Support de cours

Cours:

Initiation à la programmation (en Java)

Vidéo:

Init-JAVA-03-1-IntroIteration-pt2

Concepts (extraits des sous-titres générés automatiquement) :

Variable i. Fois i. Boucle for. Seule instruction. Valeur de l'expression i. Corps de la boucle. Instruction d'incrémentation. Condition i inférieur. Fois vraie. Mot clé for. Nouvelle instruction d'affichage. Condition i. Exécution de notre premier exemple de boucle. Tel cas. Bloc d'instructions.

vers la recherche de séquences vidéo
(dans Initiation à la programmation (en Java).)

vers la vidéo

EPFL

Itérations : introduction
(Partie 2)

Initiation à la programmation (Java)

Jamila Sam, Vincent Lepetit et Jean-Cédric Chappelier

...

notes

résumé

0m 0s

Pas-à-pas

```
for(int i = 0; i < 5; ++i) {
  System.out.println("le carre de " + i + " vaut " + i * i);
}
```

Ce qui s'affiche dans la fenêtre Terminal:

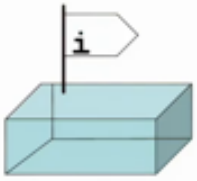
Je vais maintenant détailler pas-à-pas l'exécution

notes

résumé

0m 1s

EPFL



La variable `i` est déclarée et initialisée à 0

```
for(int i = 0; i < 5; ++i) {
  System.out.println("le carre de " + i + " vaut " + i * i);
}
```

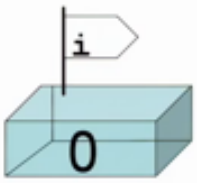
Ce qui s'affiche dans la fenêtre Terminal:

de notre premier exemple de boucle for. Tout d'abord, notre boucle for commence

notes

résumé

0m 6s



? 0<5 On entre dans

```
for(int i = 0; i < 5; ++i) {
    System.out.println("le carre de " + i + " vaut " + i * i);
}
```

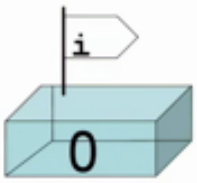
Ce qui s'affiche dans la fenêtre Terminal:

par déclarer une variable i et l'initialiser à la valeur zéro. Ensuite, on va tester la condition, qui est ici i strictement inférieur à cinq. La condition est vraie, puisque i vaut zéro, et zéro est strictement

notes

résumé

0m 14s



? 0<5 On entre dans le corps d

```
for(int i = 0; i < 5; ++i) {
    System.out.println("le carre de " + i + " vaut " + i * i);
}
```

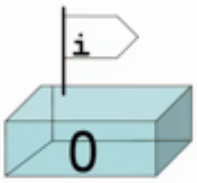
Ce qui s'affiche dans la fenêtre Terminal:

inférieur à cinq, et donc on va entrer dans le corps de la boucle.

notes

résumé

0m 37s




```
for(int i = 0; i < 5; ++i) {
    System.out.println("le carre de " + i + " vaut " + i * i);
}
```

Ce qui s'affiche dans la fenêtre Terminal:

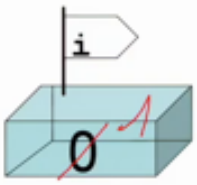
```
le carre de 0 vaut 0
```

Entrer dans le corps de la boucle, ça veut dire exécuter cette instruction-ci, qui est la seule instruction dans le corps de la boucle.

notes

résumé

0m 39s




```
for(int i = 0; i < 5; ++i) {
    System.out.println("le carre de " + i + " vaut " + i * i);
}
```

Ce qui s'affiche dans la fenêtre Terminal:

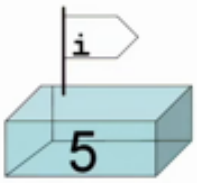
```
le carre de 0 vaut 0
```

Et cette instruction va afficher le carré de, qu'on retrouve ici, suivi de la valeur de i qui est, tout simplement zéro, suivi de vaut, et suivi de la valeur de l'expression i fois i, qui vaut simplement zéro ici. Ensuite on arrive à la fin du corps de la boucle et on va revenir sur cette ligne-ci et plus exactement, à l'instruction d'incrémentation, qui est ici ++i, qui va donc ajouter un à la variable i et i va prendre la valeur un.

notes

résumé

0m 50s



? 5<5 -> Fausse -> on sort de la boucle

```
for(int i = 0; i < 5; ++i) {
    System.out.println("le carre de " + i + " vaut " + i * i);
}
```

Ce qui s'affiche dans la fenêtre Terminal:

```
le carre de 0 vaut 0
le carre de 1 vaut 1
le carre de 2 vaut 4
le carre de 3 vaut 9
le carre de 4 vaut 16
```

On teste ensuite de nouveau la condition i strictement inférieur à cinq, qui est encore une fois vraie, et donc, on va entrer dans la boucle. Entrer dans la boucle, ça veut dire répéter cette nouvelle instruction d'affichage, avec toujours le carré de, cette fois-ci i vaut un, ensuite on affiche vaut et on affiche la valeur de l'expression i fois i qui vaut simplement un ici. On arrive encore une fois à l'instruction d'incrémentation qui va donner la valeur deux à la variable i et ainsi de suite, jusqu'à ce que la variable i ait la valeur quatre et qu'on exécute l'instruction d'incrémentation qui va faire que i va passer de quatre à cinq. On va ensuite tester la condition i strictement inférieur à cinq. Cette fois-ci, cette condition va être fausse, puisque cinq n'est pas strictement inférieur à cinq. Et comme cette instruction est fausse, on sort de la boucle.

notes

résumé

1m 25s

```
for(int i = 0; i < 5; ++i) {
  System.out.println("le carre de " + i + " vaut " + i * i);
}
```

Le programme continue en exécutant les instructions après la boucle.

La variable `i` "disparaît": Elle n'est déclarée que pour l'intérieur de la boucle.

Elle ne peut pas être utilisée à l'extérieur de la boucle.

Ce qui s'affiche dan

```
le carre de 0 vau
le carre de 1 vau
le carre de 2 vau
le carre de 3 vaut 9
le carre de 4 vaut 16
```

On va continuer après la boucle, c'est-à-dire exécuter

notes

résumé

2m 49s

Syntaxe de l'instruction `for`

```
for(déclaration_et_initialisation; condition; incrémentation) {
  bloc
}
```

- Si la condition ne devient jamais fausse, les instructions dans la boucle sont répétées indéfiniment !

les instructions qui se trouvent ici. Au passage, la variable i, qu'on avait déclarée ici, n'existe plus, c'est-à-dire qu'on ne peut plus l'utiliser, cette variable n'existe qu'à l'intérieur de boucle.

notes

résumé

2m 50s

Syntaxe de l'instruction `for`

```
for (déclaration_et_initialisation; condition; incrémentation) {
   bloc
}
```

- Si la condition ne devient jamais fausse, les instructions dans la boucle sont répétées indéfiniment !

La boucle for suit donc la syntaxe suivante. Tout d'abord, le mot clé for, puis entre parenthèses la déclaration et l'initialisation d'une variable, qui n'est pas forcément de type int, puis vient une condition qui à priori devrait porter sur

notes

résumé

3m 8s

Syntaxe de l'instruction `for`

```
for (déclaration_et_initialisation; condition; incrémentation) {
   bloc
}
```

- Si la condition ne devient jamais fausse, les instructions dans la boucle sont répétées indéfiniment !

cette variable, même s'il n'y a aucune obligation, et une incrémentation qui, elle aussi, devrait porter sur cette variable. Puis vient un bloc d'instructions, qui constitue les instructions qui seront répétées par la boucle.

notes

résumé

3m 23s

Syntaxe de l'instruction `for`

```
for (déclaration_et_initialisation; condition; incrémentation) {
   bloc
}
```

- Si la condition ne devient jamais fausse, les instructions dans la boucle sont répétées indéfiniment !

Alors je vous rappelle que les trois éléments à l'intérieur des parenthèses de la boucle

notes

résumé

3m 37s

Syntaxe de l'instruction `for`

```
for (déclaration_et_initialisation; condition; incrémentation) {
   bloc
}
```

- Si la condition ne devient jamais fausse, les instructions dans la boucle sont répétées indéfiniment !

for, sont séparés par des points-virgules, et qu'il n'y a pas de point-virgules ici. La boucle for répète les instructions qui sont dans

notes

résumé

3m 44s

Affichage d'une table de multiplication

Dans le programme suivant, la même ligne ou presque est répétée 10 fois:
Une constante prend les valeurs de 1 à 10.

```
System.out.println("Table de multiplication par 5:");

System.out.println("5 multiplie par 1 vaut " + 5 * 1);
System.out.println("5 multiplie par 2 vaut " + 5 * 2);
System.out.println("5 multiplie par 3 vaut " + 5 * 3);
System.out.println("5 multiplie par 4 vaut " + 5 * 4);
System.out.println("5 multiplie par 5 vaut " + 5 * 5);
...
```

→ il faut utiliser une boucle `for` pour éviter cette répétition.

le bloc tant que la condition ici est vraie. Si la condition ne devient jamais fausse, ces instructions seront répétées indéfiniment. Passons à un nouvel exemple de boucle fort. Supposons que je veuille afficher la table de multiplication par cinq. Sans utiliser de boucle for, je serais

notes

résumé

3m 53s

Affichage d'une table de multiplication

On peut remplacer:

```
System.out.println("5 multiplie par 1 vaut " + 5 * 1);
System.out.println("5 multiplie par 2 vaut " + 5 * 2);
System.out.println("5 multiplie par 3 vaut " + 5 * 3);
System.out.println("5 multiplie par 4 vaut " + 5 * 4);
System.out.println("5 multiplie par 5 vaut " + 5 * 5);
...
```

i ⩽ 10

par

```
for(int i = 1; i <= 10; ++i) {
  System.out.println("5 multiplie par " + i + " vaut " + 5 * i);
}
```

La variable i prend ici les valeurs de 1 à 10.

obligé de répéter dix fois quasiment, la même instruction, c'est-à-dire cette instruction-ci, qui affiche cinq multiplié par un vaut cinq fois un, suivi de cinq multiplié par deux vaut cinq fois deux, jusque cinq multiplié par dix vaut cinq fois dix. Dans un tel cas, encore une fois, il faut utiliser une boucle for et cette boucle for va s'écrire de la façon suivante. C'est-à-dire que je vais déclarer et initialiser cette fois-ci ma variable qui sert à contrôler le nombre de tour de boucle à un. Comme condition, je vais utiliser la condition i inférieur ou égal à

notes

résumé

4m 13s

dix et je vous rappelle que le symbole inférieur ou égal se note avec le caractère inférieur suivi du caractère égal. Et je vais utiliser l'opération d'incrémentation ++i. La variable i va donc prendre ici la valeur de un à dix. Cette boucle for va donc être bien équivalent à l'ensemble de ces dix instructions d'affichage, et m'afficher la table de multiplication par cinq. Le bloc d'instructions répété par une boucle for peut contenir n'importe quelle instruction, par exemple un branchement conditionnel. Voici donc un quiz, avec un branchement conditionnel à l'intérieur d'une boucle for. À votre avis, qu'affiche ce code quand on l'exécute? l'exécute?

notes

résumé

5m 1s