01234567890123456789012345

Université d'Aix-Marseille III - Licence de Math-Info

I4 : Programmation Objet - TP n^o1

1 Prise en main de l'environnement

Editer, compiler (commande `javac` avec la variable `classpath` contenant les bons chemins) et exécuter (commande `java`) le programme suivant :

```
public class PremierEssai
{
    public static void main(String args[])
    {
        System.out.println("Ca marche");
    }
}
```

2 Premier programme

Ecrire un programme (une suite d'instructions dans une méthode `main`) qui lit un entier n au clavier, crée un tableau de n entiers, remplit ce tableau avec des entiers lus au clavier puis affiche le plus petit entier du tableau (en parcourant ce tableau). En plus de la méthode `main`, vous ajouterez (sans essayer pour l'instant de la comprendre) la méthode suivante, qui permet de lire un entier au clavier :

```
static int litInt()
{
    String s = "";
    BufferedReader b =new BufferedReader(new InputStreamReader (System.in));
    try
    {
        s=b.readLine();
    }
    catch(java.io.IOException e)
    {
        System.out.println ("Erreur de lecture");
        System.exit(0);
    }
    return Integer.parseInt(s);
}
```

3 Classes et instances

Ecrire la classe `Verre` ayant les deux attributs entiers `contenance` et `quantite`, un constructeur prenant en paramètre la contenance (en cl) du verre créé, la méthode `void emplir(int q)` qui ajoute q cl de liquide à la quantité contenue dans le verre et la méthode `boire(int q)` qui fait l'inverse.

Ecrire la classe `Bouteille` ayant un attribut entier `quantite` et un attribut booléen `est_ouverte`, un constructeur prenant en paramètre la quantité initiale (en cl) de la bouteille créée, les méthodes `void ouvrir()` et `void fermer()` et la méthode `void verser_dans(Verre v, int q)` qui verse la quantité q de liquide de la bouteille dans le verre v.

Ajouter une méthode `main` à la classe `Bouteille` pour exécuter la suite d'opérations suivante : création d'une bouteille d'orange et d'une bouteille de vodka d'un litre chacune, création d'un verre de contenance

20cl, versement de 8cl de vodka puis de 12cl d'orange dans le verre, puis consommation avec modération de tout le verre.

4 Composition de classes

Nous allons modéliser des points et des droites dans un espace à deux dimensions. Un point peut se modéliser grâce à ses coordonnées : son abscisse et son ordonnée. Une droite peut se modéliser grâce au couple (a,b) de l'équation $y = a.x + b$ représentant l'ensemble des coordonnées (x,y) des points qui appartiennent à cette droite.

3.1 *Ecrire la classe* **Point** *qui contient entre autres les méthodes suivantes* :

- `public float abscisse()` qui retourne l'abscisse du point.
- `public float ordonnee()` qui retourne l'ordonnée du point.
- `public boolean confondu(Point p)` qui indique si le point est confondu avec le point p (ils ont les mêmes abscisse et ordonnée).
- `public float distance(Point p)` qui retourne la distance du point avec le point p. Rappel : la distance entre deux points de coordonnées (x_1,y_1) et (x_2,y_2) est $\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$. On pourra utiliser la méthode de classe `static public double sqrt(double)` du paquetage `Math`.
- Un constructeur qui prend comme paramètres les deux valeurs de type `float` des coordonnées du point.

3.2 *Ecrire la classe* **Droite** *qui contient entre autres les méthodes suivantes* :

- `public boolean appartient(Point p)` qui indique si le point p appartient à la droite. Rappel : un point appartient à une droite si et seulement si ses coordonnées vérifient l'équation de la droite.
- `public boolean estParallele(Droite d)` qui indique si la droite d est parallèle à la droite. Rappel : deux droites qui ont pour équation $y = a.x + b$ et $y = a'.x + b'$ sont parallèles si et seulement si $a = a'$.
- `public Point intersection(Droite d)` qui retourne le point d'intersection entre la droite d et la droite. On présupposera que les droites ne sont pas parallèles et on ne fera donc pas ce test dans la méthode. Rappel : deux droites non parallèles qui ont pour équation $y = a.x + b$ et $y = a'.x + b'$ s'intersectent en point de coordonnées $(\frac{b'-b}{a-a'}, a\frac{b'-b}{a-a'} + b)$.
- Un constructeur qui prend comme paramètres les deux valeurs de type `float` du couple (a,b) de l'équation $y = a.x + b$ de la droite.
- Un constructeur qui prend comme paramètres deux points (de type **Point**) pour définir la droite. Rappel : une droite définie par deux points de coordonnées (x_1,y_1) et (x_2,y_2) a pour équation $y = \frac{y_1-y_2}{x_1-x_2}x + \frac{x_1y_2-x_2y_1}{x_1-x_2}$.

3.3 *Compléter la classe* **Droite** *afin qu'elle constitue un programme qui définit trois points de coordonnées (1,2), (2,3) et (3,5), qu'elle définisse une droite à partir des deux premiers points puis qu'elle affiche si le troisième point appartient à la droite.*