Support de cours

Cours:

Initiation à la programmation (en C++)

Vidéo:

Itérations - introduction (partie 2)

Concepts (extraits des sous-titres générés automatiquement) :

Initialisation d'une variable i. Condition i. Boucle for. Nombre de tours de boucle. Corps de la boucle. Valeur de l'expression i. Caractère égal. Fois i. Exécution de mon premier exemple de boucle. Seule instruction. Instruction d'incrémentation. Variable i. Mot-clé for. Bloc d'instructions. Symbole inférieur.

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

vers la vidéo

EPFL

Itérations : introduction
(Partie 2)

Initiation à la programmation (C++)

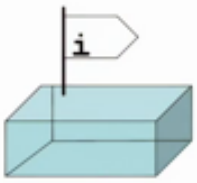
Vincent Lepetit, Jean-Cédric Chappelier et Jamila Sam

...

notes

résumé

0m 0s




```
for(int i(0); i < 5; ++i) {
  cout << "le carre de " << i << " vaut " << i * i << endl;
}
```

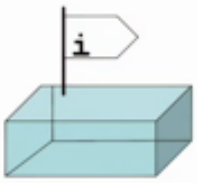
Ce qui s'affiche dans la fenêtre Terminal:

Je vais maintenant, détailler pas à pas, l'exécution de mon premier exemple de boucle for.

notes

résumé

0m 1s




```
for(int i(0); i < 5; ++i) {
    cout << "le carre de " << i << " vaut " << i * i << endl;
}
```

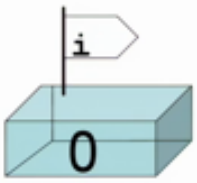
Ce qui s'affiche dans la fenêtre Terminal:

Ma boucle for commence, ici, avec la déclaration et l'initialisation

notes

résumé

0m 13s




```
for(int i(0); i < 5; ++i) {
    cout << "le carre de " << i << " vaut " << i * i << endl;
}
```

Ce qui s'affiche dans la fenêtre Terminal:

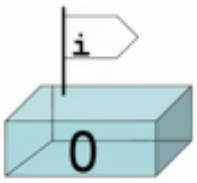
```
le carre de 0 vaut 0
```

d'une variable i, qui sera donc, initialisée ici, à la valeur 0, et qui va servir à contrôler le nombre de tours de boucle. On va ensuite, tester la condition i inférieur à 5, i vaut 0, 0 est inférieur à 5, Cette condition est donc vraie ; et on va continuer, donc, dans le corps de la boucle. "Continuer dans le corps de la boucle", ça veut dire exécuter cette instruction-ci,

notes

résumé

0m 14s




```
for(int i(0); i < 5; ++i) {
    cout << "le carre de " << i << " vaut " << i * i << endl;
}
```

Ce qui s'affiche dans la fenêtre Terminal:

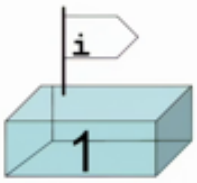
```
le carre de 0 vaut 0
```

qui va simplement faire, un affichage; donc, afficher "le carré de", en suivi de la valeur de i, c'est-à-dire 0, suivi de "vaut", et suivi de la valeur de l'expression i fois i, c'est-à-dire tout simplement 0. C'est la seule instruction dans le corps de la boucle. La boucle va maintenant boucler et revenir ici. Et, l'instruction d'incrémentation va, maintenant, être exécutée.

notes

résumé

0m 49s




```
for(int i(0); i < 5; ++i) {
    cout << "le carre de " << i << " vaut " << i * i << endl;
}
```

Ce qui s'affiche dans la fenêtre Terminal:

```
le carre de 0 vaut 0
le carre de 1 vaut 1
```

C'est-à-dire qu'on va exécuter ++i qui va faire que i va passer de la valeur 0 à la valeur 1. Ensuite, on va tester de nouveau la condition i inférieur à 5. i vaut 1, 1 est aussi inférieur à 5. Cette condition est donc toujours vraie, et de nouveau, on va continuer dans le corps de la boucle, avec cette fois-ci i qui contiendra la valeur 1. Continuer dans le corps de la boucle, ça veut dire exécuter cette instruction-ci, qui va simplement afficher, "le carré de", suivi de la valeur de i, c'est-à-dire, maintenant 1, suivi de "vaut", et suivi de la valeur de l'expression i fois i, qui vaut, cette fois-ci, tout simplement 1. On va de nouveau boucler, c'est-à-dire revenir, ici, et exécuter l'opération, l'instruction d'incrémentation.

notes

résumé

1m 23s

```
for(int i(0); i < 5; ++i) {
  cout << "le carre de " << i << " vaut " << i * i << endl;
}
```

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
```

C'est-à-dire que i va passer de la valeur de 1, à 2, et ainsi de suite. Á un moment donné, i va valoir 4. Et on va exécuter l'instruction d'incrémentation, qui va faire que i va passer de la valeur 4, à la valeur 5. Ensuite, la condition i inférieur à 5, va être testée. 5 n'est pas strictement inférieur à 5. Et donc, maintenant, cette condition va devenir fausse. Et comme elle est fausse, on va sortir de la boucle. Sortir de la boucle, ça veut dire continuer en exécutant les instructions qui sont après la boucle. Notez au passage que la variable i, qu'on

notes

résumé

2m 25s

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 !

avait déclarée au début de la boucle, ne peut plus être utilisée après la boucle. La variable i n'existe que dans le corps de la boucle. C'est-à-dire, ici.

notes

résumé

3m 25s

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, a priori, devrait porter sur 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. Alors, je vous rappelle que les trois éléments, à l'intérieur des parenthèses de la

notes

résumé

3m 38s

boucle for, sont séparés par des points-virgules, et qu'il n'y a pas de point-virgule, ici. La boucle for répète les instructions qui sont dans le bloc tant que la condition, ici, est vraie. Si la condition ne devient jamais fausse, ces instructions seront répétées indéfiniment. Alors, notez au passage, qu'en C++ 2011, il existe une autre forme de boucle for, que nous verrons, au moment du cours sur les tableaux. Continuons avec un nouvel exemple de boucle for.

notes

résumé

4m 13s

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.

```
cout << "Table de multiplication par 5:" << endl;

cout << "5 multiplie par 1 vaut " << 5 * 1 << endl;
cout << "5 multiplie par 2 vaut " << 5 * 2 << endl;
cout << "5 multiplie par 3 vaut " << 5 * 3 << endl;
cout << "5 multiplie par 4 vaut " << 5 * 4 << endl;
cout << "5 multiplie par 5 vaut " << 5 * 5 << endl;
...
```

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

Supposons que je veuille afficher la table de multiplication par 5. Si je n'utilise pas de boucle for, je serais obligé de répéter quasiment 10 fois la même instruction.

notes

résumé

4m 45s

Affichage d'une table de multiplication

On peut remplacer:

```
cout << "5 multiplie par 1 vaut " << 5 * 1 << endl;
cout << "5 multiplie par 2 vaut " << 5 * 2 << endl;
cout << "5 multiplie par 3 vaut " << 5 * 3 << endl;
cout << "5 multiplie par 4 vaut " << 5 * 4 << endl;
cout << "5 multiplie par 5 vaut " << 5 * 5 << endl;
...
```

i ⩽ 10

par

```
for(int i(1); i <= 10; ++i) {
  cout << "5 multiplie par " << i << " vaut " << 5 * i << endl;
}
```

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

C'est-à-dire, cette instruction-ci, qui affiche 5 multiplié par 1, vaut 5 fois 1. Suivi de l'instruction, 5 multiplié par 2, vaut 5 fois 2, et cetera, jusque, 5 multiplié par 10, vaut 5 fois 10. Dans un cas comme celui-là, il faut utiliser une boucle for, pour éviter cette répétition. Cette boucle for va s'écrire ainsi : j'ai initialisé la variable qui va me servir à contrôler le nombre de tours de boucle, cette fois-ci, à 1. Comme condition, je vais utiliser la condition i inférieur ou égal à 10, et je vous rappelle que le symbole inférieur ou égal se note avec le caractère inférieur, suivi du caractère égal.

notes

résumé

5m 1s

Donc, avec cette initialisation à 1, avec la condition i inférieur ou égal à 10, et avec l'incrémentation à chaque tour de boucle de i, i va prendre cette fois-ci, les valeurs de 1 à 10. Le corps de la boucle va s'écrire de cette façon-ci : c'est-à-dire qu'à chaque tour de boucle, je vais afficher la valeur de i, ainsi que la valeur de l'expression 5 fois i. Et donc, cette boucle for, va me permettre d'afficher la table de multiplication par 5. 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, dans une boucle for ; à votre avis, qu'affiche ce code, quand on l'exécute ? l'exécute ?

notes

résumé

5m 49s