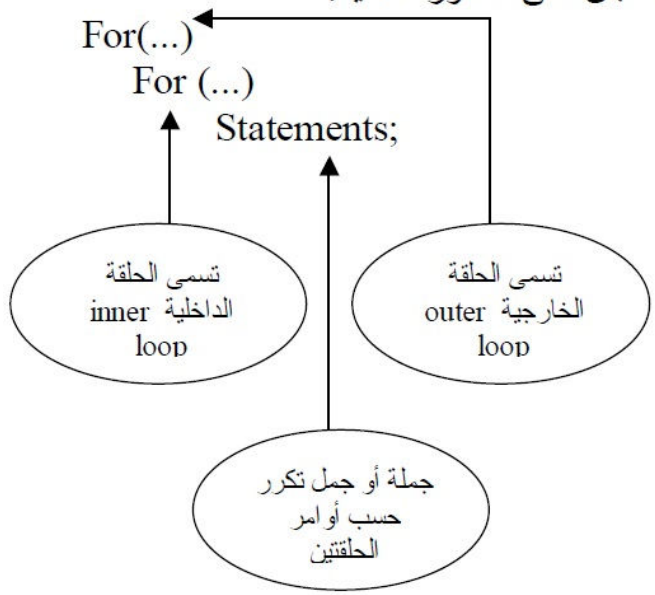
الحلقات المتداخلة

تأخذ الحلقات For المتداخلة الشكل العام التالي :

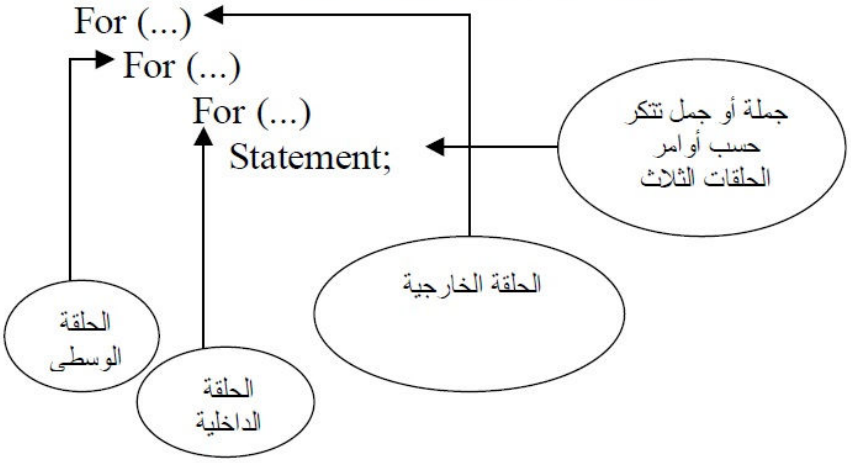
```
for (.....)
{
 for (.....)
 {

  statements;
 }
}
```

فلو أخذنا حالة حلقتين متداخلتين فانهما تكتبان على الصورة التالية:



وفي حالة الثلاث حلقات المتداخلة ، فإنها تكتب على الصورة التالية:



مثـــــال /

```
#include<iostream.h>
main( )
{
int i,j;
for (i=1 ; i<3;++i)
{
for (j=1 ; j<4;++j)
 {
 cout << i*j<<' ' <<endl;
  }
}
}
```

Output :

```
1 2 3 4
2 4 6 8
6 9 12
```

مثـــــــال /

```
#include<iostream.h>
main()
{
int,i,j;
for(i=1;i<=4;i++)
{
for(j=1;j<=i;j++)
{
cout<<"* ";
}
cout<<endl;
}
}
```

Output

```
*
* *
* * *
* * * *
```

مثـــــــال /

```
#include<iostream.h>
main()
{
int,i,j;
for(i=4;i>=1;i--)
{
for(j=1;j<=i;j++)
{
cout<<"* ";
}
cout<<endl;
}
}
```

Output

```
* * * *
* * *
* *
*
```

مثـــــال /

```
#include<iostream.h>
main()
{
int i,j,k=1;
for(i=1;i<=4;i++)
{
for(j=1;j<=i;j++)
{
cout<<k<<" ";
k++;
}
cout<<endl;
}
}
```

Sample Output

```
1
2 3
4 5 6
7 8 9 10
```

مثـــــــــال :

```
#include<iostream.h>
main()
{
int i,j,n;
cout<<"Enter the value of n : ";
cin>>n;
cout<<"The number triangle :\n";
for(i=1;i<=n;i++)
{
for(j=1;j<=i;j++)
{
cout<<j<<" ";
```

```
}
cout<<endl;
}
}
```

Sample Output :

```
Enter the value of n : 5
The number triangle :
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
```

المتغيرات المرقمة والمصفوفات
Arrays and Matrices

مقدمة introduction

أن طرق التعامل مع أسماء المتغيرات والثوابت العددية والرمزية ، التي وردت في الفصول السابقة ، تعد صالحة للتعامل مع عدد محدود من هذه الثوابت والمتغيرات ، سواء في عمليات الإدخال والإخراج أو في العمليات الحسابية والمنطقية ، وعندما يصبح عدد المتغيرات كبيرا جدا ، تصبح تلك الطرق غير عملية ، فمثلا لو أردنا إدخال مائة قيمة للمتغيرات x2,x1- .... إلى x100 ، فكم الحيز المطلوب من البرنامج لعمليات الإدخال والإخراج والعمليات الحسابية والمنطقية لهذه المتغيرات ؟ هذا من جهة ، ومن جهة أخرى : فأننا نوفر مخزنا خاصا لكل متغير نتعامل معه ، أثناء تنفيذ البرنامج ، ولذلك لحفظ قيمته في مخون ، ومن ثم لاستعمال قيمته في عمليات أخرى تالية ، ومن ناحية ثالثة ، فان من الصعوبة بمكان ، بل من المستحيل استعمال اسم المتغير العددي أو الرمزي كمصفوفة ذات بعدين ، وثلاثة أبعاد ...الخ

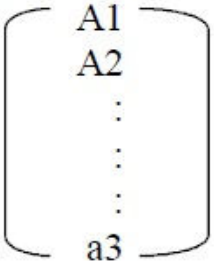
للأسباب الثلاثة الواردة أعلاه ، جاءت فكرة استعمال متغير جماعي يضم تحت اسمه عددا من العناصر يسمى بالمتغير الرقمي subscripted variable ، ويتم ترقميه بين قوسين مربعين [ ] يوضع بينهما قيمة العداد المرقم subscript ، وقد تسمية الدليل index أحيانا ، ويمكننا تشبيه المتغير المرقم بقسم الهاتف لمؤسسة ما ، فهو مقسم واحد ، تنظم تحته عدد من الأرقام الفرعية للموظفين وكل رقم من هذه الأرقام مستقل ومتميز عن الأرقام الفرعية الأخرى ، وله مخزن خاص في الذاكرة ، الآن انه كغيره من الأرقام الفرعية تابع للرقم العام لمقسم المؤسسة ، كما يمكن تشبيه المتغير المرقم بالجيش الذي يعامل كاسم متغير واحد ، لكن يضم عددا كبيرا من العناصر ، فمثلا العناصر التالية :(من اليمين إلى اليسار):

A[n] ...a[2], a[1], a[0]

تابع للمتغير الجماعي a[ ]
وكل عنصر من هذه العناصر له عنوان في الذاكر address ، فالعنوان الأول يكون للعنصر الأول والثاني للثاني والثالث للثالث ... وهكذا.
ويستعمل المتغير الجماعي [المرقم] أو المصفوفة ، في لغة c++ وغيرها ، حجز جماعي مسبق في الذاكرة لجميع عناصره ، فلو كان يتبعه خمسون عنصرا ، فانه يحجز له 50 مخزنا ، على الأقل في الذاكرة .

من الفوائد المهمة للمتغيرات المرقمة والمصفوفات : هو استعمالها في الترتيب التصاعدي والتنازلي للعناصر والقيم المختلفة ، وعمليات ترتيب الأسماء الأبجدي

النصوص الرمزية ، وفي عمليات ضرب المصفوفات ، وإيجاد معكوس المصفوفة وعملياتها الأخرى ، وفي التحليل العددي ... الخ.
المتغير المرقم (المصفوفة) ذو البعد الواحد one-dimensional Array
المتغير المرقم ذو البعد الواحد هو مصفوفة ذات بعد واحد أو متجه (vector) ويمثل في الجبر على النحو الأفقي [a1 a2 ...a3]
أو العمودي



ويأخذ المرقم المتغير في c++ الشكل العام التالي:

```
Type-specifier array-name[size];
```

ويبدأ العداد المرقم عادة من الصفر ، آي أن العنصر الأول من المصفوفة []a هو a[0] والثاني a[1] ... وهكذا فمثلا المصفوفة التالية:

Int a[20];

اسمها a ، وقد حجز لها 20 موقعا لعشرين عنصرا من النوع الصحيح .

والمصفوفة التالية:

Char name[15];

مصفوفة رمزية ، اسمها name يحجز لها خمسة عشر عنصرا من النوع الرمزي لها .

وهكذا...

عنوان عناصر المصفوفة في الذاكرة Addressing Array Elements in Memory

ذكرنا من قبل أن آي متغير أو عنصر من متغير ذاتي مرقم ، يحتل موقعا من الذاكرة يستعمل عادة مؤشرا لكل متغير أو عنصر ، ليكون دليلا على استعمال هذه المتغيرات والعناصر بسهولة ويسر ، والمثال التالي يوضح هذه العملية بالنسبة للمصفوفة ذات بعد واحد.

Int x[5];

يمكن تمثيل عناصر المصفوفة x المعلن عنها ، مع عناوينها بالشكل التوضيحي التالي (من اليسار إلى اليمين)

الأول	الثاني	الثالث	الرابع	الخامس
X[0]	X[1]	X[2]	X[3]	X[4]
100	101	102	103	104

مثال/ على عملية إدخال ذاتي لقيم عناصر متغير مرقم مصفوفة ذي بعد واحد

```
#include < iostream.h>
main ()
{
int a[20];
int I;
for (I=0;I<20;++I)
 {
    a[I]=I+1;
 }
}
```

في هذه الحالة يتم إدخال عشرين عنصرا من عناصر المصفوفة a
A[0]=1 عندما يكون I=0
A[1]=2 عندما يكون I=1
...
...
...
a[19]=20 عندما يكون I=19

```
#include < iostream.h>
main ()
{
int x[5], y[5];
int I;
for (I=0;I<5;++I)
{
x[I]=I;
y[I]=I*I;
cout<<endl<<x[I]<<y[I];
}
}
```

وستكون قيم النتائج على النحو التالي:

0	0
1	1
2	4
3	9
4	16

إعطـاء قيمة أولية للمصفوفة ذات البعد الواحد Array Initialization

مثال على إدخال عدة عناصر من مصفوفة الدرجات grade[]

```
Int grade[5]={80,90,54,50,95}
```

ومثال على إدخال قيم عناصر المصفوفة الرمزية name[]

```
Char name[4]="nor"
```

لاحظ أن المتغير المرقم name[] مكون من أربعة عناصر بينما تم إعطاؤه ثلاثة عناصر فقط والسبب أن العنصر الرابع بالنسبة إلى المعطيات الرمزية يكون خاليا.

```
#include < iostream.h>
main ()
{
int a[6]={40,60,50,70,80,90}
int I;
for(I=0;I<6;I++)
{
cout<<a[I]<<endl;
}
}
```

والناتج طبعا سيكون كالتالي :

```
40
60
50
70
80
```

90

مثال ٤:
قم بكتابة برنامج يقوم بإيجاد مجموع ، ومعدل علامات الطالب في 5 مواد وهذه العلامات كالتالي:
87,67,81,90,55

```
#include < iostream.h>
main ()
{
int i;
int a[5]={87,67,81,90,55}
int s=0;
float avg;
for(i=0;i<5;i++)
{
s=s+a[i];
}
 avg=s/5;
cout<<avg<<endl;<<s<<endl;
}
```

والناتج سيكون كالتالي:

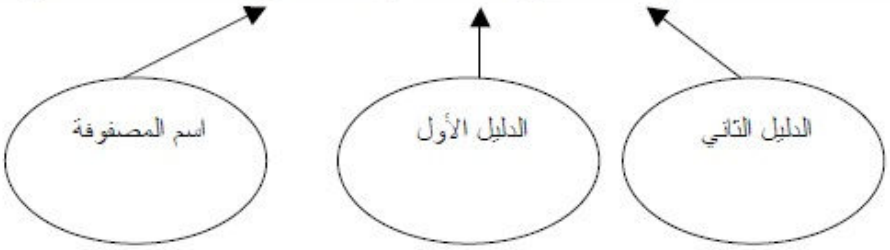
87
735

المعدل 87
والمجموع 735

المصفوفة ذات البعدين Two-Dimensional Arrays

تشبة المصفوفة ذات البعدين في طريقة تعاملها ، المصفوفة ذات البعد الواحد إلا أن لها عداديـن(index2) دليليـن أو مرقميـن إحداهمـا عـداد للصـفوف ، والأخـر عـداد للأعمدة ويأخذ الإعلان عن المصفوفة الشكل العام التالي:

```
Type-specifier arraey_name [index 1][index 2];
```



فمثلا المصفوفة :

```
Int x[2][3];
```

وهي مصفوفة صحيحة العناصر int أبعادها هي عدد الصفوف=2 ، وعدد الأعمدة =3

لاحظ أن عدد الصفوف يوضع بين قوسين وحده ، وكذلك عداد الأعمدة .

مثال / شاهد هذا المثال الذي يستخدم 5 طلاب و 3 علامات:

```
#include <iostream.h>
main ()
{
int m[5][3];
int I,j;
for(I=0;I<5;I++)
 {
for(j=0;j<3;j++)
  {
    cin>>m[I][j];
  }
 }
}
```