المحاضرة الأولى

المصفوفات:

هي عبارة عن مجموعة ذات حجم ثابت من العناصر التي لها نفس نوع البيانات ولكنها تختلف في القيم المُسندة إليها.

يحتاج المبرمج في كثير من الأحيان إلى تخزين البيانات بعد قراءتها بهدف الرجوع إليها في إجراء بعض العمليات، وعندها يحتاج إلى التصريح وبشكل منفصل عن عدة متحولات لها نفس نوع البيانات مثلاً num1, num2, ...num50 ، وهنا تبرز أهمية المصفوفات في تخزين قيم عدة متحولات من نفس النوع ضمن تسلسل معين مما يسهّل الوصول إليها والتعامل معها لاحقاً، حيث يتم تعريف متحول واحد من نوع مصفوفة وليكن على سبيل المثال numbers ، ثم يتم التعامل مع عناصر هذه المصفوفة بشكل مستقل عن طريق الـ Index (الدليل)الخاص بهذا العنصر

numbers[0], numbers[1], numbers[2],.....

أنواع المصفوفات:

- المصفوفات أحادية البعد (vector).
- المصفوفات ثنائية البعد.

التصريح عن مصفوفة أحادية البعد: يتم التصريح عنها بالشكل التالي:

```
type  arrayname[n];
```

حيث:

Type : نوع عناصر المصفوفة (int , float , char, ...).

arrayName : اسم المصفوفة.

n : عدد عناصر المصفوفة

مثال: int arr [5] ;

هنا تم التصريح مصفوفة اسمها arr مؤلفة من 5 أعداد صحيحة.

للوصول إلى أي عنصر من عنصر المصفوفة لا بد من استخدام دليل يدل على هذا العنصر، علماً ان ترتيب عناصر المصفوفة يبدأ من الصفر، وبالتالي فإن:

ترتيب آخر عنصر من عناصر المصفوفة = عدد العناصر - 1

مثال : اذا كانت عناصر المصفوفة X بالشكل التالي [0 , 7, 4, 3, 6, 2] فإن :

x[0]=2

x[1]=6

x[2]=3

x[3]=4

x[5]=0

إدخال قيم عناصر المصفوفة:

يتم التعامل مع عناصر المصفوفة كأي مجموعة من المتغيرات المشتركة بالنوع، لكن كونها تقع ضمن مصفوفة فإن هذا الأمر يقدم مجموعة من الميزات كإمكانية العودة إليها باستمرار كما ذكرنا سابقا، وكذلك إمكانية إدخال أو قراءة قيمها بشكل أبسط وأسرع من إدخال قيمة كل متغير على حدا، الأمر الذي يصبح معقدا جدا عند التعامل مع أعداد ضخمة من المتغيرات، يمكننا استخدام الحلقات للتعامل مع عناصر المصفوفة مثل حلقة For،

مثال : اكتب برنامج لإدخال عناصر مصفوفة مؤلفة من 10 عناصر:

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

```
{
int i ;
int  myarr [10] ;
for ( i = 0 ; i < 10 ; i++)
   {
     Cout << " myarr[ "<< i << " ]= " ;
      Cin >> myarr[ i ];
     }

}
```

مثال : اكتب برنامج لإدخال عناصر مصفوفة مؤلفة من 10 عناصر، ثم طباعة العنصر الأكبر من عناصر المصفوفة.

```
#include <iostream.h>
main()
{
int i ;
int max;
int  arr [10] ;
for ( i = 0 ; i < 10 ; i++)
   {
      cin >> arr[ i ];
     }
max = arr[0];
```

```
for (  i = 0 ; i < 10 ; i++)
    {
      if (arr[ i ] > max)
         max = arr[ i ];
     }
cout <<"Maximum number of array is: " << max;
}
```

المصفوفات ثنائية البعد: تتكون من أكثر من سطر وأكثر من عمود ، ويتم التصريح عنها بالشكل التالي:

```
type  array_name[n][m];
```

حيث:

type : نوع عناصر المصفوفة (int , float ,.....).

array-name : اسم المصفوفة.

n : عدد أسطر المصفوفة.

m : عدد أعمدة المصفوفة.

مثال: `int  arr [ 5 ][ 3 ] ;`

هنا عرفنا مصفوفة اسمها arr مؤلفة من 5 أسطر و 3 أعمدة ، عناصر هذه المصفوفة أعداد صحيحة.

للوصول إلى أي عنصر من عنصر المصفوفة لا بد من استخدام دليل يدل على هذا العنصر، في المصفوفات الثنائية نحتاج إلى دليل للأسطر ودليل للأعمدة.

مثال : اكتب برنامج لإدخال عناصر مصفوفة مؤلفة من 3 أسطر و4 أعمدة:

```
#include <iostream.h>
main()
{
int i , j ;
int  x [3][4] ;
for ( i = 0 ; i < 3 ; i++)
for (  j = 0 ; j < 4  ; j++)
   {
          Cin >> x[ i ][ j ] ;
     }
 }
```