INF1120 – Programmation 1

Structures de contrôle

Jacques Berger

Objectifs

Introduire le switch et les boucles

Prérequis

Structures conditionnelles

Structure de contrôle

Une structure de contrôle est une instruction qui change le flot habituel d'exécution des instructions

Boucles

Les boucles permettent de répéter un bloc d'exécution un certain nombre de fois ou jusqu'à ce qu'une condition soit vraie

Boucles

Il existe 3 types de boucles

for
while
do/while

for

La boucle for est utile lorsqu'on connait d'avance le nombre de tours de boucle à effectuer

```
for (int i = 0; i < nombre; i++) {
    // à chaque tour de boucle, i est augmenté de 1
}
```

for

L'instruction for est divisée en 3 sections (la partie entre parenthèses)

La 1ère : l'initialisation

On peut déclarer une nouvelle variable et lui attribuer une valeur de départ
Cette valeur de départ sera la valeur de la variable lors du 1er tour de boucle

for

La 2ème : La condition

Après chaque tour de boucle, la condition est évaluée
Si la condition est vraie, on met à jour la variable et on fait un autre tour de boucle
Si la condition est fausse, la boucle est terminée

for

La 3ème : L'incrémentation

Après un tour de boucle, avant l'évaluation de la condition, l'incrémentation est exécutée, ce qui change la valeur de la variable pour le prochain tour de boucle

while

On se sert de la boucle while lorsqu'il n'est pas toujours nécessaire d'exécuter la boucle

```
int i = 0;
while (i < nombre) {
  // Ne pas oublier de changer la valeur de la
  //variable à la fin de la boucle pour éviter de
  //tomber dans une boucle infinie
  i++;
}
```

while

On vérifie d'abord la condition, si elle est vraie, on exécute le premier tour de boucle

Après chaque tour de boucle, on revient au début, et on réévalue la condition

do/while

Le do/while fonctionne comme le while, sauf que l'évaluation de la condition se fait à la fin du tour de boucle

Le do/while effectue toujours le premier tour de boucle

```
int i = 0;
do {
    // il faut aussi modifier la variable pour éviter
    // une boucle infinie
    i++;
} while (i < 10);
```

do/while

On fait toujours un premier tour de boucle

On évalue la condition après le tour de boucle, si la condition est vraie, on recommence au début de la boucle; si la condition est fausse, la boucle se termine

break

L'instruction break dans une boucle met immédiatement fin à l'exécution de la boucle

```
int i = 0;
while(true) {
  if (i % 15 == 0) {
    break;
  }
  i++;
}
```

continue

L'instruction continue permet de passer immédiatement au tour de boucle suivant

Elle s'utilise comme le break

```
int i = 0;
do {
  i++;
  if (i % 5 == 0) {
    continue;
  }
  System.out.println(i);
while(i < 25);
```

switch

Le switch permet de faire un choix d'instructions en fonction de la valeur d'une variable

```
int valeur = 0;
switch (valeur) {
  case 0:
    // instructions
    break;
  case 1:
    // instructions
    break;
  case 2:
    // instructions
    break;
}
```

switch

Le break sert à mettre fin au switch, sinon il continue à exécuter les instructions qui suivent (c'est parfait ce que l'on veut)

```
switch (valeur) {
  case 0:
  case 1:
    // instructions
    break;
  case 2:
    // instructions
    break;
}
```

switch

On peut ajouter une clause default pour gérer toutes les valeurs qui ne sont déjà dans des clauses case

```
switch (valeur) {
  case 0:
    // instructions
    break;
  case 5:
    // instructions
    break;
  default:
    // instructions
}
```

switch

La majorité des switch devraient avoir un default pour attraper les bogues potentiels

La valeur peut être un entier, un réel, un caractère ou même une chaîne de caractères

Plus loin...

Le livre de Java premier langage, 11ème édition
Anne Tasso
Eyrolles, 2016
Chapitres 3 et 4