Java DUT 1 Feuille TP1
Université Paris-Est Marne-la-Vallée

Exercice 1.—

a) Utilisez un éditeur de texte pour créer le fichier de nom HelloWorld.java

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

b) Compilez le programme avec `javac HelloWorld.java` et observez les fichiers dans le répertoire.

b) Exécutez le programme avec `java HelloWorld`.

Exercice 2.—

a) Créez dans un fichier `ToUpper.java` une classe `ToUpper` qui contient une méthode `main` à compléter

```
public class ToUpper {
   public static void main(String[] args) {
         // ......
   }
}
```

pour que, lorsqu'on lance `java ToUpper mot1 mot2`, le programme affiche les arguments `mot1` et `mot2` en majuscule.

Aide. Les arguments `mot1` et `mot2` seront les éléments d'indice 0 et 1 du tableau `args` qui contient des chaînes de caractères. Pour mettre une chaîne de caractères en majuscule, on utilisera la méthode `toUpperCase` de la classe `String`. Regarder la documentation pour savoir comment l'utiliser.

b) Modifiez la méthode `main` pour afficher tous les arguments en majuscule.

Aide. On utilisera pour parcourir tout le tableau `args` une boucle appelée "boucle foreach" similaire aux boucles Python sur les listes. La syntaxe de cette boucle est

```
for (String s: args) {
// ..... a completer
}
```

Exercice 3.—

On considère la classe `Pixel` vue en cours pour représenter des points du plan qui a deux champs `x` et `y` et un constructeur.

```
public class Pixel {
  private int x;
  private int y;
  public Pixel (int x, int y) {
    this.x = x; this.y = y;
  }
}
```

a) Ajoutez à cette classe des méthode `getX`, `getY`, `setX`, `setY`, permettant d'accéder et de modifier les champs `x` et `y` d'un Pixel.

b) Ajoutez une méthode `public String toString` qui renvoie une représentation d'un Pixel sous forme de chaîne de caractères.

c) Créez **dans un autre fichier** une classe `PixelTest` qui contient une méthode `main` où l'on va pouvoir manipuler des objets Pixel. Dans cette méthode `main`, créez deux points `p1` et `p2` de coordonnées $(4, 5)$ et $(10, 11)$ et affichez les points sous la forme suivante : `[4, 5]` et `[10, 11]`. On utilisera la méthode `toString`.

d) Pourquoi est-il plus pratique de disposer dans une classe d'une méthode `toString` plutôt que d'une méthode `print` ?

e) Créez un autre point `p3` de coordonnées $(10, 11)$ et comparer le à `p2` avec `==`. Que remarque t-on ? Écrivez une méthode qui teste l'égalité entre deux points en considérant que deux points sont égaux s'ils ont les mêmes coordonnées.

f) Écrire une méthode `distanceToOrigin` qui calcule la distance du point au point origine de coordonnées $(0, 0)$.

Aide. La méthode retournera une valeur de type `double` et on pourra utiliser la méthode statique `Math.sqrt()` de la classe `Math` pour obtenir une racine carrée.

Exercice 4.— (S'il vous reste du temps)

a) Écrire une classe `Rectangle` qui permet de représenter un rectangle dans le plan. Un rectangle contiendra deux champs qui sont les deux points (haut-gauche et bas-droite) donnant les extrémités du rectangle.

b) Ajoutez dans la classe `Pixel` une méthode `isInsideRectangle` prenant en argument un rectangle et retournant `True` si le point est inclus dans le rectangle donné en argument, et `False` sinon. Tester cette méthode.