

Programmation & Algorithmique 2

TP - Séance 1

18 février 2022

Matière visée : se familiariser avec la syntaxe de Java (en programmation impérative), variables (quelques types primitifs), appel à une méthode (static), décision (if/else), boucles, compiler et exécuter en ligne de commande.

1 Exercices introductifs

1. Veuillez créer un programme HelloWorld (fichier HelloWorld.java) dont le comportement sera d'afficher la chaîne "HelloWorld" à l'écran.
2. Compilez et exécutez.
3. Ajoutez à votre programme les variables *locales* suivantes :
 - **prenom** : une chaîne de caractères. Affectez lui votre prénom comme valeur ;
 - **age** : un entier. Affectez lui votre âge comme valeur ; et
 - **taille** : un double. Affectez lui votre taille en mètre comme valeur.
4. Modifiez votre programme, en utilisant les variables créées au point précédent, pour qu'il affiche dans la console :
Hello World!
Mon nom est <votre prénom>,
j'ai <votre age> ans et je mesure <votre taille> mètre(s) :-D...
5. Ajoutez à votre programme une méthode

```
public static void printHello(    )
```

qui prend les trois paramètres cités au point 3. Cette méthode doit afficher le même contenu que le point précédent avec les informations entrées en paramètres ;
6. Modifiez votre méthode **main**, pour qu'elle affiche les informations liées à votre voisin de gauche et votre voisin de droite (utilisez la méthode **printHello**).

2 Cercle

Le but est d'écrire un programme qui permette d'obtenir une approximation du périmètre et de l'aire d'un cercle pour un rayon donné. Pour cela vous devez :

1. Créer la classe Cercle (fichier Cercle.java) ;
2. Ajouter la méthode **perimetre** qui prend en paramètre un double **rayon**. Créer une *constante* locale **PI** de type **double** ayant pour valeur 3,14159265. Cette méthode retourne le périmètre (double) du cercle de rayon **rayon** ;
3. Ajouter une méthode **aire** qui prend en paramètre un double **rayon**. Créer une *constante* locale **PI** de type **double** ayant pour valeur 3,14159265. Cette méthode retourne l'aire (double) du cercle de rayon **rayon** ;
4. Améliorer votre programme en créant une *constante de classe* **PI** ; et
5. Compléter votre programme pour qu'il affiche dans la console l'aire et le périmètre des cercles de rayon 1, 2, 3,..., 50.

3 Les suites de nombres

Comme son nom l'indique, une suite de nombres est une série de valeurs (entières dans notre cas) qui dépendent de certains paramètres et/ou des valeurs qui la précèdent dans la suite.

Créez la classe Suite (fichier Suite.java). Il vous est demandé d'implémenter trois méthodes liées au concept de suite :

1. `suiteArithmetique(int depart, int raison, int k)`
Entrée: Trois entiers `depart`, `raison` et `k`
Sortie: Affiche les `k` premiers éléments de la suite arithmétique de raison `raison` ayant `depart` comme valeur initiale.
2. `suiteGeometrique(int depart, int raison, int k)`
Entrée: Trois entiers `depart`, `raison` et `k`
Sortie: Affiche les `k` premiers éléments de la suite géométrique de raison `raison` ayant `depart` comme valeur initiale.
3. `suiteFibonacci(int k)`
Entrée: Un entier `k`
Sortie: Affiche le k^{e} nombre de la suite de fibonacci.

Une fois ces méthodes implémentées :

- Affichez les 20 premiers nombres de la suite arithmétique ayant pour départ la valeur -200 et pour raison 99 ;
- Affichez les 32 premiers nombres de la suite géométrique ayant pour départ la valeur 1 et pour raison 2 ;
- Affichez le $20^{\text{ème}}$ élément de la suite de fibonacci.

4 Droites

Pour cet exercice, nous allons créer une série de méthodes *static* concernant le travail sur les droites. Veuillez créer un fichier `Droites.java` et y placer l'implémentation des méthodes décrites ci-dessous.

Notations :

Nous représentons une droite d dans \mathbb{R}^2 par trois valeurs $a, b, c \in \mathbb{R}$ de telle manière à ce que :

$$d \equiv ax + by = c$$

Rappelons que (a, b) est un vecteur normal de d .

De plus, nous représenterons un point de \mathbb{R}^2 par deux réels x et y .

Implémentez les méthodes suivantes (`public static`) :

1. `droite(double x1, double y1, double x2, double y2)`
Entrée: Deux points $(x1, y1)$ et $(x2, y2)$.
Sortie: Affiche dans la console les valeurs de a , b et c qui représentent la droite passant par $(x1, y1)$ et $(x2, y2)$.
2. `appartient(double a, double b, double c, double x, double y)`
Entrée: Une droite caractérisée par les paramètres a , b et c ; et un point (x, y) .
Sortie: `true` si le point appartient à la droite, `false` sinon.