TP de Java n°4

1 ECRIRE DES FONCTIONS EN JAVA

Nous avons vu en pseudo-langage qu'il est possible d'écrire des fonctions, c'est à dire diviser le code en modules, pour éviter de tout rassembler dans un seul et même bloc de code, afin d'améliorer la lisibilité de son programme. Cela permet aussi de diviser le programme en sous parties, qui pourront être ensuite réutilisées dans d'autres programmes.

On peut, en Java, faire la même chose. Il suffit d'écrire une fonction : une fonction s'écrit de la manière suivante :

```
public static <type_de_la_variable_de_retour> <nom_de_la_fonction>(<paramètre1>, <paramètre2>, …)
{
	Mettez du code ici
}
```

Si je souhaite par exemple faire une fonction MultiplierParDeux, qui prend en entrée un seul paramètre et renvoie en sortie l'entrée multipliée par deux, et que j'appelle cette fonction dans un programme Main, j'écrirai :

```
public class Main
{
	public static int MultiplierParDeux(int entree)
	{
		int sortie;
		sortie = entree * 2;
		return sortie;
	}

	public static void main(String args[])
	{
		int x = 3;
		int y = MultiplierParDeux(x);
		System.out.println(y);
	}
}
```

1. Écrivez une fonction qui affiche le contenu d'un tableau d'entier passé en paramètre. Votre fonction ne renvoie aucune valeur (elle ne fait qu'afficher de éléments, et ne calcule pas une valeur pour la renvoyer en sortie) : dans ce cas là, le type de la variable de retour est **void** (vide). Testez cette fonction avec le code du TP2, afin d'afficher le contenu du tableau généré dans cet exercice.

2. Écrivez une fonction qui prend en paramètre un tableau et renvoie en sortie la position du plus grand élément de ce tableau. Testez cette fonction sur un petit tableau pour vérifier qu'elle fonctionne.

3. Testez votre fonction sur un tableau de 100 000 cases. Combien de temps votre programme met-il à s'exécuter ? Que se passe-t-il, au niveau du temps de calcul, lorsque vous multipliez le nombre de cases par 2, 10, 20 ?

4. Faîtes une fonction de tri de tableau. Vous pouvez modifier la fonction réalisée à l'exercice 2 afin de la réutiliser ici.

5. Même question qu'au 3, mais avec la fonction de tri.

2 TABLEAUX À DEUX DIMENSIONS

1. En vous inspirant du code du TP n°2, faîtes un programme qui affiche un entier choisi au hasard entre 0 et 7. Testez plusieurs fois votre programme pour vérifier son bon fonctionnement.

2. Modifiez votre programme pour afficher des entiers choisis au hasard entre 1 et 8. Testez.

3. Écrivez du code permettant de construire un tableau d'entiers à deux dimensions de 7 cases sur 7.

4. Modifiez votre code pour que le programme construise un tableau dont la largeur et la hauteur soient choisis au hasard (au moins 1x1 case, au plus 8x8 cases).

5. Complétez votre code afin de remplir le tableau avec des valeurs choisies au hasard entre 100 et 200.

6. Écrivez une fonction qui permet d'afficher chaque ligne du tableau à l'écran.

7. Écrivez une fonction qui affiche la position du plus grand élément. Et si je souhaitais renvoyer la position du plus grand élément ?

3 LABYRINTHE

Considérez le code suivant :

```
import java.util.Random;

public class Main
{
    public static void main(String args[])
    {
        Random m = new Random();
        int y;
        int labyrinthe[][];
        labyrinthe = new int[10][10];

        for(int i=0; i<10; i=i+1)
        {
            for(int j=0; j<10; j=j+1)
            {
                y = m.nextInt(20);
                if(y>=15)
                {
                    labyrinthe[i][j] = 1;
                }
                else
                {
```

```
                    labyrinthe[i][j] = 0;
                }
            }
        }

        for(int i=0; i<10; i=i+1)
        {
            for(int j=0; j<10; j=j+1)
            {
                System.out.print(labyrinthe[i][j]);
                System.out.print(" ");
            }
            System.out.println();
        }
    }
}
```

Ce code permet d'afficher un tableau 2d contenant des 1 et des 0. Un 1 signifie un mur, et un 0 signifie une case valide. Considérez un robot qui commence son parcours à la case de coordonnées (0,0) (en haut à gauche). Écrivez un programme qui donne le chemin que le robot devra suivre pour rejoindre la case en bas à droite sachant que le robot se déplace d'une case à la fois, verticalement ou horizontalement, et qu'il ne doit jamais se retrouver sur un mur. Au cas où vous ne l'auriez pas compris, le labyrinthe est construit au hasard.