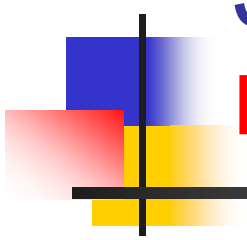
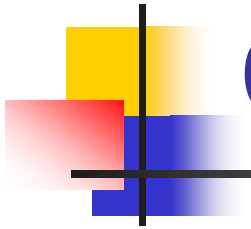


Java

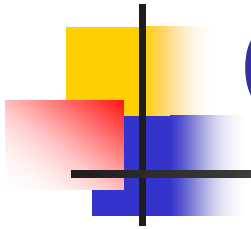
Licence Professionnelle CISII, 2009-2010

Cours 3 : Types de données



Cours 3-TD 3

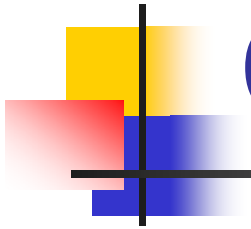
- Exercice 1
  - Créer une classe TableauDeRectangles avec les méthodes suivantes :
    - TableauDeRectangles() constructeur sans paramètre qui crée un TableauDeRectangles en déclarant un tableau de 10 rectangles,
    - TableauDeRectangles(int n) constructeur qui crée un TableauDeRectangles en déclarant un tableau de n rectangles,
    - void set (int i, Rectangle r) méthode qui met (en écrasant le rectangle se trouvant là auparavant) le rectangle r à l'indice i,
    - void inserer(int i, Rectangle r) méthode qui insère le rectangle r à l'indice i (en déplacant des rectangles vers la droite si nécessaire),
    - boolean rechercher(Rectangle r) méthode qui retourne vrai si le tableau contient un Rectangle s tel que r.compareTo(s) est 0.
    - void toutesLesSurfaces() qui affiche la surface de chaque rectangle,
    - toString() qui retourne une représentation en chaines de caract`eres de chaque rectangle contenu dans le tableau.
    - Tester l'utilisation.



Cours3-TD3

Comparaison d'objets

- Java propose une interface appelée Comparable avec une seule méthode pour comparer des objets sur lesquels il existe une relation d'ordre
- Cette méthode est définie avec les profils suivants
  - public int compareTo(String s)
  - Public int compareTo(Object o)
- Elle s'utilise comme suit :
  - element1.compareTo(element 2)
- Comme l'indique son type, elle retourne :
  - Un entier négatif si element1 arrive avant element2
  - Un entier nul si element1 est égal à element1
  - Un entier positif si element1 arrive après element1



Cours3-TD3

- toString
  - Fait parti de la classe Object
  - toString renvoie un objet de type String
  - Utilisée pour imprimer un objet quel qu'il soit, laissant libre choix au programmeur pour donner une représentation imprimable

```
public class Point {
        private int x;
        private int y;

        public Point(int x, int y) {
          this.x = x;
          this.y = y;
        }

        public void translater(int x, int y) {
          this.x = this.x + x;
          this.y = this.y + y;
        }

        public int compareTo(Point p){
            if ((this.x == p.x) &&
            (this.y == p.y)){return 0;}
            else if (this.x < p.x){
              return -1;
                }
                return 1;
        }

        public void setX(int x) {
                this.x = x;
        }

        public void setY(int y) {
                this.y = y;
        }

        public int getX() {
                return x;
        }

        public int getY() {
                return y;
        }

        public String toString() {
                return ("("+ x +"," + y + ")");
        }
}
```

```
public class Rectangle {
        private Point infGauche;
        private Point supDroit;

        public Rectangle() {
          infGauche = new Point(0,0);
          supDroit = new Point(1,1);
        }

        public Rectangle(int x1, int y1, int x2,
          int y2) {
          infGauche = new Point(x1,y1);
          supDroit = new Point(x2,y2);
        }

        public Rectangle(Point p1, Point p2) {
          infGauche = p1;
          supDroit = p2;
        }

        public int surface() {
          return (supDroit.getX()-infGauche.
          getX())* (supDroit.getY() –
          infGauche.getY());
        }
```

```
public int compareTo(Rectangle r) {
        if
      ((this.infGauche.compareTo(r.infGauche)
      == 0 ) &&(this.supDroit.
      compareTo(r.supDroit) == 0)){ return 0;}
       return (this.infGauche. compareTo
      (r.infGauche) );}

public void translater (int a,int b){
      infGauche.setX(infGauche. getX() + a);
      infGauche.setY(infGauche.getY() + b);
      supDroit.setX(supDroit.getX() + a);
      supDroit.setY(supDroit.getY() + b);
}

public String toString(){
      return ("inferieur gauche : " +infGauche
      + " superieur droit : " + supDroit);
}
}
```

```
public class TableauDeRectangles {
        Rectangle [] t;
        public TableauDeRectangles() {
          t = new Rectangle[10];
        }

        public TableauDeRectangles(int n)
      {
            t = new Rectangle[n];
        }

        public void set (int i, Rectangle r){
            t[i] = r;
        }

        public void inserer(int i, Rectangle
      r) {
            if (t[i] != null){
               int j = t.length-1;
               while (j > i){
                   t[j] = t[j-1];
                   j--;
                  }
            }
          t[i] = r;
        }
```

```
public boolean rechercher(Rectangle r){
    for (int i=0; i<t.length;i++){
       if (t[i] != null)
       if (r.compareTo(t[i]) == 0)
           return true;
     }
     return false;
}

public void toutesLesSurfaces() {
     for (int i=0;i<t.length;i++)
     if (t[i]!=null)
        System.out.println(t[i] + " surface =
     " +t[i].surface());
}

public String toString() {
     String s = "";
     for (int i=0;i<t.length;i++) {
       if (t[i] != null)
             s = s + t[i].toString() +"\n";
      }
       return s;
}
}
```

```
// fichier TestRectangle.java
public class TestRectangle {
  public static void main(String
    args []) {

    Rectangle r1= new
     Rectangle(0,0,3,4);
    Rectangle r2= new
     Rectangle(0,0,5,6);
    Rectangle r3 = new
     Rectangle(0,0,7,8);

    TableauDeRectangles t =
     new TableauDeRectangles();
    t.set(0,r1);
    t.set(1,r2);
    System.out.println(t);
    t.toutesLesSurfaces();
    System.out.println();

    //on augmente la taille de t
    t = new TableauDeRectangles(5);
      t.inserer(0,r1);
      t.inserer(1,r2);
      System.out.println(t);
      t.toutesLesSurfaces();
      System.out.println();

      t.inserer(0,r3);
      System.out.println(t);
      t.toutesLesSurfaces();

      System.out.println(t.rechercher(r2));
      Rectangle r4 = new
       Rectangle(1,1,3,4);
      System.out.println(t.rechercher(r4));
  }
}
```