Support de cours

Cours:

Initiation à la programmation (en C++)

Vidéo:

Boucles conditionnelles (partie 3)

Concepts (extraits des sous-titres générés automatiquement) :

Utilisation des boucles while. Nombre de répétition des traitements. Itérations de type for. Boucle for. Boucle while. Nombre d'étudiants. Do while. Condition du do while. Intérieur de cette boucle while. Nombre d'itérations. Somme de leurs notes. Cas de figure précis. Choix de boucle for. Erreurs classiques. Fameux petit point-virgule.

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

vers la vidéo

EPFL

Boucles conditionnelles
(Partie 3)

Initiation à la programmation (C++)

Vincent Lepetit, Jean-Cédric Chappelier et Jamila Sam

...

notes

résumé

0m 0s

Erreurs classiques

Il n'y a pas de `;` à la fin de la condition du `while...:`

```
while (i < 10); // !!
  ++i;
```

sera interprété comme

```
while(i < 10);
++i;
```

Le point-virgule est considéré comme le corps de la boucle, et l'instruction `++i` est **après la boucle**.

Si `i` est inférieur à 10, on entre dans la boucle pour ne jamais en ressortir puisque la valeur de `i` ne sera jamais modifiée.

En revanche, il y a un point-virgule à la fin du `do..while`:

```
do {
  ++i;
} while(i < 10);
```

Donc parmi les erreurs classiques lorsque l'on débute avec l'utilisation des boucles while et do while, il en est une qui peut donner des résultats un peu déconcertants et qui est liée à notre fameux petit point-virgule. Nous avons vu que ce point-virgule clôt l'instruction do while donc arrive après la condition du do while. Si d'aventure on s'amusait à mettre un point-virgule après la condition d'un while, on aurait un comportement inattendu et erroné de notre programme. Donc voyons ici concrètement ce qui se passerait si on mettait un point-virgule. Donc ici c'est interprété, cette instruction, comme si nous avions une boucle while avec comme corps une instruction vide, et donc il n'y a rien à l'intérieur de cette boucle while dans le corps qui permette de faire évoluer le i. Si l'on imagine que ce i avait pour valeur quelque chose d'inférieur à 10 donc par exemple 1, cette boucle va cycler indéfiniment parce que il n'y a aucun moyen de faire atteindre à i une valeur

notes

résumé

0m 1s

qui lui permette de faire sortir de la boucle, donc de dépasser 10. En réalité, ce ++i, que probablement on souhaitait mettre à l'intérieur du corps ne va jamais être atteint dans le cas de figure précis où i est inférieur à 10. Donc ceci, cette instruction va être considérée, cette instruction plus plus i, va être considérée comme étant en dehors de la bouche while. Vous connaissez désormais trois façons de répéter des instructions dans un programme: la boucle for, les

notes

résumé

1m 1s

Quand utiliser la boucle `while` ?
Quand utiliser la boucle `for` ?

Quand le nombre d'itérations (de répétitions) est connu avant d'entrer dans la boucle, utiliser `for`:

```
for(int i(0); i < nombre_d_iterations; ++i) {
```

Sinon, utiliser `while`:

– quand les instructions doivent être effectuées au moins une fois, utiliser `do...while`:

```
do {
  instructions;
} while (condition);
```

– Sinon, utiliser la forme `while`...

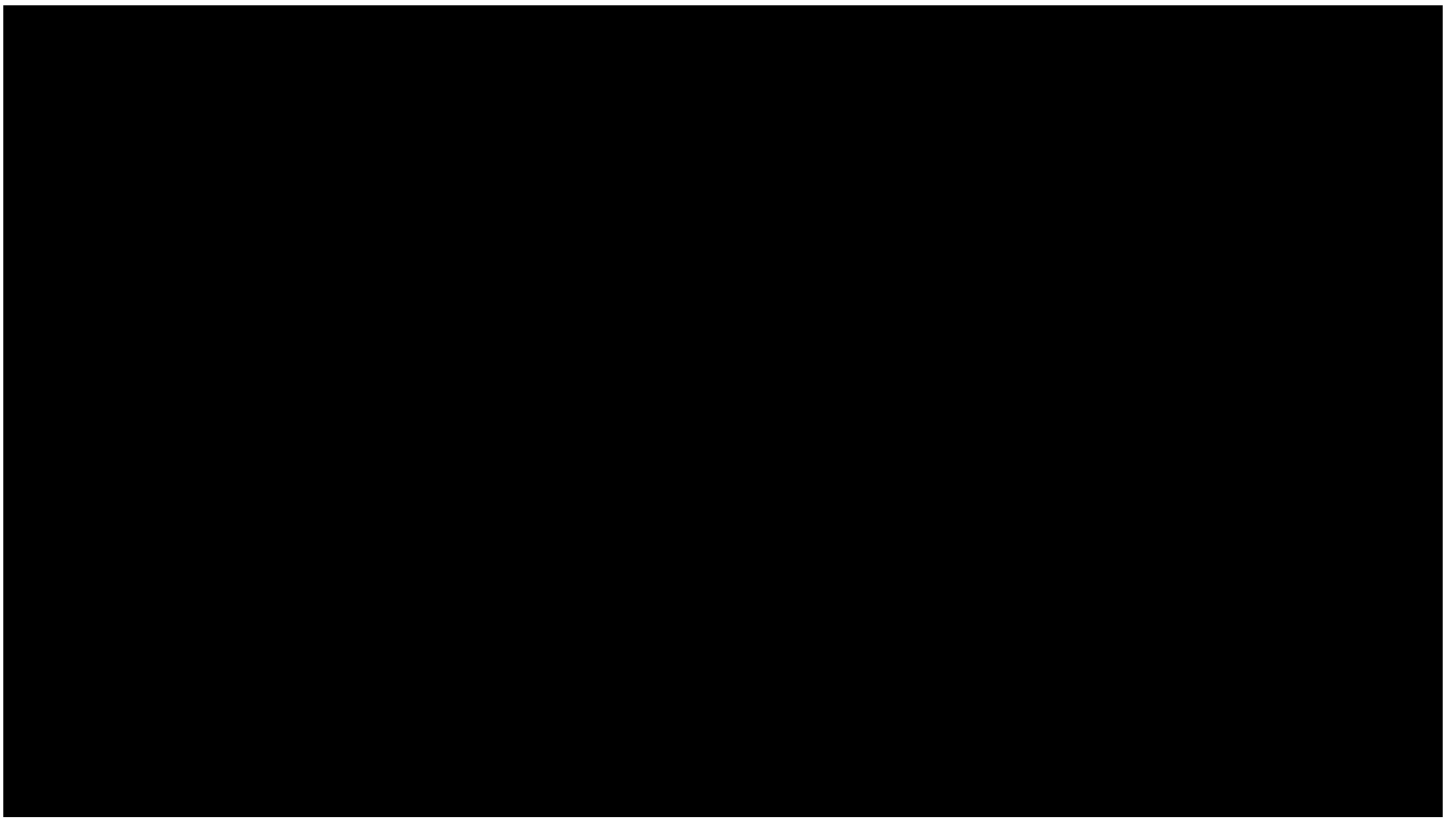
```
while (condition) {
  instructions;
}
```

itérations de type for que nous avons vue dans une séquence précédente et ce que nous venons de voir, les boucles conditionnelles qui peuvent prendre la forme soit d'un while, soit d'un do while. On va se poser la question maintenant de comment choisir entre l'une ou l'autre des formes. Le choix relève de principes assez simples. Si le nombre de répétition des traitements est connu d'avance, est connu à priori, le choix va naturellement se porter sur une itération de type fort. Imaginez par exemple que je souhaite calculer la moyenne de tous les étudiants qui suivent ce cours, je connais à priori le nombre d'étudiants suivant ce cours et donc pour calculer la somme de leurs notes et calculer la moyenne, je vais plutôt partir sur un choix de boucle for puisque le nombre d'itérations est connu a priori. Sinon, si le nombre d'itérations n'est pas connu à priori, je vais orienter mon choix vers l'utilisation d'une boucle conditionnelle de type while ou do while et à ce moment là, comment faire le choix entre les deux? Et bien simplement, on va distinguer entre le cas où l'on a besoin d'exécuter au moins une fois le corps ou pas. J'ai besoin d'exécuter au moins une fois le corps par exemple lorsque j'interagis avec l'utilisateur. Typiquement, je demande une valeur à l'utilisateur et je veux que cette valeur soit comprise entre deux bornes. J'ai besoin que l'utilisateur saisisse au moins une

notes

résumé

1m 25s



fois la valeur avant de tester si elle satisfait la condition et à ce moment-là je vais plutôt orienter mon choix vers une boucle de type do while. Sinon, et bien le choix se portera naturellement sur une forme de type while donc une boucle où l'évaluation de la condition se fait avant même d'entrer dans le corps de la boucle. boucle.

notes

résumé

2m 37s