

المصفوفات Arrays

عند التعامل مع حجم كبير من البيانات كنا نستخدم الكثير من المتغيرات لتخزين ومعالجة تلك البيانات ، لأن كل متغير يخزن فيه قيمة واحدة فقط ، ولكن هذا يجعل البرنامج طويلاً ومعقداً في بعض الأحيان ولكن باستخدام المصفوفة يستطيع المبرمج استعمال متغيرات قليلة وذلك بتقسيم كل متغير إلى عدد من العناصر المتسلسلة

تعريف المصفوفة :-

هي عبارة عن منطقة في الذاكرة تتكون من عدد محدد Finite ومتجانس Homogenous من المواقع المتجاورة أ - التجانس ويعني أن هذا الجزء من الذاكرة يستخدم لتمثيل نوع واحد من البيانات Data Type ب - محدد Finite هو عبارة عن رقم صحيح يستخدم لتحديد عدد المواقع المطلوبة ويسمى هذا الرقم بالدليل

أنواع المصفوفات :-

يمكن تقسيم المصفوفات إلى نوعين هما :

1. المصفوفات ذات البعد الواحد One Dimensional Arrays

وهي عبارة عن صف أو عمود يحتوي على مجموعة من عناصر البيانات متحدة النوع والاسم

الصيغة العامة للإعلان عن المصفوفة ذات البعد الواحد:

data type array name [index] ;

حيث :-

- a. data type : نوع بيانات المصفوفة.
- b. array name : اسم المصفوفة ويراعى فيه شروط تسمية المتغيرات.
- c. index : دليل المصفوفة (عدد عناصرها) وهي عبارة عن قيمة صحيحة يمكن أن تكون محدده (ثابتة) أو مدخلة من قبل المستخدم (متغيرة) .

أمثلة :

1. int x [50];

إعلان عن مصفوفة حجمها 50 وبياناتها من النوع الصحيح

2. float y [20];

إعلان عن مصفوفة حجمها 20 وبياناتها من النوع الحقيقي

3. char name [15] ;

إعلان عن مصفوفة حجمها 15 وبياناتها من النوع الحرفي

4. `int x [3] = { 5,10,15 } ;`

إعلان عن مصفوفة حجمها 3 وبياناتها من النوع الصحيح مع إعطائها قيم ابتدائية

5. `char y [4] = { 'a','b','c','d' } ;`

إعلان عن مصفوفة حجمها 4 وبياناتها من النوع الحرفي مع إعطائها قيم ابتدائية

الوصول إلى عناصر المصفوفة (Array Elements Allocation)

للوصول إلى المواقع داخل المصفوفة يستخدم الدليل الذي تضاف له وظيفة أخرى إذ انه يعمل كمرجع للخلايا المكونة للمصفوفة حيث ترقم الخلايا بالصورة التالية :

المواقع في الذاكرة
عناصر المصفوفة

0	1	2	n-2	n-1
1	2	3	n-1	n

وبالتالي للوصول إلى أي موقع لإجراء عملية إدخال أو معالجة أو إخراج أو اتخاذ قرار نستخدم اسم المصفوفة زائداً دليل الموقع المراد الوصول إليه .

يبين الشكل التالي مصفوفة من الأعداد الصحيحة اسمها C. وهي تتضمن 10 عناصر:

C[0]	-5
C[1]	8
C[2]	32
C[3]	11
C[4]	0
C[5]	-14
C[6]	36
C[7]	-76
C[8]	1
C[9]	9

يمكن الرجوع إلى أي عنصر من العناصر السابقة بإعطاء اسم المصفوفة متبوعاً



برقم موضع العنصر داخل قوسين من الشكل []	
موقع العنصر الأول في المصفوفة هو دائماً صفر، لذلك يتم الرجوع إلى العنصر الأول في المصفوفة C على الشكل التالي C[0]، والعنصر الثاني C[1]، أما من أجل الرجوع إلى العنصر السابع نكتب C[6]، وبشكل عام نستطيع القول أننا نرجع إلى العنصر رقم i بأن نكتب C[i-1].	x
نسمي رقم الموضع الذي نضعه بين قوسين بالدليل (subscript)، ويجب أن يكون الدليل عبارة عن عدد صحيح أو أي تعبير يعطي قيمة صحيحة.	x

برامج متكاملة عن المصفوفة ذات البعد الواحد

1. عرف مصفوفة تستخدم لتمثيل 5 قيم صحيحة حيث أن القيمة رقم 1 هي 20 ورقم 3 هي 15 والقيمة رقم 2 تساوي القيمة رقم 1 زائداً القيمة رقم 3 والقيمة الرابعة هي $4=2-3$ والقيمة الخامسة تساوي القيمة الأكبر ما بين الثانية والرابعة أي $5=2>4$ أو $5=4>2$ ؟ ثم أطلع عناصرها .

```
#include <iostream.h>
int main( )
{
    int i,x[5];
    x[0]=20;
    x[2]=15;
    x[1]=x[0]+x[2];
    x[3]=x[1]-x[2];
    if(x[1]>x[3])
        x[4]=x[1];
    else
        x[4]=x[3];
    for(i=0;i<5;i++)
        cout<<"x["<<i+1<<"]="<<x[i]<<"\n";
    return 0;
}
```

2. اكتب برنامجاً بلغة C++ لقراءة عناصر مصفوفة ذات بعد واحد تتكون من 10 عناصر ثم حساب وطباعة

مجموع عناصر هذه المصفوفة.

```
#include <iostream.h>
int main()
{
    int a[10];
    int sum,i;
    sum=0;
    for(i=0;i<10;i++)
    {
        cin>>a[i];
        sum +=a[i];
    }
    cout<<"sum="<<sum<<"\n";
    return 0;
}
```

3. اكتب برنامج باستخدام المصفوفات لحساب متوسط 5 قيم مدخلة من النوع الحقيقي ؟

```
#include<iostream.h>
int main( )
{
    float x[5] , sum=0 , avg;
    int i;
    for(i=0;i<5;i++)
    {
        cout<<"enter element["<<i+1<<"] \n";
        cin>>x[i];
        sum+=x[i];
    }
    avg=sum/5;
    cout<<"the average="<<avg<<"\n";
    return 0;
}
```

4. اكتب برنامج لإدخال 5 قيم حقيقية ثم طباعتها بصورة مرتبة تصاعدياً؟

```
#include <iostream.h>
const int n=5;
int main( )
{
float x[n],temp;
int i,j;
for(i=0;i<n;i++)
{
cout<<"enter element["<<i+1<<"]\n";
cin>>x[i];
}
for(i=0;i<n-1;i++)
for(j=i+1;j<n;j++)
{
if(x[i]>x[j])
{
temp=x[i];
x[i]=x[j];
x[j]=temp;
}
}
cout<<"elements after sorting\n";
for(i=0;i<n;i++)
cout<<x[i]<<"\n";
return 0;
}
```

2. المصفوفات متعددة الأبعاد Multi Dimensional Arrays

المصفوفات متعددة الأبعاد هي التي تتكون من عدة أبعاد. وعادة ما يتم التعامل مع المصفوفات ذات البعدين لقلة التطبيقات التي تحتاج لمصفوفات ذات أكثر من بعدين.

المصفوفة ذات البعدين تتألف من مجموعة من الصفوف rows ومجموعة من الأعمدة columns وفيها يتم إعطاء المصفوفة دليلين $n * m$ ويساعدنا هذا النوع من المصفوفات في تمثيل المصفوفات الرياضية على الذاكرة بافتراض أن (n تمثل دليل الصفوف و m تمثل دليل الأعمدة).

الصيغة العامة للإعلان عن المصفوفة ذات البعدين:

data type array name [row size] [column size];
 حجم الأعمدة حجم الصفوف اسم المصفوفة نوع المصفوفة

<p>يتم التصريح عن المصفوفات ذات البعدين بنفس طريقة التصريح عن المصفوفات ذات البعد الواحد غير أننا نحدد عدد الصفوف والأعمدة في المصفوفات ذات البعدين</p>	X
---	---

أمثلة :

1. int y[4][3]

تصريح عن المصفوفة الصحيحة y كمصفوفة ذات بعدين وتتكون من 4 صفوف وثلاثة

2. y[2][3]=23;

إسناد القيمة 23 لعنصر الصف الثالث والعمود الرابع من المصفوفة y

3. int y[4][3]={ {5,0,-4},{-2,3,1},{4,7,6},{9,8,-1}};

إسناد قيم ابتدائية لعناصر المصفوفة y أثناء التصريح. حيث المصفوفة y عناصرها كما يلي:

$$y = \begin{bmatrix} 5 & 0 & -4 \\ -2 & 3 & 1 \\ 4 & 7 & 6 \\ 9 & 8 & -1 \end{bmatrix}$$

حيث يتم تجميع عناصر كل صف ضمن قوسين.

ومن أجل الوصول إلى أي عنصر من عناصر المصفوفة ذات البعدين يتم تحديد رقم الصف ورقم العمود الواقع فيه هذا العنصر.

والشكل التالي يوضح مصفوفة ذات بعدين:

	العمود 1	العمود 2	العمود 3	العمود 4
الصف 1	a[0][0]	a[0][1]	a[0][2]	a[0][3]
الصف 2	a[1][0]	a[1][1]	a[1][2]	a[1][3]
الصف 3	a[2][0]	a[2][1]	a[2][2]	a[2][3]

<p>يمكن تحديد أي من عناصر المصفوفة ذات البعدين باستخدام دليلين، دليل لرقم الصف ودليل آخر لرقم العمود فمثلاً لتحديد العنصر الموجود في الصف الثاني والعمود الثالث من المصفوفة x نكتب x[1][2] لاحظ هنا أن الصف الأول يأخذ الرقم 0 والعمود الأول يأخذ الرقم 0 أيضاً.</p>	✗
<p>إذا لم يتم إعطاء قيم كافية لعناصر المصفوفة فإنه يتم إسناد القيمة 0 إلى باقي العناصر التي لم يتم إسناد قيم لها. فمثلاً التصريح:</p> <p style="text-align: center;">int b[2][2]={ { 1 }, { 3,4 } }</p> <p>يعطي العنصر b[0][0] القيمة 1 والعنصر b[1][0] القيمة 3 والعنصر b[1][1] القيمة 4 وباقي العناصر القيمة 0.</p>	✗

برامج متكاملة عن المصفوفة ذات البعدين

1. برنامج يقوم بقراءة مصفوفة ذات بعدين مع طباعة البيانات المدخلة على هيئة المصفوفة الشائبة؟

```
#include<iostream>
const int row=3;
const int col=4;
int main( )
{
    int a[row][col],i,j;
    cout<<"enter the elements of array:\n";
    for(i=0;i<row;i++)
    {
        for(j=0;j<col;j++)
        {
            cout<<"a["<<i<<","<<j<<"]="";
            cin>>a[i][j];
        }
    }
    cout<<"the array looks like:\n";
    for(i=0;i<row;i++)
    {
```

```

for(j=0;j<col;j++)
cout<<a[i][j]<<"\t";
cout<<"\n";
}
return 0;
}

```

2. اكتب برنامج يقوم بجمع مصفوفتين صحيحتين من النوع 3×3 لأعداد يتم إدخالها من قبل المستخدم ثم طباعة المصفوفات على هيئة المصفوفة الشائبة؟

```

#include <iostream.h>
int main( )
{
int a[3][3],b[3][3],c[3][3],i,j;
cout<<"first array a[3][3]:\n";
for(i=0;i<3;i++)
{
for(j=0;j<3;j++)
cin>>a[i][j];
cout<<"\n";
}
cout<<"second array b[3][3]:\n";
for(i=0;i<3;i++)
{
for(j=0;j<3;j++)
cin>>b[i][j];
cout<<"\n";
}
cout<<"the first array a[3][3] look like :\n";
for(i=0;i<3;i++)
{
for(j=0;j<3;j++)
cout<<a[i][j]<<"\t";
cout<<"\n";
}
cout<<"the second array b[3][3] look like :\n";
for(i=0;i<3;i++)
{

```



```
for(j=0;j<3;j++)
cout<<b[i][j]<<"\t";
cout<<"\n";
}
cout<<"the sum of tow array c[3][3] look like:\n";
for(i=0;i<3;i++)
{
for(j=0;j<3;j++)
{
c[i][j]=a[i][j]+b[i][j];
cout<<c[i][j]<<"\t";
}
cout<<"\n";
return 0;
}
```

تمارين

1. اكتب برنامجاً بلغة C++ لحساب مجموع مصفوفتين a و b وتخزين المجموع في المصفوفة c ثم طباعة عناصر المصفوفة الناتجة c علماً بأن عدد عناصر كلا من المصفوفتين c و b هو 5.
2. اكتب برنامجاً لقراءة عناصر مصفوفة $X(4 \times 3)$ ثم عمل ما يلي:
 - a. طباعة المصفوفة بشكل مناسب.
 - b. طباعة عدد الأرقام الموجبة وعدد الأرقام السالبة وعدد الأرقام الصفرية في المصفوفة
 - c. وطباعة عدد الأرقام الزوجية والفردية.