Support de cours

Cours:

Initiation à la programmation (en Java)

Vidéo:

Init-JAVA-04-5-tableaux-multi-pt2

Concepts (extraits des sous-titres générés automatiquement) :

Tableaux de valeurs entières. Exemple précédent. Éléments de la seconde dimension. Éléments de la première dimension. Élément d'indice. Ensemble des lignes du tableau. Conditions d'arrêt. Second indice. Tableau. Taille du tableau. Cas du tableau. Seconde condition d'arrêt. Nombre de ligne. Taille de l'entrée. Deuxième boucle.

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

vers la vidéo

EPFL

Tableaux à plusieurs dimensions
(Partie 2)

Initiation à la programmation (Java)

Jamila Sam, Vincent Lepetit et Jean-Cédric Chappelier

...

notes

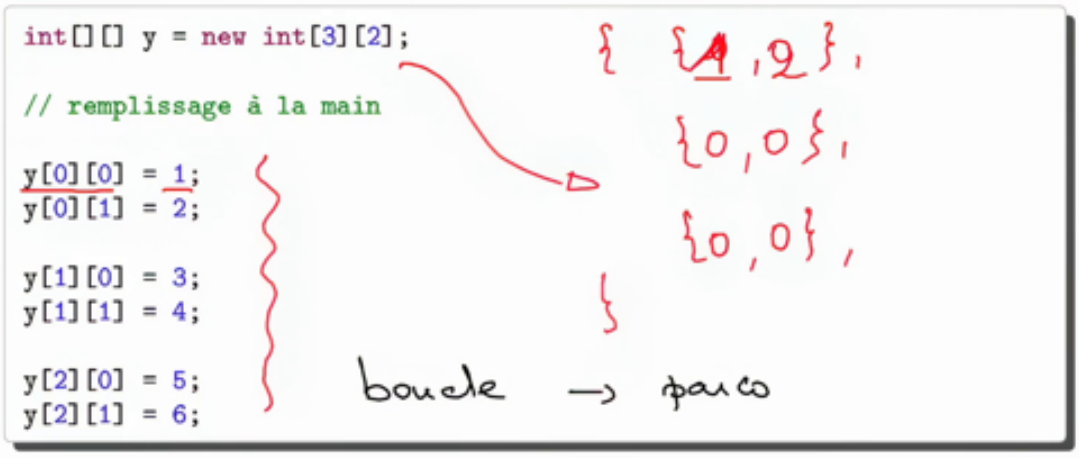
résumé

0m 0s

Déclaration-initialisation (2)

EPFL

Cas 2 : On ne connaît pas tous les éléments lors de la déclaration



Donc pour résumer la discussion qui a précédée, si on connaît à priori les éléments à mettre dans le tableau au moment où on le déclare, il est possible de l'initialiser par une tournure de cette nature. Ce que nous avons vu dans l'exemple précédent, c'est que les éléments de la première dimension sont chacun des tableaux. Ici en l'occurrence il s'agit de tableaux de valeurs entières. Donc concrètement, dans l'exemple que vous avez sous les yeux, vous avez un tableau de 3 tableaux : y[0], y[1], y[2] Le tableau y[0] est ce tableau. Le tableau y[2] est ce dernier tableau. Pour accéder aux éléments de la seconde dimension, j'ajoute un niveau de crochets, et bien évidemment la valeur à ce moment-là qui est accédée est un entier et non plus un tableau. Par exemple si je veux accéder à cet élément-là « y de [1][1] », je dois d'abord aller sur le tableau y[1], qui est celui-ci, ensuite j'accède à l'élément d'indice 1, qui est celui-ci, donc ici je tombe sur la valeur 4. Dans le cas le plus général, où on ne connaît pas à priori les valeurs à mettre dans le tableau, et bien on va se contenter comme dans le cas du tableau à 1D, de réserver les emplacements nécessaires au moment de l'initialisation. Donc ici concrètement, on va aboutir à la construction d'un tableau qui contient 3 lignes et chacune des lignes contient 2 éléments, donc c'est la structure à laquelle nous allons aboutir en mémoire. Et nous avons vu qu'il existait des valeurs par défaut pour l'initialisation de ces valeurs en JAVA. Donc les valeurs sont les mêmes que pour les tableaux à 1D, donc ici j'aurai des 0 partout puisqu'il s'agit d'entiers. Donc ensuite il faut procéder par un remplissage ad-hoc, donc à la main. Ici je le fais très explicitement en accédant case par case et en y mettant une valeur, donc

notes

résumé

0m 1s

Déclaration-initialisation (2)

EPFL

Cas 2 : On ne connaît pas tous les éléments lors de la déclaration

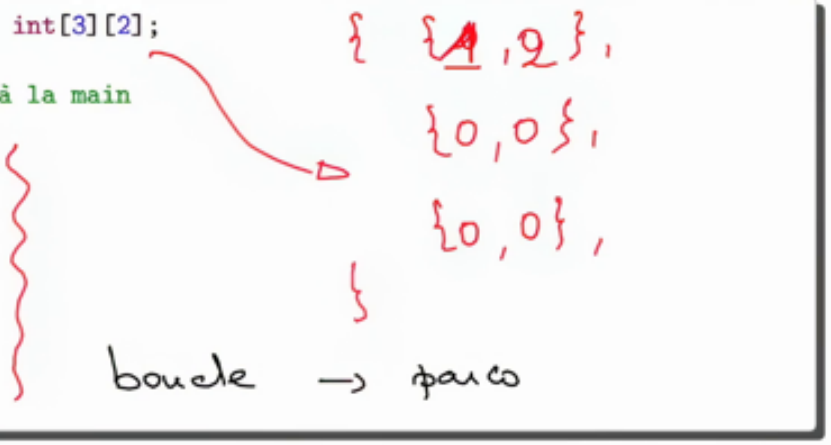
```
int[][] y = new int[3][2];

// remplissage à la main

y[0][0] = 1;
y[0][1] = 2;

y[1][0] = 3;
y[1][1] = 4;

y[2][0] = 5;
y[2][1] = 6;
```



ici je vais mettre 1 et je vais mettre 2 ici et ainsi de suite. Et on peut bien évidemment s'imaginer que c'est fastidieux de procéder de cette façon si l'on a affaire à des grands tableaux, ce qui nous fait naturellement penser à des boucles et à la notion de parcours.

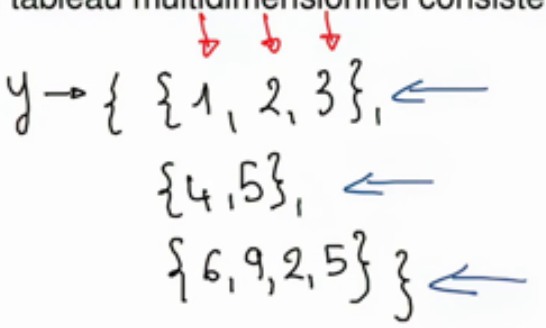
notes

résumé

Parcours

Le moyen le plus naturel de parcourir un tableau multidimensionnel consiste à utiliser des boucles `for` imbriquées :

1. 1re boucle : fait varier le 1er indice
2. 2e boucle : fait varier le 2e indice



Exemple :

lignes

```
for(int i = 0; i < y.length; ++i) {
 for(int j = 0; j < y[i].length; ++j) {
        System.out.println(y[i][j]);
}
}
```

colonne

Comment procéder si l'on veut parcourir un tableau à 2D, voire plus. Le moyen le plus naturel pour parcourir un tableau à 2D consiste à imbriquer deux boucles « for ». La première boucle « for » va permettre d'itérer sur l'ensemble des lignes du tableau, donc concrètement ici, nous permettra d'itérer sur chacune de ces lignes, donc de parcourir chacune de ces lignes en séquences. Le second indice, la deuxième boucle, va nous permettre d'itérer sur toutes les colonnes de chaque ligne. Donc ici, pour chacune des lignes, nous voulons itérer sur chacune des colonnes. C'est pourquoi les deux boucles sont en réalité imbriquées. C'est parce que pour chacun des « i », nous voulons parcourir tous les « j » possibles.

notes

résumé

2m 13s

Donc on notera ici en particulier les conditions d'arrêt. Donc le nombre de ligne s'arrête évidemment à « taille du tableau - 1 », ce qui correspond à cette notation. Et la seconde condition d'arrêt correspond à la taille de l'entrée « i » du tableau, ce qui correspond à cette notation-là. ce qui correspond à cette notation-là.

notes

résumé

3m 1s