Support de cours

Cours:

Initiation à la programmation (en C++)

Vidéo:

Utilisation des vector (partie 3)

Concepts (extraits des sous-titres générés automatiquement) :

Fonctions spécifiques. Nom du tableau. Type d'itération. Nom de la variable. Taille du tableau. Façon d'itérer. Type de retour de cette fonction. Fonction spécifique. Tableau des âges. Éléments d'un tableau. Seul tableau. Taille de l'index. Âge variable. Nouveau type. Façon d'itérer.

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

vers la vidéo

EPFL

Tableaux : utilisation des `vector`

(Partie 3)

Initiation à la programmation (C++)

Vincent Lepetit, Jean-Cédric Chappelier et Jamila Sam

...

notes

résumé

0m 0s

C++11 Itérations sur un tableau (2)

EPFL

Notez que les itérations C++11
`for(auto nom_de_variable : tableau)`
ne permettent pas :

for (auto age : ages)

- d'itérer sur plusieurs tableaux à la fois :
  par exemple il n'est pas possible de traverser en une passe deux tableaux pour les comparer, les mélanger, ...
- d'accéder à plusieurs éléments
  par exemple on ne peut pas comparer un élément du tableau et son suivant
- de sauter des éléments, d'avancer de deux en deux, ...

Cette façon d'écrire des boucles est donc très utile parce qu'il est compact, il suffit d'écrire simplement le mot-clé pour indiquer que l'on veut itérer, pour mettre le nom de la variable, deux points, le nom du tableau, comme nous l'avions écrit précédemment, écrivez simplement pour "(auto", indiquez que nous avons un âge variable qui itère sur le tableau des âges. Voilà, plus rien à écrire, donc c'est très compact. Cette façon d'itérer ne vous permet pas d'itérer sur plusieurs tableaux à la fois. En effet, nous avons ici un seul tableau sur lequel nous pouvons itérer, nous ne pourrions donc pas par exemple comparer des tableaux ou insérer un tableau dans un autre... Nous ne pouvons pas non plus accéder à plusieurs éléments, parce qu'à un moment donné, nous avons une seule variable qui indique une seule entrée. Nous ne pouvons donc pas changer de position par exemple les éléments d'un tableau en prenant un élément et son élément suivant dans une même boucle. Enfin, nous ne pouvons pas ignorer les éléments du tableau. Cette façon d'itérer se déroule sur les différents éléments du tableau les uns après les autres, on ne peut pas sauter un élément pour passer à l'élément suivant. Donc si nous avons besoin de l'un de ces trois cas, parcourir plusieurs tableaux ou accéder à plusieurs éléments, ou en sautant des éléments pendant une itération, nous devrons utiliser une autre boucle for,

notes

résumé

0m 1s

Itérations sur un tableau (3)

EPFL

On peut aussi itérer sur un tableau avec un `for` classique... ...mais il faut pour cela connaitre la taille.

Il existe pour cela une « *fonction spécifique* », c'est-à-dire propre à chaque variable `vector` : la fonction `size()` :

`tab.size()` retourne la taille du tableau `tab`.
Cette taille est de type `size_t`, un « `int` » particulier, toujours positif.

On pourra donc écrire :

```
for(size_t i(0); i < tab.size(); ++i)
```

Exemple :

```
vector<int> ages(5);

for(size_t i(0); i < ages.size(); ++i) {
  cout << "Age de l'employé " << i+1 << " ? ";
  cin >> ages[i];
}
```

utiliser les boucles for que nous avons vues dans le cours à propos des instructions de contrôle. Dans ce type d'itération, nous aurons généralement besoin de indique la taille du tableau, par exemple pour indiquer le dernier élément, qui a la taille de l'index moins un. Alors, comment pouvons-nous indiquer la taille du tableau ? Pour y parvenir, nous utiliserons ce que l'on appelle des fonctions spécifiques, dans les langages orientés objet, on parle de méthodes, qui peut renvoyer la taille du tableau. une fonction spécifique est simplement une fonction qui est associé à une variable, et que nous utilisons de la manière suivante,

notes

résumé

1m 25s

si j'ai une variable, un tableau dynamique nommé tab, pour connaître sa taille, j'utiliserai tab.size(), la fonction size est appliquée à la variable tab, tab étant le nom du tableau. Le type de retour de cette fonction spécifique, la taille, est un nouveau type connu sous le nom de size_t, c'est juste un int particulier, qui a la garantie propriété, selon la norme, d'être toujours positive. Comme le dit la norme, le type de retour de la taille de la fonction spécifique, des tableaux dynamiques, est size_t, vous devez déclarer, lorsque vous utilisez des variables pour stocker une telle taille, il faut les déclarer de type size_t. pour stocker une telle taille, il faut les déclarer de type size_t.

notes

résumé

2m 1s