Support de cours

Cours:

Initiation à la programmation (en C++)

Vidéo:

Déclaration et initialisation des vector (partie 1)

Concepts (extraits des sous-titres générés automatiquement) :

Tableaux dynamiques. Tableau dynamique. Liste des valeurs. Variable de ce type. Tableau des âges des étudiants. Éléments du tableau. Ensemble d'éléments de même type. Variable de type. Valeur identique. Type des éléments d'un tableau. Cours de l'exécution du programme. Initialisation explicite. Tableau de type. Variable de ce type vector. Troisième cas d'initialisation.

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

vers la vidéo

EPFL

Tableaux : déclaration et initialisation des `vector`

(Partie 1)

Initiation à la programmation (C++)

Vincent Lepetit, Jean-Cédric Chappelier et Jamila Sam

...

notes

résumé

0m 0s

Les tableaux en C++

En C++, on utilise :

		taille initiale connue *a priori* ? non	taille initiale connue *a priori* ? oui
taille pouvant varier lors de l'utilisation du tableau ?	oui	`vector`	(`vector`)
	non	(`vector`)	`array` (C++11) tableaux « à la C »

Dans cette séquence, nous allons étudier les tableaux dynamiques de C++

notes

résumé

0m 1s

Les vector

Un **tableau dynamique**, est une *collection* de données homogènes, dont *le nombre peut changer* au cours du déroulement du programme, par exemple lorsqu'on ajoute ou retire des éléments au/du tableau.

Les tableaux dynamiques sont définis en C++ par le biais du type

`vector`

Pour les utiliser, il faut tout d'abord importer les définitions associées à l'aide d'un `include` :

```
#include <vector>
```

les "vectors". Un tableau dynamique c'est un tableau, c'est à dire une collection homogène, un ensemble d'éléments de même type, dynamique c'est à dire dont la taille peut changer au cours de l'exécution du programme, comme par exemple le tableau des âges des étudiants qui suivent un cours, qui pourrait évoluer, au fur et à mesure des inscriptions des étudiants à un cours. Le type "tableau dynamique en C++" est représenté par le type "vector".

notes

résumé

0m 5s

Déclaration d'un tableau dynamique

EPFL

Une variable correspondant à un tableau dynamique se déclare de la façon suivante :

```
vector<type> identificateur;
```

où *identificateur* est le nom du tableau et *type* correspond au type des éléments du tableau.

Exemple :

```
#include <vector>
...
vector<int> tableau;
```

Le type des éléments peut être n'importe quel type C++ valide.

Pour pouvoir utiliser le type "vector", il faut indiquer au compilateur que l'on veut ce type et donc inclure la bibliothèque correspondante avec la directive #include au début de son programme. Pour pouvoir utiliser un tableau dynamique, il faut bien sûr déclarer une variable de ce type, c'est à dire déclarer une variable de type "vector". Cela ce fait comme d'habitude en indiquant le nom de la variable précédé par le type et ici le type, c'est vector avec entre signes comme ça, entre ce qu'on appelle chevrons, entre signe inférieur et signe supérieur, le type de chacun des éléments du tableau. Donc par exemple, si je veux déclarer un tableau qui va contenir des entiers,

notes

résumé

0m 37s

C++11 Initialisation d'un tableau dynamique

EPFL

En C++11, il y a cinq façons d'initialiser un tableau dynamique :

- vide
- avec un ensemble de valeurs initiales
- avec une taille initiale donnée et tous les éléments « nuls »
- avec une taille initiale donnée et tous les éléments à une même valeur donnée
- avec une copie d'un autre tableau

ce que je vais faire, c'est déclarer une variable ici qui est un tableau de type vector int étant ici le type de chacun des éléments du tableau. J'aurais bien sûr au préalable, pour que tout ceci compile, pas oublié de mettre l'inclusion de la bibliothèque "vector". Comme déjà mentionné, le type des éléments d'un tableau peut être n'importe quel type disponible en C++. Donc ce que vous mettez ici entre chevrons dans la définition du type c'est n'importe quel type valide et y compris éventuellement un autre tableau, on y reviendra un petit peu plus tard. Lorsque vous déclarez une variable de type tableau, vous pouvez aussi, comme toutes les autres variables, les initialiser. Il existe en fait, depuis la nouvelle norme C++ 2011, cinq façons d'initialiser un tableau dynamique.

notes

résumé

1m 25s

C++11 Initialisation d'un tableau dynamique

EPFL

tableau vide (sans aucun élément !) :

```
vector<int> tableau;
```

---

avec des valeurs initiales différentes (C++11) :

```
vector<type> identificateur({ val1, ..., valn });
```

Exemple :

```
vector<int> ages({ 20, 35, 26, 38, 22 });
```

Note : on peut aussi écrire

```
vector<int> ages = { 20, 35, 26, 38, 22 };
```

Vous pouvez soit l'initialiser comme vide, soit donner explicitement la liste des valeurs que vous voulez mettre au départ, soit lui donner une taille avec éventuellement une valeur identique pour tous les éléments que vous allez mettre au départ dans ce tableau, soit faire une copie d'un tableau qui existait déjà. Regardons tour à tour ces différentes façons d'initialiser un tableau. Pour initialiser un tableau vide, c'est très simple : il n'y a rien de particulier à faire. Vous avez juste à déclarer une variable de ce type vector, par exemple si vous voulez un tableau d'entiers, mais sans rien ajouter ici à la fin de la déclaration, sans mettre d'initialisation explicite. A ce moment-là, le tableau sera un tableau initialisé par le compilateur comme étant un tableau vide sans aucun élément. Si par contre vous voulez donner initialement des valeurs, par exemple mettre les valeurs 20, 35, 26, 38 et 22 déjà au départ dans votre tableau, ça n'en reste pas moins un tableau dynamique, il pourra évoluer. Mais si vous voulez au départ avoir ces valeurs-là, alors vous aller utiliser la syntaxe qui donne une liste ici explicite dans l'initialisation donc entre les deux accolades ouvrantes et fermantes, une liste de valeur. Bien sûr les valeurs que vous allez donner dans cette liste doivent être de même type que le type que vous avez spécifié comme type à votre tableau. On peut aussi écrire, bien que ce soit pas encore toléré par tout les compilateurs,

notes

résumé

2m 13s

Initialisation d'un tableau dynamique

EPFL

Une taille initiale peut être indiquée si nécessaire :

```
vector<type> identificateur(taille);
```

Exemple :

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

correspond à la déclaration d'un tableau initialement constitué de 5 entiers, tous nuls.

Attention ! Ce n'est pas la même chose que (à venir)

```
array<int, 5> tab;
```

car dans ce dernier cas on *ne* pourra *plus* changer la taille du tableau, alors que dans le cas de `vector` on le peut.
(et en plus le `array` n'est pas initialisé)

au lieu d'avoir une syntaxe ici d'initialisation comme on a toujours eu pour les autres variables, avoir une syntaxe ici qui initialise avec le signe =. Troisième cas d'initialisation, si je veux fixer une taille au départ, à ce moment-là la syntaxe va être la suivante : on va ici entre parenthèses non pas fixer une liste d'éléments, mais on va fixer un entier, une taille. Donc par exemple, si je veux au départ démarrer mon tableau dynamique avec cinq éléments, je vais déclarer un tableau dynamique d'entiers qui s'appelle "tab" et derrière ici, je vais dire qu'il démarrera initialement avec cinq éléments. En anticipant un peu avec une syntaxe qui va venir un peu plus tard, il faut faire très attention au fait qu'ici c'est bien un tableau dynamique qui contient initialement cinq éléments et que ça n'a rien à voir avec un tableau statique qui lui, contiendrait tout le temps cinq éléments

notes

résumé

3m 37s

		taille initiale connue *a priori* ?	
		non	oui
taille pouvant varier lors de l'utilisation du tableau ?	oui	`vector`	(`vector`)
	non	(`vector`)	`array` (C++11) tableaux « à la C »

et dont on ne pourrait pas faire varier la taille.

notes

résumé

4m 25s

Initialisation d'un tableau dynamique

EPFL

Avec taille initiale et initialisation des éléments à la *même* valeur :

```
vector<type> identificateur(taille, valeur);
```

On peut aussi initialiser un tableau dynamique à l'aide d'une copie d'un autre tableau dynamique :

```
vector<type> identificateur(reference);
```

où *reference* est une référence à un tableau de même type de base *type*.

Exemples :

```
vector<int> tab1(5, 1);
vector<int> tab2(tab1);
```

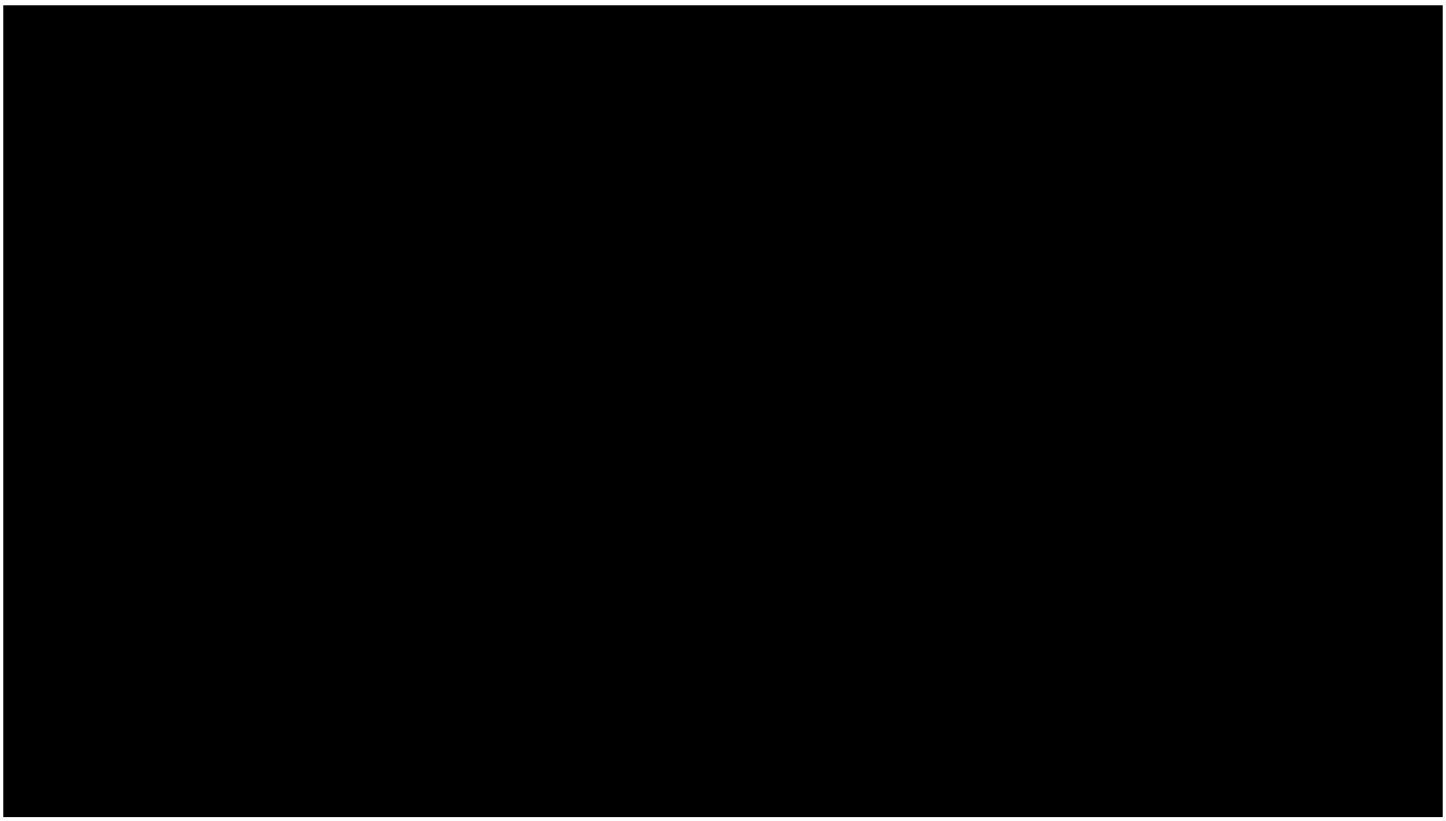
correspondent toutes deux à la déclaration d'un tableau d'entiers dont les 5 éléments de départ sont initialisés à la valeur 1.

Nous sommes bien ici en train de parler des tableaux dynamiques et nous ne sommes pas du tout encore en train de parler des tableaux dont la taille est fixée et qui ne peut plus évoluer. Donc un tableau dynamique, on peut lui donner une taille. Si on lui donne juste la taille, à ce moment-là le tableau est initialisé au départ avec des éléments qui sont tous nuls avec cette taille-là. Mais on peut aussi les initialiser avec différents éléments ayant la même valeur. Si par exemple je veux au départ démarrer mon tableau, qui reste dynamique, avec cinq éléments qui auront tous la valeur 1. A ce moment-là, je vais déclarer mon tableau comme un tableau dynamique d'entiers avec cinq éléments

notes

résumé

4m 28s



qui ont tous la valeur 1. Donc ici, ce deuxième élément, c'est un élément de valeur qui doit bien sûr être de même type que le type du tableau dynamique, le type des éléments du tableau dynamique. Donc ici je mets un 1 de type entier qui fait que les cinq éléments que j'ai initialisé, sont tous à la valeur 1. Le 5 ici, lui par contre sera toujours un entier, c'est une taille. Quelque soit le type du tableau, c'est toujours une taille que je donne. c'est toujours une taille que je donne.

notes

résumé

5m 13s