



Haute école d'ingénierie et d'architecture Fribourg  
Hochschule für Technik und Architektur Freiburg

## PROGRAMMATION ET TP

T1A

RÉSEAU ET SÉCURITÉ

---

# S13 Classes et Objet

---

ROTEN MARC

2017/2018

# Table des matières

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Exercice 1</b>	<b>2</b>
<b>3</b>	<b>Exercice 2</b>	<b>3</b>
<b>4</b>	<b>Exercice 3</b>	<b>4</b>
<b>5</b>	<b>Exercice 4</b>	<b>5</b>
<b>6</b>	<b>Exercice 5</b>	<b>6</b>
<b>7</b>	<b>Conclusion</b>	<b>7</b>

# 1 Introduction

cette série sur les classe et les objets a pour objectif de nous aider a mieux comprendre ce qu'est la programmation orientée objet et la notion de classes. je me met gentiment au LaTeX dans le but d'uniformiser mes différents travaux.

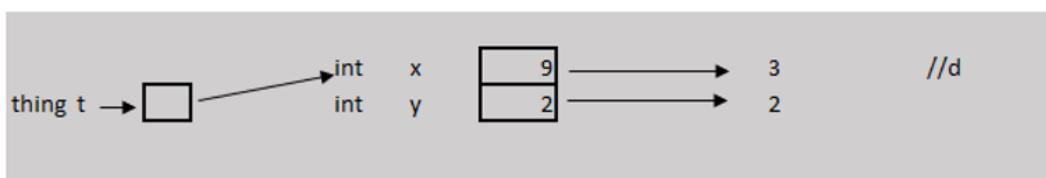
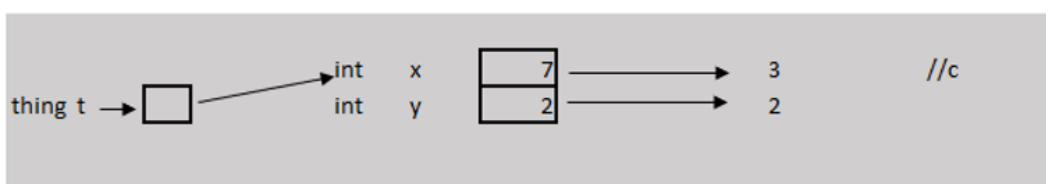
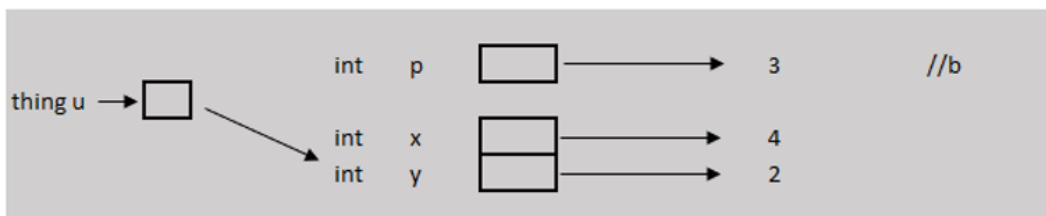
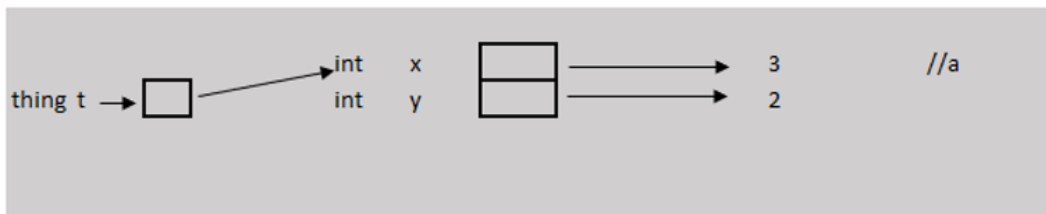
## 2 Exercice 1

- 1 Représenter l'état de la mémoire à chaque passage à un (\*) du programme suivant.

```
public class Thing {  
    public int x=0;  
    public int y;  
}
```

*Soigner la notation !*

```
public class Pg1301 {  
    public static void main(String[] args) {  
        Thing t = null;  
        t = new Thing();  
        t.x = 3; t.y = 2;           // (*) a  
        t.x = 4 + f(t, t.y);  
        t.x = t.x + t.y;           // (*) c  
        t = new Thing();  
        t = null;                 // (*) d  
    }  
    public static int f(Thing u, int p) {  
        u.x = 2*p;  
        p++;                       // (*) b  
        return p;  
    }  
}
```



### 3 Exercice 2

- 2 Déclarer les attributs ainsi que l'entête des constructeurs et des méthodes d'une classe B, de telle manière qu'elle soit compatible (pour la compilation) avec l'extrait de code suivant :

```
B b = new B(10);
b.c[b.d][b.e()] = -6.2;
b.f = b.g(b.e(b).substring(0,2)).charAt(3);
b.h = new B();
b.i(3)[2] = b.d;
```

---

```
// Hello.java
package s13;

public class Ex2 {
    public static void main(String[] args) {
        B b = new B(10);
        b.c[b.d][b.e()] = -6.2;
        b.f = b.g(b.e(b).substring(0,2)).charAt(3);
        b.h = new B();
        b.i(3)[2] = b.d;
    }
}
```

---

```
package s13;

public class B {

    public B h;
    public double[][] c;
    public int d;
    public Object f;
    public B(int i) {
    }
    public B() {
    }
    public int e() {
        return 0;
    }
    public String e(B b) {
        return null;
    }
    public String g(String substring) {
        return null;
    }
    public int[] i(int i) {
        return null;
    }
}
```

---

## 4 Exercice 3

- 3 La classe Account simule un compte en banque, où l'on peut déposer de l'argent (deposit), en retirer (withdraw), ou tout transférer (moveTo) vers un autre compte.

Dans certaines situations, cette classe Account ne se comporte pas correctement.

Identifier, expliquer et corriger l'erreur.

```
public class TestAccount {
    public static void main(String[] args) {
        Account a = new Account(10);
        Account b = new Account(20);
        Account c = new Account(5);

        b.deposit(20);
        a.withdraw(2);
        a.moveTo(b);
        c.moveTo(c);

        System.out.println(a.consult()+" "+
                           b.consult()+" "+
                           c.consult());
    }
}
```

```
public class Account {
    public int dollars;
    public Account(int money) {
        dollars = money;
    }
    public Account() {
        this(0);
    }
    public int consult() {
        return dollars;
    }
    public void deposit(int x) {
        dollars += x;
    }
    public void withdraw(int x) {
        dollars -= x;
    }
    public void moveTo(Account dest) {
        dest.dollars += dollars;
        dollars = 0;
    }
}
```

Avec la fonction MoveTo, il faut vérifier que le compte de destination n'est pas le même que le compte émetteur sinon, le compte va simplement se vider vu que dollars = 0, est placé après dest.dollars+=dollars;

---

```
package s13;
```

```
public class Account {
    public int dollars;
    public Account(int money) {
        dollars = money;
    }
    public Account() {
        this(0);
    }
    public int consult() {
        return dollars;
    }
    public void deposit(int x) {
        dollars += x;
    }
    public void withdraw(int x) {
        dollars -= x;
    }
    public void moveTo(Account dest) {
        if(dest!=this) {
            dest.dollars += dollars;
            dollars = 0;
        }else return;
    }
}
```

---

## 5 Exercice 4

- 4 Pour représenter les informations suivantes, déclarer une variable en choisissant un type approprié (primitif, prédéfini, classe, tableau, ...) :

- a) une **température**
- b) une **liste de prix** (nom de l'article, et le montant en francs et centimes)
- c) un **triangle** dans l'espace
- d) une **fiche personnelle** (nom, prénom, date de naissance, photographie)
- e) un **ordre d'arrivée** d'une course de chevaux
- f) une **image** noir/blanc

Si nécessaire, déclarer de nouvelles classes avec les bons attributs (ni constructeurs ni méthodes).

Quels sont les critères qui doivent aider à faire le bon choix ? Pourquoi, par exemple, ne pas représenter une date par une chaîne de caractères (type String) dans une application bancaire ?

---

```
//a
public double temp;

//b
public Ex4DeclVar (String Name, double Price) {
}

//c
public class Triangle {
    public Triangle() {
        Point A;
        Point B;
        Point C;
    }
}

//c
public class Point {
    public Point() {
        int x;
        int y;
        int z;
    }
}

//e
public class HorseArrival {
    public HorseArrival(String Name, int Rank) {
    }
}

//f
public class ImgBlackAndWhite {
    public ImgBlackAndWhite(double[][] BlackTint) {
    }
}
```

---

## 6 Exercice 5

- 5 Écrire une classe `Complex` permettant de définir et de manipuler des *nombres complexes*.

Les nombres complexes constituent une extension de l'espace des nombres réels. Ils sont composés d'une partie réelle  $x$  et d'une partie imaginaire  $y$ . On les représente généralement sous la forme :  $x + yi$   $i$  étant l'unité des nombres imaginaires (la racine carrée de -1).

On peut définir les relations suivantes :

[module]	$ x + yi  = \sqrt{x^2 + y^2}$
[addition]	$(x_1 + y_1i) + (x_2 + y_2i) = (x_1 + x_2) + (y_1 + y_2)i$
[multiplication]	$(x_1 + y_1i) \cdot (x_2 + y_2i) = (x_1x_2 - y_1y_2) + (x_1y_2 + y_1x_2)i$

La classe doit permettre de créer des nombres complexes, d'afficher un nombre complexe, de calculer le module d'un nombre complexe, d'additionner et de multiplier deux nombres complexes.

Avant de coder la classe, lire (et comprendre) l'exemple d'utilisation dans la classe `TestComplex` ci-contre (qui servira de base pour tester ensuite votre classe).

Indication : Pour le calcul de la racine carrée, utiliser la fonction `Math.sqrt(x)` qui retourne un résultat de type `double`.

<pre>package s13; public class TestComplex {     public static void main(String[] args) {         Complex c1 = new Complex( 3.0, 4.0);         Complex c2 = new Complex(-1.0, 6.0);         Complex c3, c4;          c3 = c1.addOp(c2);          // c3 = c1 + c2         c4 = c3.multiplyOp(c1);     // c4 = c3 * c1          System.out.println(c1.modulus());         c3.display();         c4.display();          c1.add(c2);                 // c1 += c2         c3.multiply(c1);            // c3 *= c1          System.out.println(c1.modulus());         c1.display();         c3.display();     } }</pre>	<p><b>Affichage</b></p> <p>5.0 (2.0)+(10.0)i (-34.0)+(38.0)i</p> <p>10.198039027185569 (2.0)+(10.0)i (-96.0)+(40.0)i</p>
---	--

Code utilisé

```
package s13;

public class Complex {
    public double x;
    public double y;
    public Complex(double d, double e) {
        // constructeur
        this.x=d;
        this.y=e;
    }
    public Complex addOp(Complex c) {
        return new Complex(this.x + c.x, this.y + c.y);
    }
    public Complex multiplyOp(Complex c) {
        double tempo1,tempo2;
        tempo1 = this.x * c.x - this.y * c.y;
        tempo2 = this.x * c.y + this.y * c.x;
        return new Complex(tempo1,tempo2);
    }
    public double modulus() {
        double tempo=0;
```

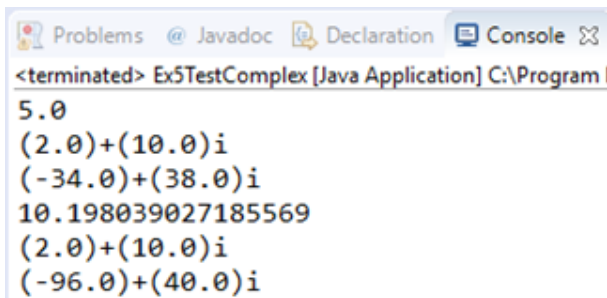
```

    double res = 0;
    tempo = (this.x*this.x)+(this.y*this.y);
    res = Math.sqrt(tempo);
    tempo = 0;
    return res;
}
public void display() {
    System.out.println("(" + this.x + ") + (" + this.y + ")i");
}
public void add(Complex c) {
    this.x += c.x;
    this.y += c.y;
}
public void multiply(Complex c) {
    this.x = (this.x*c.x) - (this.y*c.y);
    this.y = (this.x*c.y) - (this.y*c.x);
}
}

```

---

Résultat à la console



```

<terminated> Ex5TestComplex [Java Application] C:\Program I
5.0
(2.0)+(10.0)i
(-34.0)+(38.0)i
10.198039027185569
(2.0)+(10.0)i
(-96.0)+(40.0)i

```

## 7 Conclusion

Grâce à ce Travail, j'ai pu mieux comprendre le concept d'objet et de classe qui est le point clé du langage de programmation JAVA, vu que java est un langage orienté objet.