Support de cours

Cours:

Initiation à la programmation (en C++)

Vidéo:

Boucles conditionnelles (partie 2)

Concepts (extraits des sous-titres générés automatiquement) :

Instruction while. Condition de la boucle. Boucle do while. Variante de boucle conditionnelle. Exécution du do while. Petits exemples basiques. Conditions de continuation. Cas d'une variable i. Corps de la boucle. Différence principale. Principe de fonctionnement. Do while. Emblée fausse. Cas précédent. Première fois.

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

vers la vidéo

EPFL

Boucles conditionnelles
(Partie 2)

Initiation à la programmation (C++)

Vincent Lepetit, Jean-Cédric Chappelier et Jamila Sam

...

notes

résumé

0m 0s

L'instruction `while...`

Il existe également la forme suivante:

```
while (condition) {
  bloc
}
```

Le principe est similaire à celui de la boucle `do...while` que nous venons de voir.

La différence est que la condition est testée **avant** d'entrer dans la boucle. Si la condition est fausse, les instructions dans la boucle ne sont donc pas exécutées.

Alors il existe des situations où il est souhaitable de tester la condition de la boucle avant même d'avoir exécuté le corps au moins une fois. Et dans ce cas-là nous allons avoir recours à une variante de boucle conditionnelle, qui est l'instruction while, qui se rédige comme suit: le mot réservé "while" suivi directement de la condition entre, toujours, paire de parenthèses fermantes ouvrantes et suivi du corps de la boucle entre accolades ouvrantes et fermantes. Le principe de fonctionnement est le même, est analogue à celui de la boucle do while donc on va répéter les traitements tant que la condition est évaluée à "true". La différence principale avec la boucle do while est que la condition est testée avant même d'entrer dans la boucle.

notes

résumé

0m 1s

Exemples

```
int i(100);
do {
  cout << "bonjour" << endl;
} while (i < 10);
```

affichera une fois `bonjour`.

Dans les 2 cas,
la condition `i < 10` est fausse.

```
int i(100);
while (i < 10) {
  cout << "bonjour" << endl;
}
```

n'affichera rien.

25

Donc si on imagine que la première fois où on aborde cette instruction, la condition ici est d'emblée fausse on va directement passer à l'instruction suivant le while et donc le bloc ne sera même pas exécuté une seule fois, ce qui est différent de l'instruction "do while" où ce bloc, on vient de le voir, est exécuté au moins une fois, toujours. Voici maintenant deux petits exemples basiques permettant d'illustrer la différence fondamentale entre un do while et un while.

notes

résumé

0m 49s

Exemples

```
int i(100);
do {
  cout << "bonjour" << endl;
} while (i < 10);
```

affichera une fois bonjour.

bonjour

false

Dans les 2 cas,
la condition `i < 10` est fausse.

```
int i(100);
while (i < 10) {
  cout << "bonjour" << endl;
}
```

false

n'affichera rien.

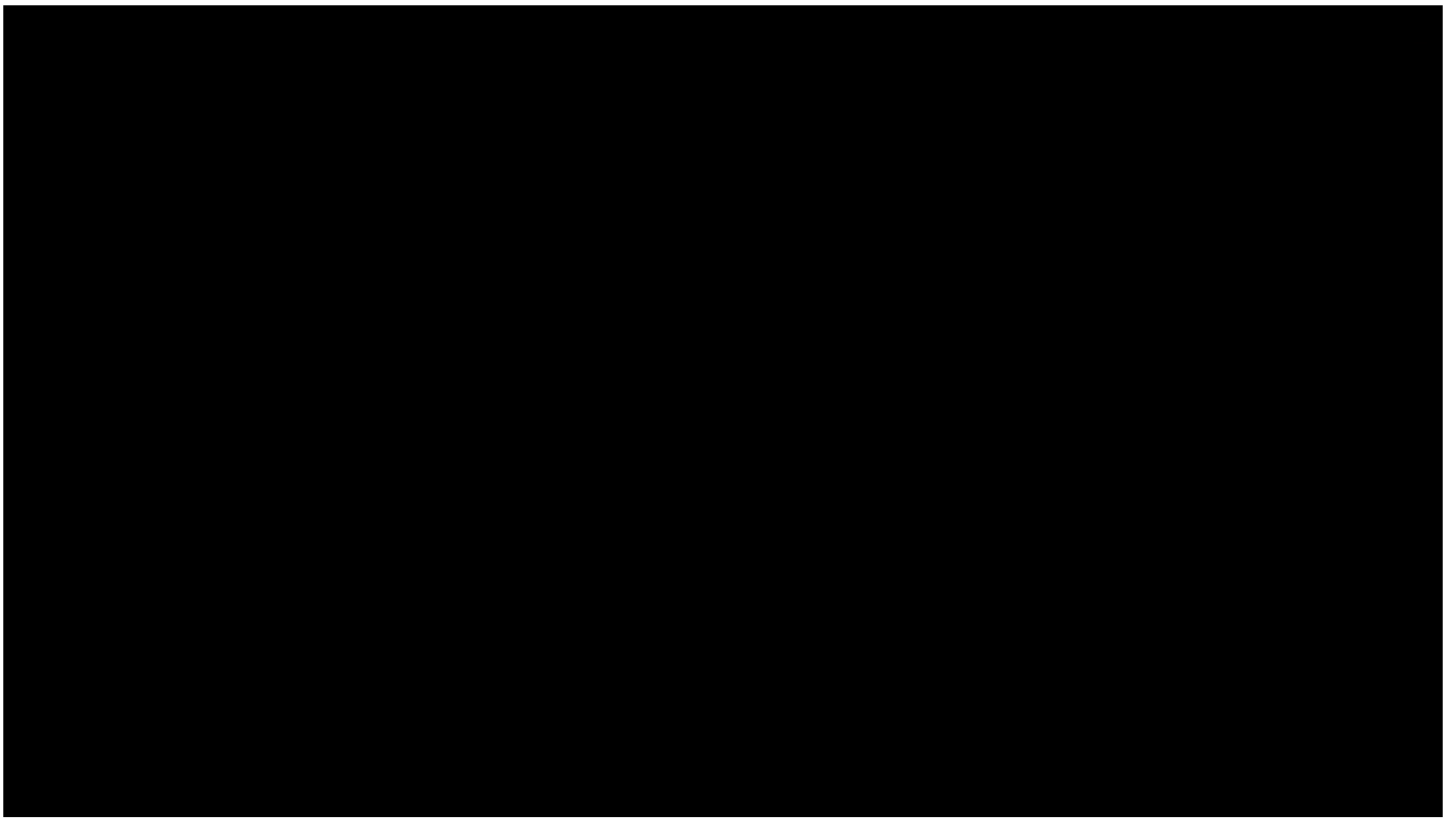
25

Nous avons ici affaire à deux boucles très similaires, dont les deux conditions de continuation sont les mêmes. Ces conditions de continuation vont dépendre dans les deux cas d'une variable i qui est initialisée à 100. Commençons par examiner ce qui se passe lors de l'exécution du do while. Lorsque nous atteignons l'exécution du do while rien ne nous empêche d'entrer directement dans le corps de la boucle et à ce moment-là, nous allons afficher sur le terminal le message bonjour. Nous atteignons ensuite cette partie de la boucle do while où il s'agit d'évaluer la condition de continuation. Ici, puisque i vaut 100, le résultat de l'évaluation sera bien évidemment "false". Ce qui veut dire que nous allons sortir de la boucle et continuer l'exécution avec les éventuelles instructions suivant le do while. Entre-temps, le message "bonjour" aura été affiché sur le terminal. Concernant la bouche while, lorsque nous abordons l'exécution de cette ligne, nous allons d'emblée évaluer la condition de continuation qui, comme pour le cas précédent, puisque nous sommes exactement dans la même situation, va être évaluée à "false", ce qui veut dire que nous

notes

résumé

1m 13s



n'allons jamais entrer dans le corps de la boucle et continuer l'exécution directement après le while Donc, dans ce cas-ci, la boucle n'affichera rien. Donc typiquement nous sommes ici dans une situation où deux boucles très similaires, un do while et un while ne font pas les mêmes choses en raison du moment où la condition de continuation est évaluée. est évaluée.

notes

résumé

2m 13s