تمرين: اكتب صنفاً بلغة الجافا يقوم بما يليه:

1) البحث عن عنصر مدخل x فيما إذا كان موجود في متجهة أعداد صحيحة من البعد n حيث n عدد طبيعي مدخل.

2) ترتيب عناصر المتجهة ترتيباً تنازلياً.

3) طباعة أو حساب مجموع عناصر المتجهة ذات الدلالة الفردية.

4) طباعة عناصر المتجهة.

ثم اكتب برنامج باستخدام الصنف السابق يقوم بما يليه:

1) إدخال متجهة أعداد صحيحة من البعد m ∈ N*.

2) طباعة المتجهة مرتبة تصاعدياً وكذلك تنازلياً.

3) البحث عن عنصر y المدخل فيما إذا كان موجود أم لا؟

```
class MyArray{
    int A[];        int n;
    MyArray(int n){ This.n=n;
            A=new int[n]; }
    Void input(){
    for(int i=0 ; i<A.length ; i++)
        A[i]=Stdin.readInt();
    }
1)  boolean Search(int x){
    for(int i=0 ; i<A.length ; i++)
        if(A[i]==x)   return true;
            return flase;
        }
2)  int [] Sort(){
    for(int i=0 ; i<A.length-1 ; i++)
    for(int j=i+1 ; j<A.length ; j++)
        if(A[i]<A[j]){
```

باني (MyArray)

تقرأ المتجهة ولا تعيد شيء.

داخل الصنف كل الدوال تُرى الأعضاء.

لو كتبنا else return flase; خطأ لأننا نكون بيننا القراءة في أول عنصر

x=5 إذا كان A: | 1 | 5 | 11 | --- |

i=0 , A[0]=1 ≠5 ⇒ flase

/ /

```
int tmp = A[i];
    A[i] = A[j];
    A[j] = tmp;
    }
    return A;
```

3)
```
int sum(){
int s=0;
for (int i=1 ; i<A.length; i= i+2)
  S = S+ A[i];
  return S;
  }
```

4)
```
Void print(){
for (int i=0 ; i<A.length ; i++)
  S.O.p (A[i]+" ");
  }
}//end of class
```

أو (4//)
```
void print (int M[]);
for (int i=0 ; i< m.length; i++).
      S.O.p (M[i]+" "); }.
```

كتابة البرنامج

```
class Use Myarray{
public static Void main(String[]args){
  int m;// m موجود أن يغير n الـ
do { m=Stdin.readInt(); } While (m<=1);
Myarray ma ;
ma = new MyArray (m);
ma.input ();
ma.print (ma.Sort());
```

قراءة
```
for (int i=0; i<ma.A, i++)
    ma.A[i]=Stdin.readInt();
```

```
Myarray ma;                    او بطريقة ادخال ثانية وهي الافضل
int B[] = ma.Sort();
  ma.print (B);
for ( int i= B.length-1 ; i>=0 ; i--);
  S.O.p ( B[i]+ "  ");

3)   int y = Stdin.readInt ( );
     boolean t= ma.Search(y);
     if (t) S.O.p ( y + "is found");
     else S.O.p ( y + "is not found");
     }// main
   }// class.
```

تمرين :

اكتب صنفا بلغة الجافا يقوم بما يلي .

1) ايجاد منقول مصفوفة محارف من البعد mxn

2) حصه ارجاع الكلمة المشكلة من عناصر قطرها الرئيسي .

3) " " " " " " الثانوي

4) البحث عن عنصر ما فيما اذا كان موجود في هذه المصفوفة أم لا .

5) ارجاع الكلمة المشكلة من عناصر سطر معين .

6) " معكوس كلمة مشكلة من عناصر عمود معين ،

7) طباعة مصفوفة على شكل مصفوفة .

باستخدام الصنف السابق اكتب برنامجا يقرأ بادخال مصفوفة محارف من البعد CXD حيث C,DEN9

1) طباعة منقول هذه المصفوفة على شكل مصفوفة .

2) " الجملة المكونة من عناصر قطرها الرئيسي وقطرها الثانوي .

3) البحث عن حرف مدخل لا فيما اذا كان موجود أو لا .

4) طباعة الجملة المكونة من كلمات أسطرها .

5) " " " معكوس كلمات أعمدتها .

```
public static void main( --- ){
```
public ← عامة

إن الدالة main هي نقطة بداية التنفيذ وتكتب ضمن الصف ولكي نستخدم أي مكون من الصف ننشئ ، والدالة ثم نستدعيها.

مثال:
```
my main m = new mymain();
m.main();
```

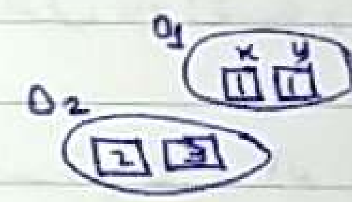
اما static : يمكن ان نستدعي الدالة من خلال اسم الصف فوراً ولا حاجة إلى إنشاء object لها.

```
my main . main();   // هنا يمكن كتابة هذا الكلام فقط
```
(my main: اسم الصف ، main: اسم الدالة)

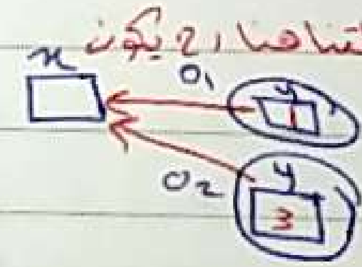
لأن دالة main مطورة على static

آخر مثال:
```
class XX{
  static int x = 0;
  int y;
  XX(int y){ this.y=y ; x=x+1 ;}
}
XX O1 = new XX(1);   ← ننشئ غرضين من الصف
XX O2 = new XX(3);
S.O.p (XX.x);   رح يطبع ②
```

بالشكل العادي



في حالتنا هنا رح يكون



يعني x مرتبطة بالصف وليست مرتبطة بالغرض فقط.

مثال: اكتب برنامج بلغة جافا يقوم بإدخال عدد طبيعي n وطباعة عدد طبيعي n! باستخدام دالة لحساب العاملي.

```
class fact{
  static long fact(int n){
    if (n==0) return 1;
```

```
else return (n * fact (n-1));
}

public static void main (String [] args) {
    int n;
    do { n = Stdin.readInt(); } while (n < 0);
    System.out.print (Fact.fact(n));
    }
}
```

استدعيناها باسم الصف هنا لأننا وضعنا static ببداية الدالة

او بالشكل العادي يلي فهمناه :

```
Fact f = new fact ( );
S.o.p (f.fact(n));
```

تعريف static : هي كلمة مفتاحية بالجافا ، تستخدم في موقعين إما أمام معطى أعضاء أو أمام الدوال.

إذا أتت أمام معطى بياني تجعل هذا المعطى عام للصف ويمكن الوصول لهذا المعطى البياني من خلال اسم الصف فوراً.

أما إذا وضعت أمام الطريقة (الدالة) فمعنى هذه الدالة لا تتعلق بأجزاء من الصف ويمكننا استدعاءها من خلال اسم الصف مباشرة.

مثال :

```
class A{
int x; int y;
A (int x, int y){
this.x = x ; this.y = y ; }
int sum() { return (x+y) }
} // class.
```

هنا لا يمكن وضع static لأنها تتعلق بالأجزاء من "مباشرة"

ملاحظة: اي صنف لا يحوي معطيات أعضاء يمكننا جعل كل دوال static مثل الصنف ((Stdin)).

```
A a = new A(0,0);
S.O.p(a.Sum());        // يطبع 0

A b = new A(3,5);
S.O.p(b.Sum());        // يطبع 8.
```

public : يمكن أن تأتي هذه الكلمة أمام :

1) اسم الصنف : يكون الصنف عام و يمكن رؤويته ويجب أن يخزن في ملف بنفس الاسم ((يمكن رؤويته لا تعني أنه يمكننا الوصول إلى جابداخله))
2) عضو بياني : فيكون هذا العضو عام اي يمكن رؤويته و الوصول اليه من خارج الصنف
3) أمام دالة : اي يمكن استدعاء هذه الدالة من خارج الصنف.

```
class A {
public int x;
private int y;
public A(int x, int y) {
        ⋮
}
int Sum() { return (x+y); }
}
```

```
class B {
        ⋮
A a = new A(3,2);   ✓
a.x = 5;            ✓
a.Sum();            ✗ لا يمكن لأنها دالة عامة
a.y = 1;            ✗ لا يمكن لأنها خاصة
}
```

بالصنف A.

ملاحظة: إن محدد الوصول الافتراضي بلغة الجافا هو ((package))

private تعني أنه خاص بالصنف و يمكن أن تأتي أمام اسم الصنف او أمام عضو بياني او أمام الطريقة.

/ /

final تعني النهائي ويمكن أن تأتي أيضاً

1) أمام الصف: أعني أن هذا الصف بشكله النهائي « يعني ما فينا نعدل عليه شي »

وتعني أيضاً أنه يمنع الوراثة من هذا الصف.

2) أمام عضو: تعني أن هذه القيمة لهذا المتحول هي نهائية.

finale int x = 5 ; « يعني طالما ال x أخذت قيمة (5) ما عاد فينا نغيرها نهائياً »

3) أمام دالة: تعني أن هذه الدالة بشكلها النهائي وممنوع التعديل عليها « يمنع تطبيق مبدأ تحميل الزائد عليها ».

```
class A{
finale int x;
 int y;
 }
A a=new A();
 a.y = 2;
 a.x = 5;
A b=new A();
 b.x = 3 ✓
}
```

✗ a.x=6 لأن أعطينا الx قيمة 5.

~ انتهت المحاضرة ~

ALADIB.net

منتدى الحضارة

مفهوم الوراثة في لغة الجافا

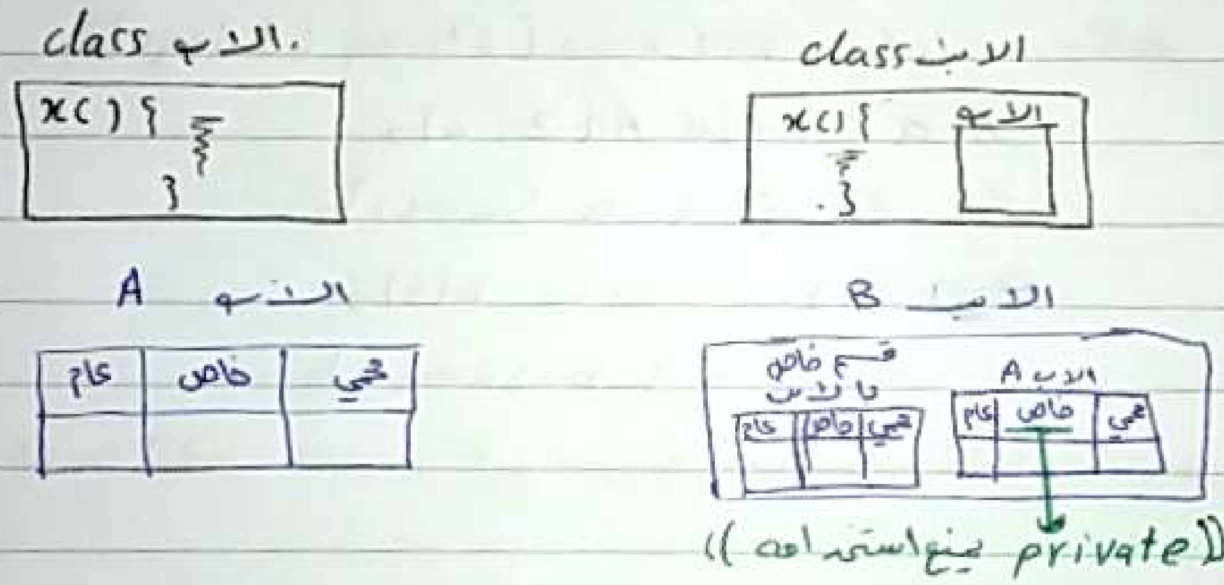
- البرمجة غرضية التوجه تعتمد على المبادئ (الأسس) الثلاثة:
  1) تجريد (صنوف - أغراض).
  2) الوراثة.
  3) تعددية الأشكال.

- الوراثة: صف يرث من صف آخر.

الابن	الأب
المشتق	القاعدة
الوارث	المورث

- تتم عملية الوراثة عند كتابة الصف الابن من خلال الكلمة المحجوزة extands

```
class الابن extands الأب {
}.
```



مثال:

```
class A{          صف الأب
 public int x;
 private int y;
 A() {    }  ⟺  A() { x=0, x=y;}   ← إذا لم نكتب تكون افتراضية
 A(int x, int y) { this.x = x; this.y = y; }
 public int sum(){
      return (x+y);
      }
 }
```

باني خالي افتراضي

صف الابن

```
class B extands A{
  int z;
  B (int z){
  Supe (3,3);
  this.z = z;
  }
  int sum(){
  return (Super.Sum()+A); }
```

لو بدنا نبعث أي رقم

```
B(int x, int y, int z){
super (x,y);
this.z = z;
}
```

Super تعني اسم الصف الاب للصف الحالي (A)

ويجب أن تكون أول تعليمة في باني الصف الابن ← هو استدعاء لباني الصف الاب.

ولو كتبنا return(x+y+z); صونها ما بيحسن يوصل لـ y لأنه خاص بالصف الاب A.

```
Class Main{
public static void main (String [] args){
A a = new A(3,2);
S.O.p (a.Sum());
B b = new b(5);
S.O.p (b.Sum());
```

// أي غرض من الصف الابن لازم يبتدي باني من الصف الاب //