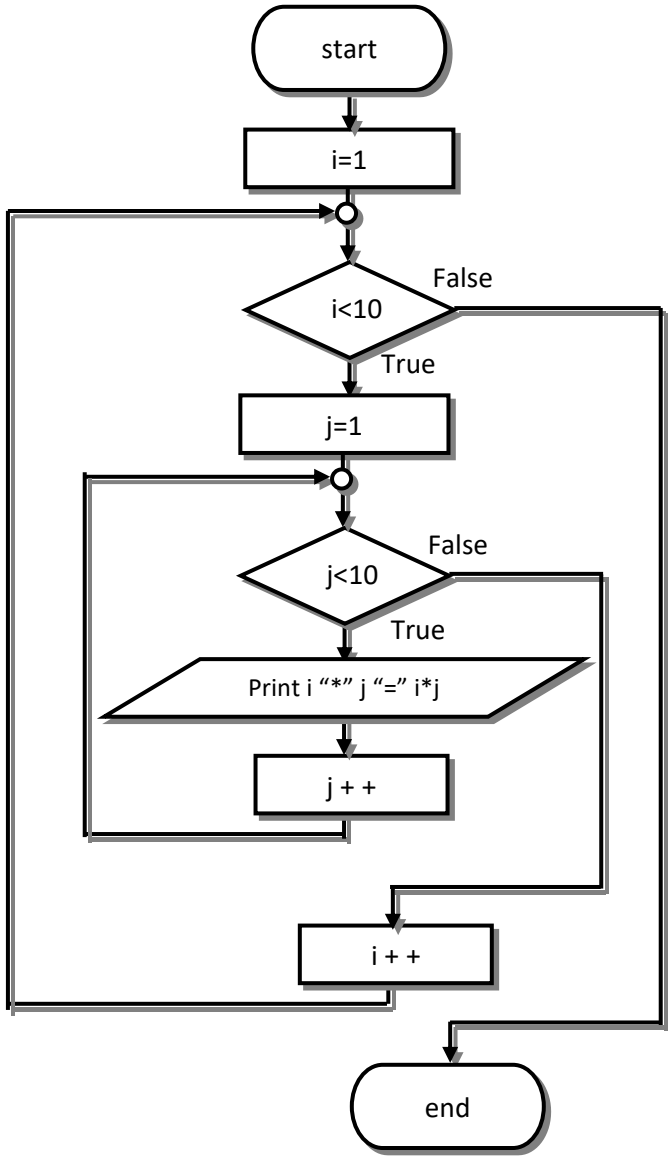
عندما يتطلب الأمر إجراء حلقة لكل قيمة من قيم حلقة أخرى يكون من الضروري وضع حلقة ضمن حلقة أخرى وهذا ما يدعى بتداخل الحلقات، وفي هذه الحالة فإنه من الضروري إنهاء الحلقة الداخلية قبل إنهاء الحلقة الخارجية.

مثال 5-1 :

اكتب برنامج يعطي جدول الضرب للأعداد من 1 حتى 9 بالأعداد من 1 حتى 9.

```
#include<iostream.h>
void main(){
     for(int i=1;i<10;i++)
           for(int j=1;j<10;j++)
             cout<<i<<"*"<<j<<"="<<i*j<<endl;}
```

وهنا يتم تنفيذ الحلقة الداخلية من أجل كل قيمة لـ j من 1 حتى 9 (تسع مرات) ويتكرر ذلك بمجمله أيضاً من أجل كل قيمة لـ i من 1 وحتى 9 (أيضا تسع مرات)، ويبين المخطط الانسيابي (5-1) سير العمل.



المخطط الانسيابي(5-1) للحلقات المتداخلة

مثال 5-2 :

بفرض لدينا عدد من القوائم والمطلوب كتابة برنامج لحساب معدل الأعداد في كل قائمة.

```
#include <iostream.h>

void main (){
      int n‹x‹count‹loops‹loopcount;

      float average‹sum;

      cout<<"how many lists?";
      cin>>loops;

      for(loopcount=1;loopcount<=loops;++loopcount)
      {
            sum = 0;
            cout<<"\n list number"
                  <<"\n how many numbers?";
            cin>>n;

            for(count=1;count<=n;++count)
            {
                  cout<<"x=";
                  cin >>x;
                  sum +=x;
            }

            average = sum/n;
            cout<<"\n the average is:"<<average<<endl;}}
```

خرج البرنامج:

```
how many lists?  2
list number
how many numbers? 3
x=1
x=6
x=4

tha average is:3.66667

list number
how many numbers? 5
x=1
x=3
x=7
x=9
x=6
```

في هذا البرنامج تم إدخال عدد القوائم من خلال المتغير الصحيح loops،عدد العناصر في كل قائمة تم إدخالها من خلال المتغير n ، عناصر كل قائمة تم إدخالها من خلال المتغير x.

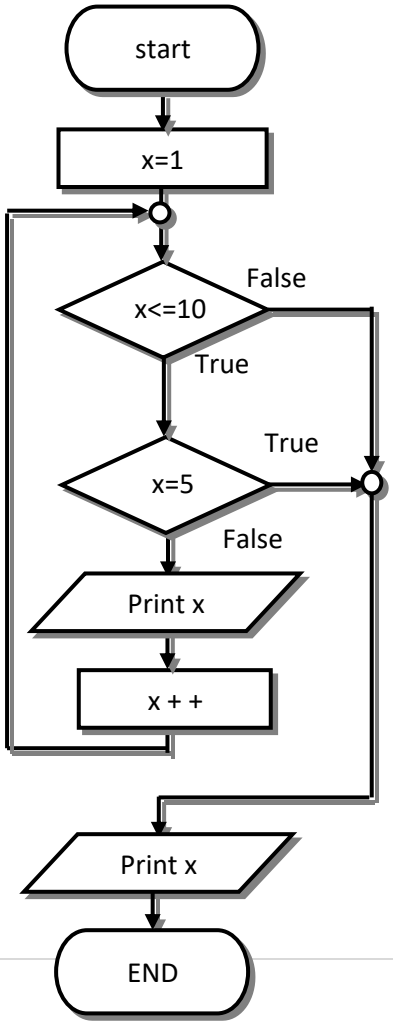
5-2- التعليمتان continue،break :

تستخدم التعليمتان break وcontinue من أجل تغيير مجرى التحكم ضمن البرنامج.

أ- التعليمة break:

تسبب تعليمة break عند تنفيذها مع إحدى البنى التكرارية for،while،do\while ومع بنية الاختيار switch الخروج مباشرة من البنية ويتابع البرنامج بعدها تنفيذه مع أول تعليمة تليها مباشرة.

نستخدم تعلية break من أجل قطع تنفيذ الحلقة عند القيمة 5 في المثال التالي ونرسم المخطط الانسيابي كما في الشكل (5-2) أولاً ثم نكتب البرنامج.



مثال 5-3:

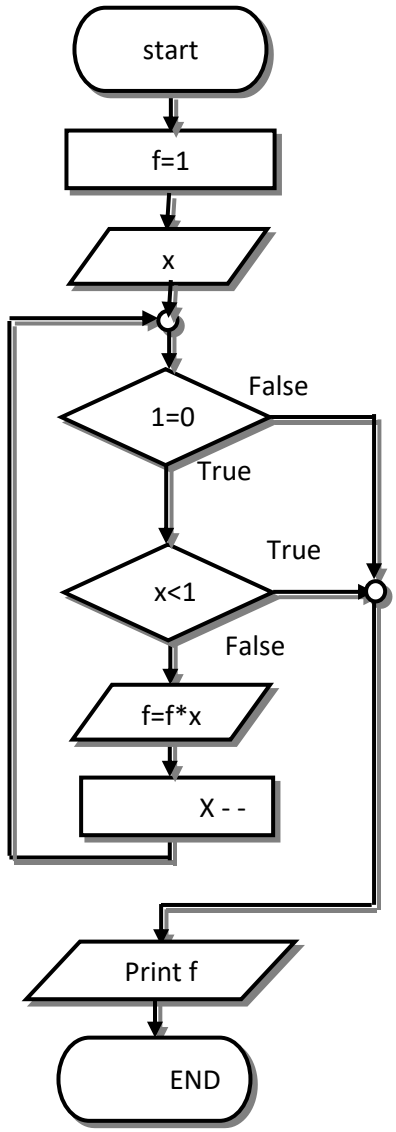
```
#include<iostream.h>
void main()
{
	for(int x=1;x<=10;x++)
	{
		if(x==5)break;
		cout<<x<<" ";
	}
	cout<<endl<<"broke out of loop at x=="<<x<<endl;
}
```

عندما يصبح x=5 يتم الخروج من حلقة for وهذا ما يؤكده خرج البرنامج:

```
1 2 3 4
broke out of loop at x == 5
```

تنفذ حلقات لانهائية لا يمكن الخروج منها إلا باستخدام تعليمة break.

```
}
```



المخطط الانسيابي(5-3) يبين كيفية تنفيذ التمرين 5-5

مثال 5-6 :

البرنامج التالي يحسب n! باستخدام for.

```
#include<iostream.h>
void main()
{
```

```
int x،f=1;
cout<<"enter a positive integer number:";
cin>>x;
for( ; ; )
{
      if(x<1) break;
      f=f*x;
      x--;  }
cout<<x<<" factorial is:"<<f<<endl;}
```

ب- التعليمة continue :

تسبب التعليمة continue عند تنفيذها مع أحد البنى while،do\while ،for ، في تجاوز ماتبقى من جسم البنية التكرارية والمتابعة مع المرور التالي للبنية.

يتم في البنيتين while،do\while القيام بالتحقق من صحة شرط استمرار التكرار مباشرة بعد تنفيذ التعليمة continue.

أما مع البنية for فيتم القيام بعملية الزيادة أولاً ثم يبدأ بعدها تقييم شرط استمرار الحلقة.

ذكرنا سابقاً أن البنية while يمكن أن تستخدم في كثير من الحالات بدلاً من البنية for ، لكن ذلك غير ممكن في حال وجود تعليمة continue قبل تعليمة زيادة قيمة متحول التحكم بالتكرار داخل البنية while.

في الحالة المذكورة لا يتم تنفيذ تعليمة الزيادة قبل التحقق من صحة شرط الاستمرار بالتكرار مما يتناقض مع جوهر عمل البنية for.

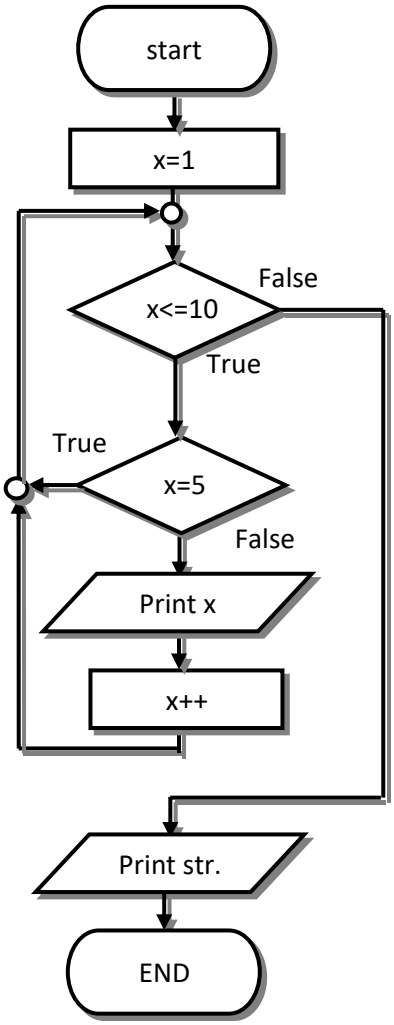
مثال 5-6 :

البرنامج التالي يوضح عمل تعليمة continue مع البنية for

```
#include<iostream.h>
void main(){
      for(int x=1;x<=10;x++){
                  if(x==5)continue;
                  cout<<x<<" ";}
      cout     <<endl
          <<" used continue to skip printing the value 5"
            <<endl ;}
```

خرج البرنامج:

1 2 3 4 6 7 8 9 10
used continue to skip printing the value 5



المخطط الانسيابي(5-4) يبين كيفية تنفيذ التمرين 5-8