

## Algorithmique et programmation

### TP 4 : Le langage Java, itérations

## 1 Structures de contrôle de type boucles

Les structures de contrôle de type boucles permettent d'exprimer des traitements répétitifs. En Java, les boucles sont au nombre de trois : la boucle **while**, la boucle **do...while** et la boucle **for**.

### 1.1 La boucle **while**

La boucle **while** permet de répéter une séquence d'instructions tant qu'une certaine condition est vérifiée :

```
while (condition) {  
    // bloc d' instructions  
}
```

Lors de l'exécution du programme, si la condition est vérifiée (*vraie*), alors le bloc d'instructions est exécuté. Ce traitement est ensuite répété tant que la condition reste vérifiée. Si la condition n'est pas vérifiée dès le départ, le bloc d'instructions n'est pas exécuté.

La boucle **while** permet de traduire la structure « *tant que* » du langage algorithmique.

### 1.2 La boucle **do...while**

Comme la boucle **while**, la boucle **do...while** permet de répéter une séquence d'instructions tant qu'une certaine condition est vérifiée.

```
do {  
    // bloc d' instructions  
} while (condition);
```

Lors de l'exécution du programme, le bloc d'instructions est exécuté. Ensuite, si la condition est vérifiée (*vraie*), alors ce traitement est répété. La différence avec la boucle **while** est que le bloc d'expression est exécuté au moins une fois.

La boucle **do...while** permet de traduire la structure « *répéter* » du langage algorithmique.

## 1.3 La boucle **for**

La boucle **for** est un cas particulier de la boucle **while**. La boucle **for** suivante :

```
for ( expression A ; expression B ; expression C ) {  
    // bloc d' instructions  
}
```

est équivalente à la boucle **while** :

```
{  
    expression A;  
    while ( expression B ) {  
        // bloc d' instructions  
        expression C;  
    }  
}
```

Ainsi, « *expression A* » correspond à l'initialisation de la boucle, « *expression B* » correspond à la condition de la boucle, et « *expression C* » correspond à une instruction exécutée après chaque itération de la boucle.

La boucle **for** est généralement utilisée pour faire varier un compteur entre deux bornes définies. Par exemple, pour afficher les nombres entiers entre  $-10$  et  $10$ , on utilisera :

```
int i ;  
for ( i = -10 ; i <= 10 ; i++ ) {  
    System.out.println ( i );  
}
```

La boucle **for** permet ainsi de traduire la structure « *pour* » du langage algorithmique.

## 2 Exercices

### 2.1 Un kilomètre à pied...

Écrire un programme Java affichant les 15 premiers couplets de la chanson :

```
1 kilomètre à pied, ça use, ça use,  
1 kilomètre à pied ça use les souliers.  
2 kilomètres à pied, ça use, ça use,  
2 kilomètres à pied ça use les souliers.  
...
```

*NB* : Attention à l'accord du mot « kilomètre ».

### 2.2 Moyenne

Écrire un programme qui :

1. demande un nombre entier  $n$  à l'utilisateur ;
2. demande  $n$  nombres réels à l'utilisateur ;
3. affiche la somme de ces  $n$  nombres ;
4. affiche la moyenne de ces  $n$  nombres.

## 2.3 Sapins

Écrire un programme permettant de dessiner divers sapins de hauteur  $h$  donnée. Voici quelques exemples pour  $h = 5$ .

- Un sapin plein :

```
  *
 ***
*****
*****
*****
```

- Un sapin vide :

```
  *
 * *
 *  *
 *   *
*****
```

- Un sapin couché :

```
*
**
***
****
*****
****
***
**
*
```

## 2.4 Deviner un nombre

Écrire un programme qui réalise le petit jeu suivant. Tout d'abord, l'ordinateur choisit un entier  $x$  entre 1 et 100. L'utilisateur essaie ensuite de le deviner. Il entre pour cela successivement des entiers, et à chaque coup l'ordinateur lui indique si l'entier est supérieur à  $x$ , inférieur à  $x$ , ou égal à  $x$  auquel cas la partie s'arrête et le nombre de coups est affiché.

Modifier ensuite le programme afin que l'utilisateur puisse, s'il le souhaite, recommencer une partie. En quittant le programme, le meilleur score doit s'afficher.

Voici un exemple de programme choisissant quelques valeurs au hasard en utilisant la classe `Random`. Vous pouvez vous en inspirer.

```
/* Exemple de génération de nombres aléatoires */
import java.util.*;

class ChoisirNombre {

    static final Random random = new Random();

    public static void main(String[] args) {
        System.out.println("Un entier (int) entre 0 inclu et 20 exclu");
        System.out.println(random.nextInt(20));

        System.out.println("Un réel (float puis double) entre 0.0 inclu et 1.0 exclu");
        System.out.println(random.nextFloat());
        System.out.println(random.nextDouble());
    }
}
```