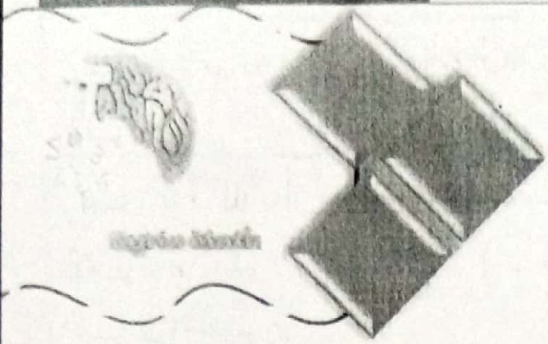

المحاضرة الخامسة

◄ دكتور المادة: سمير جعفر

◄ عنوان المحاضرة: حل تمارين من المحاضرة الرابعة

نظري ☑ عملي ☐

أهلاً بكم أصدقائي في المحاضرة الخامسة من مقررنا البرمجة المتقدمة، والتي سنتناول فيها حل برنامجين وظيفة من المحاضرة السابقة - قام الدكتور بحل البرنامج الثاني (عن المصفوفات). لنبدأ معاً...

برنامج المصفوفات: اكتب برنامجاً يقوم بإدخال مصفوفة محارف من البعد n x m حيث n,m أعداد طبيعية مدخلة أكبر من الواحد ثم ينجز مايلي:

1. طباعة الكلمة المشكلة من عناصر قطرها الرئيسي.
2. طباعة الكلمة المشكلة من عناصر قطرها الثانوي.
3. البحث عن المحرف المدخل ch فيما إذا كان موجوداً في المصفوفة أم لا، مع تحديد مكانه في حال وجوده. (سطره وعموده)
4. طباعة منقول المصفوفة على شكل مصفوفة.
5. طباعة المصفوفة المدخلة على شكل مصفوفة بعد حذف عناصر سطرها الثاني وعمودها ماقبل الأخير.

```
class Mymatrix{
pubric static void main(string args[]){
char c[][];
int n,m;
do{ n=stdin.readInt();} while(n<=1);
do{ m=stdin.readInt();} while(m<=1);
```

المطلوب مصفوفة محارف لذلك عرّفنا عن مصفوفة محارف وأسماها c ← المصفوفة من البعد nxm، لذلك عرفنا عن متغيرين صحيحين n,m

نريد البعدين n,m أكبر من الواحد لذلك استخدمنا الحلقة (do..while) لتصيد إدخال البعد طالما أنه أصغر أو يساوي الواحد.

c = new char [n][m]; ← حجز مكان في الذاكرة للمصفوفة

```
for(int i=0 ; i< c.length ; i++)
for( int j=0 ; j < c[i]. length ; j++)
  c[i][j] = stdin.readChar ( );
```

حلقتا for للأسطر والأعمدة لإدخال عناصر المصفوفة.

```
if ( c.length != c[0]. length ) System.out.print ( "error" );   ①
else { for(int i=0 ; i< c. length ; i++)                          } ②
        System.out. print ( c[i][i]);
```

```
for(int i=0 ; i< c.length ; i++)                                  } ③
System.out.print ( c[i][c.length -1-i ]); }
```

① حلقة شريطية - إذا كان عدد الأسطر لا يساوي عدد الأعمدة تعطي خطأ
② وإلا تقوم بطباعة عناصر القطر الرئيسي.
((استخدمنا التعليمة if هنا لأننا نريد المصفوفة مربعة))
③ طباعة عناصر القطر الثانوي.

```
char  ch = stdin.readChar ( );   ④
boolean  t = false ;   ⑤
int a,b;
for(int i=0 ; i< c. length ; i++)                                 } ⑥
for(int j=0 ; i< c[0]. length ; j++)
  if ( c[i][j] == ch) { t = true;   ⑦
                        a = i ;
                        b = j ; }
```

```
if (t == true) S.O.p("exist" + a + " " + b); }(8)
   else S.O.p("not exist");
```

(4) مررنا عن متغير من النوع المحرف وأدخلنا قيمته (وزيد الحث عنه في المصفوفة)
(5) عرفنا متغير بولياني (t) وأعطيناه القيمة (false) مبدئياً.
(6) حلقنا for المرور على جميع عناصر المصفوفة.
(7) في if وجود قيمة العنصر المحرف في المصفوفة نجعل قيمة المتحول البولياني (true) ونسند قيمة السطر إلى a والعمود إلى b.
(8) نطبع النتيجة مع دليل السطر والعمود، وإلا فإنه غير موجود ونطبع النتيجة.

```
char C1 = new char[C[0].length][C.length];
for(int i=0 ; i<C.length ; i++)
for(int j=0 ; j<C[0].length ; j++)
   C1[j][i] = C[i][j];
```
} (9)

```
for(int i=0 ; i<C1.length ; i++){
for(int j=0 ; j<C1[0].length ; j++)
   S.O.p(C1[i][j] + " ");
   S.O.pln(); }
```
} (10)

سنقول مصفوفة ينتج من قلب أسطرها أعمدة وأعمدتها أسطر.
(9) حلقنا for للمرور على جميع العناصر ومن ثم قلب الأسطر أعمدة والأعمدة أسطر.
(10) طباعة المنقول بشكل مصفوفة: يطبع عناصر كل سطر ثم ينزل سطراً في كل دورة.

```
for (int i=0 ; i<c.length ; i++){
 for (int j=0 ; j<c[0].length ; j++)
 if (i!=1 || j!= c[0].length - 2)
   S.O.p (c[i][j] + " ");
 else  S.O.p (" " + " ");
   S.O.pln ( ); }
```

المرور على جميع العناصر

وإذا وصلنا إلى دليل السطر الثاني أو دليل العمود قبل الأخير نقوم بطباعة فراغ.

} } (قوس نهاية البرنامج) (قوس نهاية main)

ملاحظات:

- يمكن كتابة `if(t)` بدلاً من `if(t==true)` و `if(!t)` بدلاً من `if(t==false)`
- مع كتابة S.O.p اختصاراً للتعليمة System.out.println
  ((لكن عند كتابة البرنامج على الحاسوب لا يمكن اختصارها))
- C.length تدل على عدد أسطر المصفوفة.
- C[i].length ((أو C[0].length)) تدل على عدد عناصر المتجهة المكونة من عناصر السطر (i) ((أو السطر (0))) ، وبالتالي تدل على عدد أعمدة المصفوفة.
- في الخطوة ① : لا يمكننا كتابة C[i].length بدلاً من C[0].length في مثل هذه الحالة ، لأننا كنا قد عرّفنا المتغير (i) كعداد للحلقة for (أي أنه غير معرّف خارج الحلقة) ، لذلك بعد خروجنا من الحلقة لا يتعرّف الـ compiler على (i) في حال استخدامه.

برنامج المتجهات: اكتب برنامجاً بلغة Java يقوم بإدخال متجهة أعداد حقيقية من البعد n حيث n عدد طبيعي مدخل أكبر من الواحد، ثم ينجز مايلي:

1- حساب وطباعة مجموع عناصرها.
2- حساب المتوسط الحسابي لعناصرها.
3- حساب جداء العناصر ذات الأدلة الفردية وطباعته.
4- حساب وطباعة مجموع مربعات عناصرها الزوجية.
5- البحث عن العدد الحقيقي المدخل x فيما إذا كان موجوداً في المتجهة أم لا.
6- طباعة عناصر المتجهة بشكل عكسي.

```
class Sondos{
public static void main(string args[]){
double A[];
int n;
do{ n=stdin.readInt(); } while (n<=1);
A=new double[n];
for(int i=0; i<A.length; i++)
   A[i]=stdin.readDouble();

double sum=0;
for(int i=0; i<A.length; i++)
sum+=A[i];
System.out.print(sum);
System.out.print(sum/A.length);
```

sum: مجموع العناصر
sum/A.length: المتوسط الحسابي

جداء العناصر ذات الدلالة الفردية.

```
double mult = 1;
for (int i=0 ; i < A.length ; i++)
{if (i % 2 != 0) { mult = mult * A[i];} }
S.O.p(mult);
```

مجموع مربعات العناصر الزوجية.

```
double sum2 =0 ;
for (int i=0 ; i< A. length ; i++)
 if (A[i] % 2 == 0){ sum2 = sum2 + (A[i] * A[i]);}
S.O.p(sum2);
```

البحث عن عنصر:

```
double x = stdin.readDouble(); boolean b = false;
for(int i = 0 ; i< A.length ; i++)
   if (A[i] == x) {b= true;}
if (b==true) S.O.p (x + " is in the array");
   else S.O.p (x+ " is not in the array");
```

طباعة عكسية

```
for (int i=0 ; i< A. length ; i++)
 S.O.p ( A[ A. length -1 -i] + " ");
```

طباعة عكسية بطريقة ثانية

```
for(int i = A.length -1 ; i >=0 ; i--)
 S.O.p (A[i]+ " ");
}
}
```

انتهت المحاضرة.