Support de cours

Cours:

Initiation à la programmation (en C++)

Vidéo:

Définitions (partie 2)

Concepts (extraits des sous-titres générés automatiquement) :

Instruction return. Instructions return. Return de façon. Exemple simple. Deuxième remarque. Troisième remarque. Variable m. Instruction if. Corps de cette fonction. Fonction max. Type de retour. Type double. Fonction lire. Seul return. Bibliothèque standard.

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

vers la vidéo

EPFL

Fonctions : définitions
(Partie 2)

Initiation à la programmation (C++)

Vincent Lepetit, Jean-Cédric Chappelier et Jamila Sam

...

notes

résumé

0m 0s

Remarques sur l'instruction `return` (2/4)

EPFL

Le type de la valeur retournée doit correspondre au type dans l'en-tête :

```
double f() {
  bool b(true);
  ...
  return b; // Erreur : mauvais type
}
```

Quelques remarques concernant l'instruction return sur lesquelles il est peut-être important d'insister. Tout d'abord, on peut placer plusieurs instructions return dans une même fonction. Prenons un exemple simple ici d'une fonction qui va retourner le maximum de deux valeurs, disons ici, a et b. J'appelle ma fonction max2 parce que tout simplement il existe déjà une fonction max dans la bibliothèque standard du C++. Regardons le corps de cette fonction, on commence par déclarer une variable m et puis, si a est plus grand que b, alors on va recopier a dans m et si b est inférieur ou égal à a, alors à ce moment là, la valeur de m sera la valeur de b et puis on retourne la valeur de m. Mais, on aurait pu très bien écrire ce code ici avec deux return de façon un petit peu plus compacte, le prototype reste le même mais ici, si a est plus grand que b, on return a ce qui aura l'effet que si effectivement a est plus grand que b on va évaluer cette expression et la retourner et on va s'arrêter ici. Le return fait que l'on s'arrête à cet endroit-là. Si par contre b était inférieur ou égal à a alors à ce moment-là ce qui va se passer c'est que cette instruction if va directement sauter ici et à ce moment-là, ça sera ce return là qui sera exécuté. Notez bien qu'il y aura qu'un seul return exécuté pour un appel donné de cette fonction. Deuxième remarque, l'expression qui suit une instruction return dans une fonction doit bien sûr être de même type que le type de retour indiqué dans l'entête de cette fonction.

notes

résumé

0m 1s

Remarques sur l'instruction `return` (3/4)

`return` doit être la toute dernière instruction exécutée :

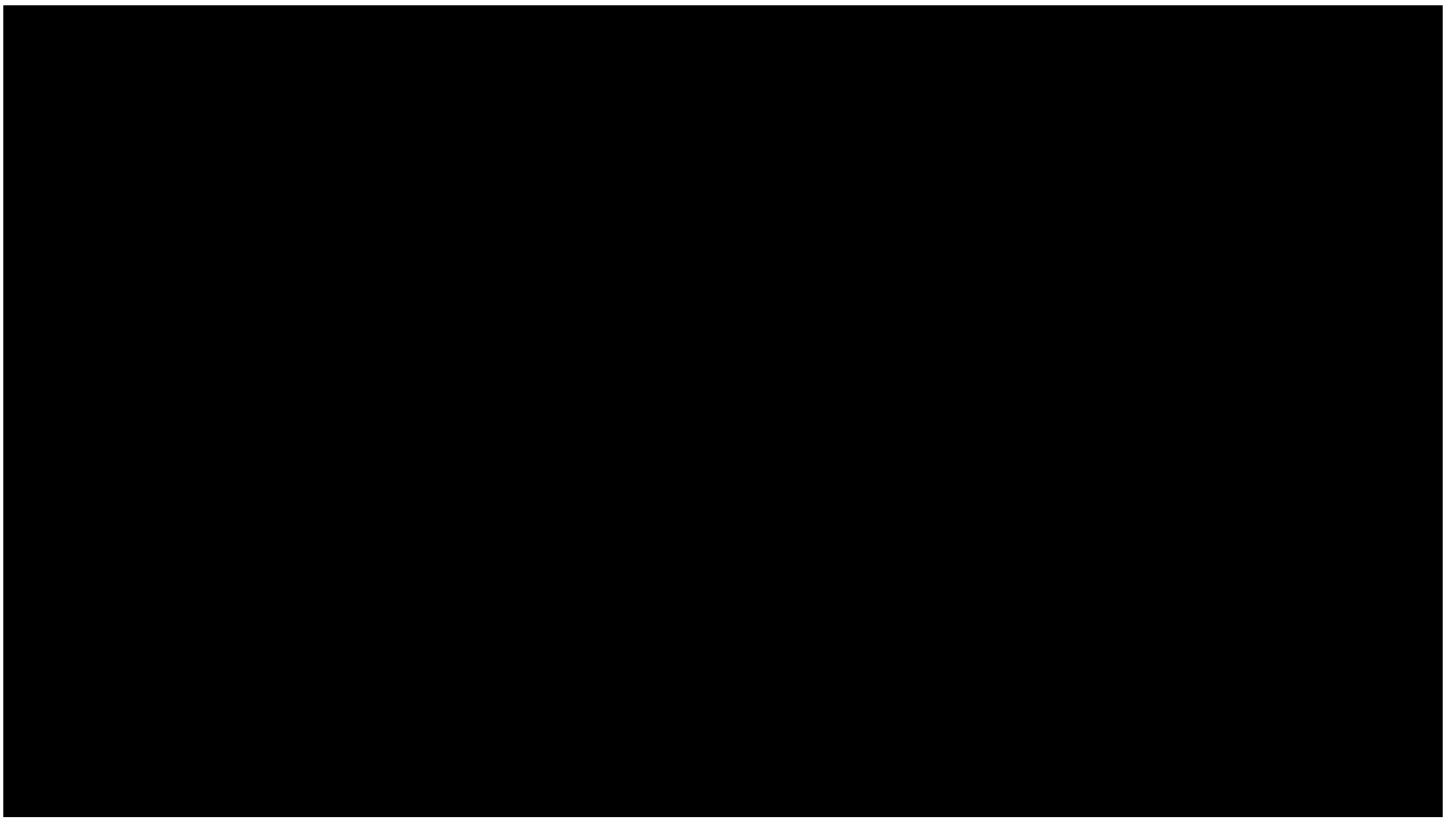
```
double lire() {
  cout << "Entrez un nombre : ";
  double n(0.0);
  cin >> n;
  return n;
  cout << "entré : " << n << endl; // Jamais exécuté
}
```

Par exemple ici, si j'ai une fonction f qui ne prend pas d'arguments mais qui retourne un double, alors à ce moment-là, tous les return dans cette fonction devront être de type double et ici j'ai une expression qui retourne b b de type bool 1, ceci n'est pas correct, cela va me générer une erreur. Troisième remarque concernant l'instruction return, elle doit être bien sûr la dernière instruction exécutée puisque l'instruction return met fin à l'exécution de la fonction, ça n'a pas de sens de mettre des instructions derrière return Prenons l'exemple ici d'une fonction lire qui doit retourner un double en l'ayant lu sur l'entrée standard. Elle commence par afficher un message sur la sortie standard puis ensuite, elle déclare une variable de type double de nom n et initialisé à 0.

notes

résumé

1m 37s



Elle lit la valeur de ce double sur l'entrée standard et enfin elle retourne au reste du programme la valeur de n. A ce moment-là, la fonction lire est terminée. Il n'y a donc absolument aucune raison de mettre une instruction ici derrière, c'est complètement erroné. c'est complètement erroné.

notes

résumé

2m 25s