

Exercice 01 : 06 pts (4 x 1,5)

- 1- En respectant la sémantique d'exécution, donner pour chaque morceau d'algorithme qui suit le code Java correspondant.

<p>Fonction isEgax (x , y : entier) : booléen</p> <pre> Début isEgax ← vrai Si (x ≠ y) alors isEgax ← faux finSi Fin </pre>	<pre> boolean isEgax(int x, int y) { if(x!=y) { return false; } return true; } </pre> <p>1,5pt</p>
<p>Procédure produire ()</p> <pre> var y : entier Début y ← 5 tant que (y > 0) faire écrire ("la valeur de y est :", y) y ← y -1 finTq Fin </pre>	<pre> void produire (){ int y=5 ; while (y>0) { System.out.println("la valeur de y est :"+y) ; y-=1 ; } } </pre> <p>1,5pt</p>

- 2- Que donne l'exécution des instructions suivantes

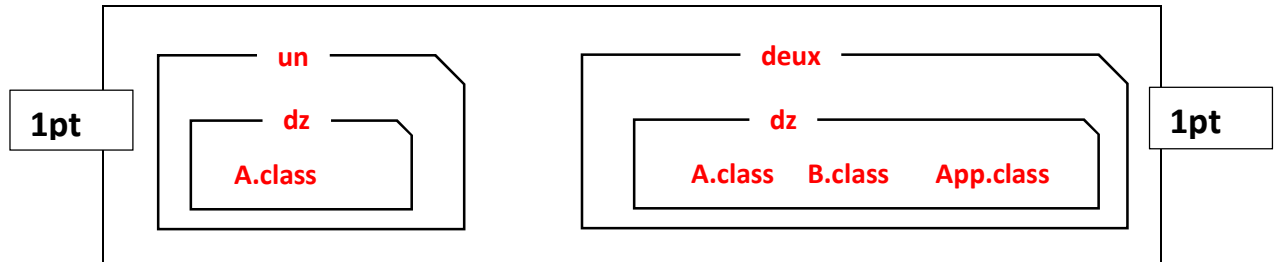
<pre> int a = 3; switch(a){ case 3 : System.out.println("3"); case 4 : System.out.println("4");break; case 5 : System.out.println("5"); default: System.out.println("0"); } </pre>	<p>3</p> <p>0,5pt</p> <p>4</p> <p>1pt</p>
<pre> int x = 5 ; while (x>0) { x-- ; if(x==4) { System.out.println(x); continue; } if (x==3) { System.out.println("sortir"); break; } System.out.println("quitter"); } </pre>	<p>4</p> <p>sortir</p> <p>0,25pt</p> <p>1,25pt</p>

Exercice 02 : 07 pts (2-3-2)

Considérons les classes suivantes

<pre>package un.dz; public class A { protected int y; int z; public int t; }</pre>	<pre>package deux.dz; import un.dz.A; public class B extends A { protected int b; int c; }</pre>	<pre>package deux.dz; public class C extends App { protected int p; int q; }</pre>	<pre>package deux.dz; public class App { protected int f; int m; }</pre>
--	--	--	--

1- Donner où seront créés les fichiers **A.class** , **B.class**, **C.class** et **App.class** Après la compilation



2- La classe **App** contient la méthode **main** suivante :

<pre>package deux.dz; public class App { protected int f; int m public static void main(String[] args){</pre>	
A objA = new A();	<input type="checkbox"/>
B objB = new B();	<input type="checkbox"/>
C objC = new C();	<input type="checkbox"/>
objB.b = 5;	<input type="checkbox"/>
objA.z = 5;	<input type="checkbox"/>
objB.y = 5;	<input type="checkbox"/>
objB.z = 5;	<input type="checkbox"/>
objA.y = 5;	<input type="checkbox"/>
objB.c = 5;	<input type="checkbox"/>
objC.p = 5;	<input type="checkbox"/>
objC.q = 5;	<input type="checkbox"/>
objA.z = 5;	<input type="checkbox"/>
objC.m = 5;	<input type="checkbox"/>
objC.f = 5;	<input type="checkbox"/>
objA.y = 5;	<input type="checkbox"/>
objA.t = 5;	<input type="checkbox"/>
<pre>}</pre>	

2 réponses correctes = 0.75 pt

3- Considérons les classes suivantes

<pre>public class B { int x = 4; public String toString(){ return x+""; } }</pre>	<pre>public class A { int x; static void incrementer(int x) { x++ ; } static void incrementer(B b){ b.x++ ; } }</pre>
<pre>public class Application { public static void main(String[]){ B b = new B(); int a = 5; A.incrementer(a); A.incrementer(b); System.out.println(a); System.out.println(b); }} </pre>	<p><u>Mentionner les résultats ici</u></p> <div> <div>5</div> <div>1pt</div> </div> <div> <div>5</div> <div>1pt</div> </div>

Exercice 03

<pre>public interface Geometrie { double ANGLE = Math.PI/4; double surface(); double perimetre(); }</pre>	<div> <div>0.5pt</div> <div>0.25pt</div> <div>0.25pt</div> </div>	<div> Bonus : Constante en majuscule + 0.25pt </div>
---	---	---

```
public class Cercle extends Point implements Geometrie{
```

```
    private double rayon;
```

Bonus + 0.25pt

```
    public Cercle(Point p, double rayon) {  
        super(p.x, p.y);  
        this.rayon = rayon;  
    }
```

0.5pt

```
@Override
```

```
    public double surface() {  
  
        return Math.PI*rayon*rayon;  
    }
```

0.25pt

```
@Override
```

```
    public double perimetre() {  
  
        return 2*Math.PI*rayon;  
    }
```

0.25pt

```
    public double getRayon() {  
  
        return rayon;  
    }
```

```
    public void setRayon(double rayon) {  
  
        this.rayon = rayon;  
    }
```

Bonus + 0.25pt

```
    public String toString () {  
  
        return "Cercle de centre"+getP()  
            + "est de rayon :"+rayon;  
    }
```

Bonus + 0.5pt

```
public class Huit implements Geometrie{
    private Cercle haut;
    private Cercle bas;
    private Point p;
```

0.25pt

Réponses correcte même si vous ne précisez pas **private**

```
    public Huit(Point p) {
        this.p = p;
        this.haut = new Cercle(new Point(p.x,p.y+1),1);
        this.bas = new Cercle(new Point(p.x,p.y-1),1);
    }
```

0.5pt

```
@Override
    public double surface() {

        return haut.surface()+bas.surface();
    }
```

0.5pt

```
@Override
    public double perimetre() {

        return haut.perimetre()+bas.perimetre();
    }
```

0.5pt

```
    public Point getP() {

        return p;
    }
```

Bonus 0.25pt

```
    public void setP(Point rayon) {

        this.p = p;
    }
```

```
    public Cercle getHaut() {

        return haut;
    }
```

0.25pt

```
    public void setHaut(Cercle haut) {

        this.haut = haut;
    }
```

```
    public Cercle getBas() {

        return bas;
    }
```

0.25pt

```
    public void setBas(Cercle bas) {

        this.bas = bas;
    }
```

```
}
```

3 - Qu'affiche le programme suivant ?

Très très simple

```
public static void main(String[] args) {  
    Point p1 = new Point(1,1); Point p2 = p1;    // ici p2 pointe vers le  
    même objet que p1 → p2=p1  
    Point p3 = p2; // ici p3 pointe vers le même objet que p2 → p3=p2  
    Donc p1=p2=p3 → p1 ,p2 et p3 pointent vers le même objet  
    p2.y = 4; // → p1.y = 4 et p3.y =4  
    p3.x =2; // → p1.x = 2 et p2.y =2  
    t = charger (p1,p2,p3); → la méthodes elle crée des points  
    avec (pi.x + pi.y , 0) → pour chaque point  
    → (4+2 , 0) → (6,0)  
    for (Point r : t){  
        System.out.println(r);  
    }  
}
```

(6.0,0.0)	}	1pt
(6.0,0.0)		1pt
(6.0,0.0)		0.75pt