كلية العلوم

القسم : **الرياضيات**

السنة : **الثانية**

المادة : لغات البرمجة٢

المحاضرة : الثانية/عملي/

{{ مكتبة A to Z }}

Facebook Group : مكتبة A to Z

2026

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

الجمهورية العربية السورية
جامعة طرطوس
كلية العلوم قسم الرياضيات
السنة الثانية
المادة:لغات برمجة٢_عملي

المحاضرة الثانية
المصفوفات ثنائية البعد

2025-2026

أولاً: مفهوم المصفوفة الثنائية Two-Dimensional Array:

هي عبارة عن هيكل بيانات يستخدم لتخزين مجموعة من العناصر من نفس النوع (int,float,char,...) على شكل جدول مكون من صفوف(rows) وأعمدة(columns)، يمكن تخيلها كجدول رياضي أو كجدول بيانات (مثل جدول علامات الطلاب).

مثلاً: لدينا المصفوفة التالية بحجم ثلاث صفوف وأربع أعمدة.

	j=0	j=1	j=2	j=3	j=4
i=0	a[0][0]	a[0][1]	a[0][2]	a[0][3]	a[0][4]
i=1	a[1][0]	a[1][1]	a[1][2]	a[1][3]	a[0][4]

حيث كل عنصر داخل المصفوفة يمكن الوصول إليه عن طريق دليلين:

الدليل الأول: يمثل رقم الصف(row index)

والدليل الثاني: يمثل رقم العمود(column index)

ملاحظة: تخزن المصفوفة الثنائية صفاً صفاً (Row major order) في الذاكرة ، أي أن كل العناصر موجودة في الذاكرة بشكل متتابع.

ثانياً: التصريح عن المصفوفات الثنائية Declaring Arrays:

الصيغة العامة للتصريح:

```
ArrayType arrayName [row num][column num];
```

حيث أن:

ArrayType: نوع عناصر المصفوفة (ثوابت صحيحة أو حقيقية أحادية أو مضاعفة الدقة أو محرفية أو سلسلة محارف).

arrayName: اسم المصفوفة.

row num: عدد الصفوف ضمن المصفوفة.

column num:عدد الأعمدة ضمن المصفوفة.

ولدينا امثلة على ذلك:

- مصفوفة محارف اسمها letters مكونة من ثلاث صفوف وثلاث أعمدة char letters[3][3];
- مصفوفة أعداد صحيحة اسمها grade مكونة من ثلاث صفوف وأربع أعمدة ; int grade[3][4]

ويمكن أن نصرح عنها بالشكل التالي أيضاً:

```
int m=3, n=4;
int grade[m][n];
```

ثالثاً: تهيئة عناصر المصفوفة الثنائية Initializing The Elements of Array:

إعطاء قيم أولية لعناصر المصفوفة أثناء تعريفها أو بعد تعريفها،أي قبل أن يبدأ البرنامج باستخدامها في العمليات الحسابية أو الطباعة، وتوجد عدة طرق لتهيئة المصفوفة الثنائية وهي:

- التهيئة عند التعريف مباشرة: وهي أبسط وأوضح الطرق، حيث نضع القيم داخل أقواس مزدوجة {} بحيث تمثل كل مجموعة صفاً كاملاً داخل المصفوفة.

الصيغة العامة:

```
ArrayType arrayName [row num][column num]={
{value1,value2,value3,.....}, //  الصف الأول
{value1,value2,value3,.....}, //  الصف الثاني
...};
```

مثال توضيحي:

```
int arr[2][3]={

{1,2,3},

{5,6,7}};
```

هنا تمثل القائمة {1,2,3} عناصر السطر الأول ، وتمثل القائمة {5,6,7} عناصر السطر الثاني وتصبح المصفوفة بالشكل التالي:

1	2	3
4	5	6

- التهيئة الكاملة في سطر واحد: في هذه الطريقة نضع جميع القيم في سطر واحد دون الفصل بين الصفوف بأقواس داخلية.

مثال:

```
int arr[2][3]={1,2,3,4,5,6};
```

في هذه الحالة يقوم المترجم بملء الصف الأول بالقيم الثلاثة الأولى، ثم ينتقل للصف الثاني ويملؤه بالقيم الثلاث التالية وتصبح المصفوفة كما في المثال السابق.

- التهيئة الجزئية: في هذه الطريقة نضع عدداً من القيم أقل من عدد العناصر الكلي، والمترجم تلقائياً يملأ الباقي بالأصفار.

مثال:

```
int arr[2][3]={1,2,3};
```

في هذه الحالة يأخذ الصف الأول القيم 1,2,3 ويملأ المترجم قيم الصف الثاني كلها بالأصفار.

- التهيئة التلقائية لجميع العناصر بالصفر: يمكننا تهيئة كل عناصر المصفوفة بالصفر باستخدام قوسين فارغين{} .

مثال:

```
int arr[3][4]={};
```

أو:

```
int arr[3][4]={0};
```

في كلتا الحالتين، جميع العناصر في المصفوفة تملأ بقيمة 0 .

رابعاً: إدخال عناصر المصفوفة الثنائية Input The Elements of Array:

لكي ندخل القيم من لوحة المفاتيح لجميع العناصر في المصفوفة، نحتاج المرور على كل صف وداخل كل صف على كل عمود، لذلك نستخدم حلقتي for متداخلتين حيث الحلقة الأولى نستخدمها للمرور على الصفوف واحداً تلو الآخر، والحلقة الثانية نستخدمها للمرور على الأعمدة داخل كل صف، وبهذه الطريقة نغطي كل عنصر في الجدول.

مثلاً: يتم إدخال عناصر المصفوفة grade[m][n] من لوحة المفاتيح وفق جزء الشيفرة الأتية:

```
for ( int i = 0; i < m; i++)
{ for (int j = 0; j < n; j++) {
 cin >> a[i][j]; //أداة إدخال قيم عناصر المصفوفة من لوحة المفاتيح
```

في كل مرة، نقرأ من المستخدم قيمة العنصر ونخزنها في المكان المناسب حسب موقعه في المصفوفة .

خامساً: طباعة عناصر المصفوفة الثنائية Displaying The Elements of Array:

بعد إدخال القيم، نحتاج إلى طباعة المصفوفة بنفس ترتيبها(صفاً صفاً) فنستخدم أيضاً حلقتي for المتداخلتين بنفس الأسلوب فمثلاً لإظهار قيم عناصر المصفوفة grade[m][n] نكتب جزء الشيفرة الآتية:

```
for ( int i = 0; i < m; i++)
```

{ for (int j = 0; j < n; j++) {

cout<<a[i][j];// أداة إظهار قيم عناصر المصفوفة

}}

مثال1: ليكن لدينا مصفوفة أعداد صحيحة M[ ][ ] مكونة من صفين و أربع أعمدة والمطلوب:

1: إدخال قيم عناصر المصفوفة من لوحة المفاتيح

2: إيجاد مجموع عناصر السطر الثاني والمتوسط الحسابي لهذه العناصر

3: إيجاد جداء عناصر العمود الأول

4: إظهار عناصر المصفوفة

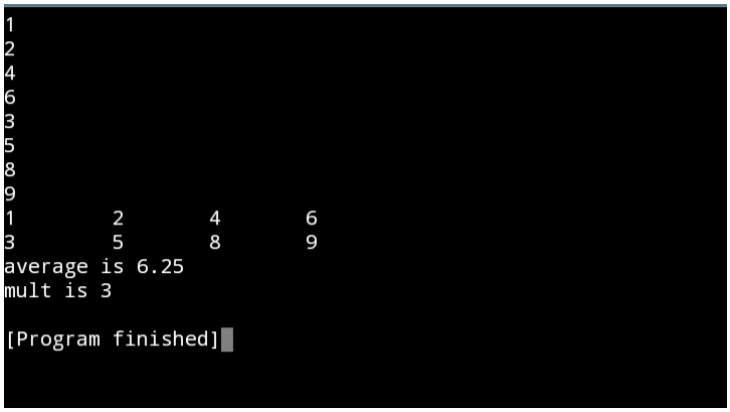
الحل:

```
1  #include<iostream.h>
2  int main()
3  {
4     float average;
5     int M[2][4];
6     int sum=0;
7     int mult=1;
8     for(int i=0;i<2;i++){
9        for(int j=0;j<4;j++){
10          cin>>M[i][j];
11          if(i==1)
12          sum+=M[i][j];
13          if(j==0)
14          mult*=M[i][j];
15       }}
16       average=(float)sum/4;
17 for(int i=0;i<2;i++){
18    for(int j=0;j<4;j++){
19       cout<<M[i][j]<<"\t ";}
20       cout<<"\n";
21    }
22    cout<<"average is "<< average<<"\n";
23    cout<<"mult is "<<mult<<"\n";
24 }
```

ملاحظة: إذا كان المطلوب حساب المتوسط الحسابي لجمعيع عناصر المصفوفة فإننا نقسم مجموع عناصر المصفوفة على عددها الكلي ، ويكون عدد العناصر= عدد الصفوف * عدد الأعمدة .

أما إذا كان المطلوب المتوسط الحسابي لعناصر صف واحد فقط فنقسم المجموع على عدد الأعمدة فقط كما هو في المثال أعلاه.

الخرج:



الأقطار في المصفوفات الثنائية والعناصر المحيطة بها:

الأقطار تعرف فقط في المصفوفات المربعة(التي يكون فيها عدد الصفوف يساوي عدد الأعمدة) لان الأقطار تمتد من زاوية إلى زاوية، وهذا لايمكن أن يحدث بشكل متناسق في المصفوفات المستطيلة.

a[0][0]	a[0][1]	a[0][2]
a[1][0]	a[1][1]	a[1][2]
a[2][0]	a[2][1]	a[2][2]

هنا لدينا مصفوفة مربعة عدد الصفوف فيها يساوي عدد الأعمدة ويساوي 3

ويكون لدينا الحالات التالية:

- القطر الرئيسي: هو الخط الي يمتد من الزاوية العلوية اليسرى إلى السفلية اليمنى، عناصره تحقق الشرط: i==j.
- العناصر فوق القطر الرئيسي: هي العناصر الموجودة أعلى يمين القطر الرئيسي ، تحقق الشرط: i < j.
- العناصر تحت القطر الرئيسي: هي العناصر الموجودة أسفل يسار القطر الرئيسي ، تحقق الشرط: i > j.
- القطر الثانوي: هو الخط الي يمتد من الزاوية العلوية اليمنى إلى السفلية اليسرى، عناصره تحقق الشرط : i==j < n-1 (حيث n هو حجم المصفوفة المربعة).

مثال 2 : ليكن لدينا مصفوفة مربعة من الأعداد الصحيحة بحجم n*n، الحد الأعظمي لعدد عناصرها 20 اكتب برنامجاً يقوم بما يلي:

1: إدخال قيم عناصر المصفوفة من لوحة المفاتيح

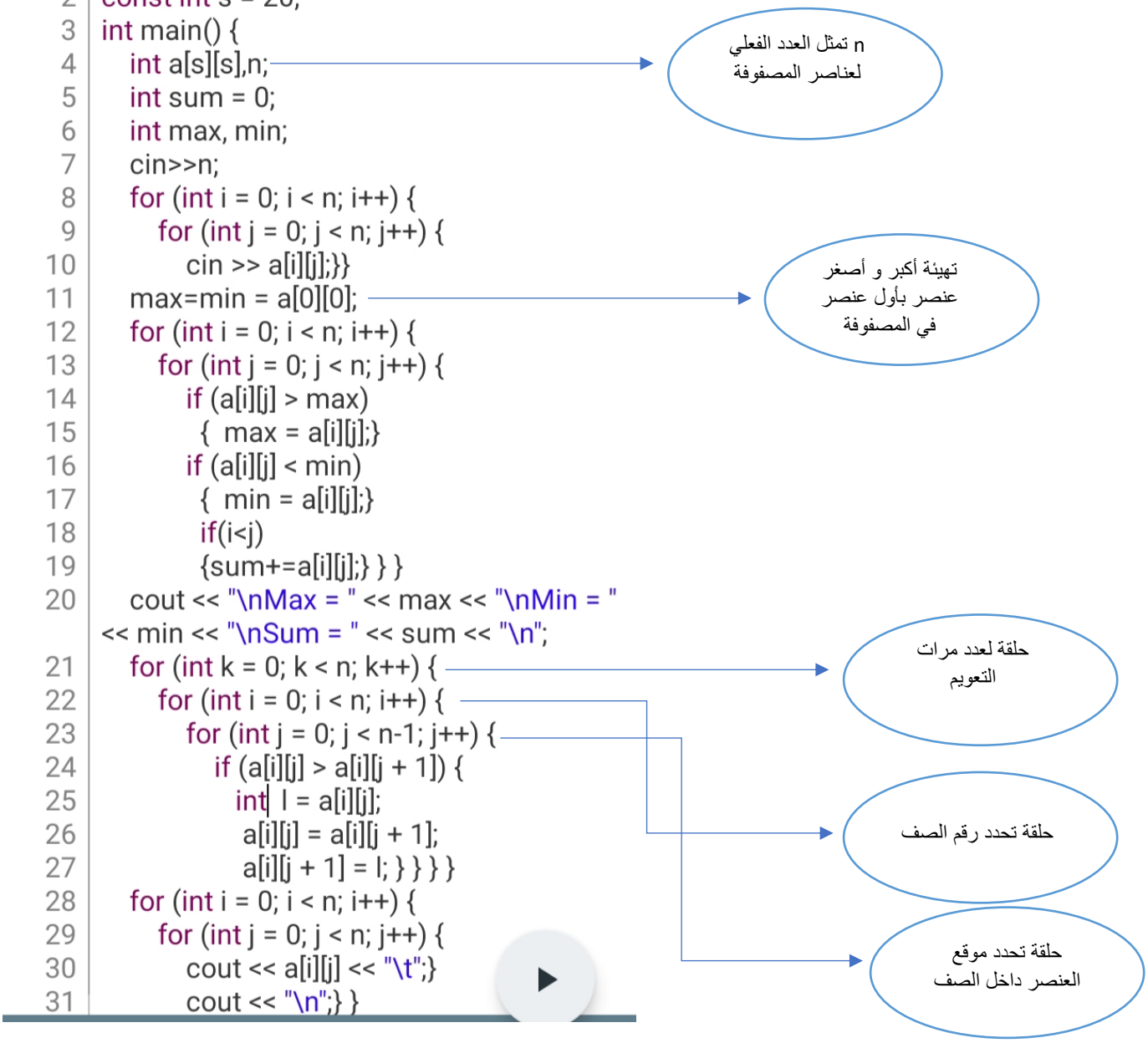
2: إيجاد القيمة الكبرى max والقيمة الصغرى min لعناصر المصفوفة

3: حساب مجموع عناصر المصفوفة الواقعة فوق القطر الرئيسي

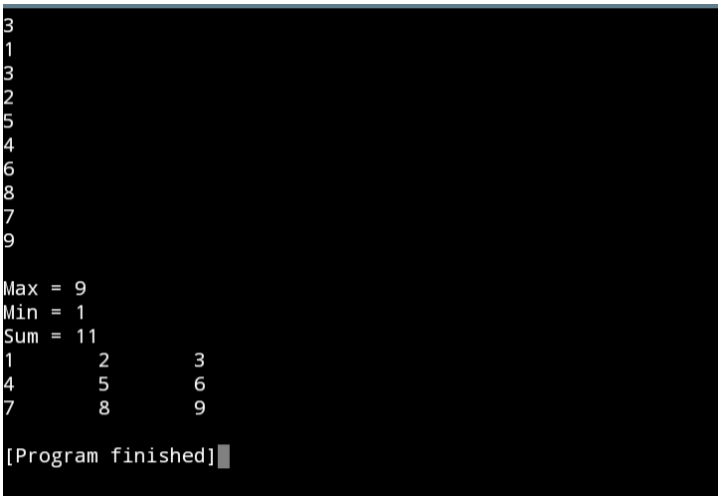
4: ترتيب عناصر المصفوفة تصاعدياً ثم عرضها

الحل:

```cpp
#include <iostream.h>
const int s = 20;
int main() {
   int a[s][s],n;
   int sum = 0;
   int max, min;
   cin>>n;
   for (int i = 0; i < n; i++) {
      for (int j = 0; j < n; j++) {
         cin >> a[i][j];}}
   max=min = a[0][0];
   for (int i = 0; i < n; i++) {
      for (int j = 0; j < n; j++) {
         if (a[i][j] > max)
          {  max = a[i][j];}
         if (a[i][j] < min)
          {  min = a[i][j];}
          if(i<j)
          {sum+=a[i][j];} } }
   cout << "\nMax = " << max << "\nMin = "
<< min << "\nSum = " << sum << "\n";
   for (int k = 0; k < n; k++) {
      for (int i = 0; i < n; i++) {
         for (int j = 0; j < n-1; j++) {
            if (a[i][j] > a[i][j + 1]) {
               int l = a[i][j];
               a[i][j] = a[i][j + 1];
               a[i][j + 1] = l; } } } }
   for (int i = 0; i < n; i++) {
      for (int j = 0; j < n; j++) {
         cout << a[i][j] << "\t";}
         cout << "\n";} }
```



الخرج:



وظيفة:

ليكن لدينا مصفوفة أعداد صحيحة [ ][ ]x مكونة من ثلاث صفوف وثلاث أعمدة ندخل قيمها من لوحة المفاتيح والمطلوب:

1: حساب مجموع عناصر المصفوفة ذات الترتيب الفردي

2: حساب مربع عناصر القطر الثانوي

3: طباعة عناصر المصفوفة

مكتبة A to Z