Page d'Accueil
Préambule : le Codage
Introduction à l'algorithmique
1. Les Variables
2. Lecture et Ecriture
3. Les Tests
4. Encore de la Logique
5. Les Boucles
6. Les Tableaux
7. Techniques Rusées
8. Tableaux Multidimensionnels
Pourquoi plusieurs dimensions ?
Tableaux à 2 dimensions
Tableaux à n dimensions
9. Fonctions Prédéfinies
10. Fichiers
11. Procédures et Fonctions
12. Notions Complémentaires
Liens
Souvent Posées Questions

Rappel : ce cours d'algorithmique et de programmation
est enseigné à l'Université Paris 7,
dans la spécialité PISE du Master MECI (ancien DESS AIGES)
par Christophe Darmangeat

Partie 8
Tableaux Multidimensionnels

« *Le vrai problème n'est pas de savoir si les machines pensent, mais de savoir si les hommes pensent* » - B.F. Skinner

« *La question de savoir si un ordinateur peut penser n'est pas plus intéressante que celle de savoir si un sous-marin peut nager* » - Edgar W. Dijkstra

Ceci n'est pas un dérèglement de votre téléviseur. Nous contrôlons tout, nous savons tout, et les phénomènes paranormaux que vous constatez sont dus au fait que vous êtes passés dans… la quatrième dimension (musique angoissante : « tintintin… »).

Oui, enfin bon, avant d'attaquer la quatrième, on va déjà se coltiner la deuxième.

1. Pourquoi plusieurs dimensions ?

Une seule ne suffisait-elle pas déjà amplement à notre bonheur, me demanderez-vous ? Certes, répondrai-je, mais vous allez voir qu'avec deux (et davantage encore) c'est carrément le nirvana.

Prenons le cas de la modélisation d'un jeu de dames, et du déplacement des pions sur le damier. Je rappelle qu'un pion qui est sur une case blanche peut se déplacer (pour simplifier) sur les quatre cases blanches adjacentes.

Avec les outils que nous avons abordés jusque là, le plus simple serait évidemment de modéliser le damier sous la forme d'un tableau. Chaque case est un emplacement du tableau, qui contient par exemple 0 si elle est vide, et 1 s'il y a un pion. On attribue comme indices aux cases les numéros 1 à 8 pour la première ligne, 9 à 16 pour la deuxième ligne, et ainsi de suite jusqu'à 64.

Arrivés à ce stade, les fines mouches du genre de Cyprien L. m'écriront pour faire remarquer qu'un damier, cela possède 100 cases et non 64, et qu'entre les damiers et les échiquiers, je me suis joyeusement emmêlé les pédales. A ces fines mouches, je ferai une double réponse de prof :

1. C'était pour voir si vous suiviez.

2. Si le prof décide contre toute évidence que les damiers font 64 cases, c'est le prof qui a raison et l'évidence qui a tort. Rompez.

Reprenons. Un pion placé dans la case numéro i, autrement dit la valeur 1 de Cases[i], peut bouger vers les cases contiguës en diagonale. Cela va nous obliger à de petites acrobaties intellectuelles : la case située juste au-dessus de la case numéro i ayant comme indice i-8, les cases valables sont celles d'indice i-7 et i-9. De même, la case située juste en dessous ayant comme indice i+8, les cases valables sont celles d'indice i+7 et i+9.

Bien sûr, on peut fabriquer tout un programme comme cela, mais le moins qu'on puisse dire est que cela ne facilite pas la clarté de l'algorithme.

Il serait évidemment plus simple de modéliser un damier par… un damier !

Retour Haut de Page

2. Tableaux à deux dimensions

L'informatique nous offre la possibilité de déclarer des tableaux dans lesquels les valeurs ne sont pas repérées par une seule, mais par **deux coordonnées**.

Un tel tableau se déclare ainsi :

```
Tableau Cases[7, 7] en Numérique
```

Cela veut dire : réserve moi un espace de mémoire pour 8 x 8 entiers, et quand j'aurai besoin de l'une de ces valeurs, je les repèrerai par deux indices (comme à la bataille navale, ou Excel, la seule différence étant que pour les coordonnées, on n'utilise pas de lettres, juste des chiffres).

Pour notre problème de dames, les choses vont sérieusement s'éclaircir. La case qui contient le pion est dorénavant Cases[i, j]. Et les quatre cases disponibles sont Cases[i-1, j-1], Cases[i-1, j+1], Cases[i+1, j-1] et Cases[i+1, j+1].

REMARQUE ESSENTIELLE :
Il n'y a aucune différence qualitative entre un tableau à deux dimensions [i, j] et un tableau à une dimension [i * j]. De même que le jeu de dames qu'on vient d'évoquer, tout problème qui peut être modélisé d'une manière peut aussi être modélisé de l'autre. Simplement, l'une ou l'autre de ces techniques correspond plus spontanément à tel ou tel problème, et facilite donc (ou complique, si on a choisi la mauvaise option) l'écriture et la lisibilité de l'algorithme.

Une autre remarque : une question classique à propos des tableaux à deux dimensions est de savoir si le premier indice représente les lignes ou le deuxième les colonnes, ou l'inverse. Je ne répondrai pas à cette question non parce que j'ai décidé de bouder, mais **parce qu'elle n'a aucun sens**. « Lignes » et « Colonnes » sont des concepts graphiques, visuels, qui s'appliquent à des objets du monde réel ; les indices des tableaux ne sont que des coordonnées logiques, pointant sur des adresses de mémoire vive. Si cela ne vous convainc pas, pensez à un jeu de bataille navale classique : les lettres doivent-elles désigner les lignes et les chiffres les colonnes ? Aucune importance ! Chaque joueur peut même choisir une convention différente, aucune importance ! L'essentiel est qu'une fois une convention choisie, un joueur conserve la même tout au long de la partie, bien entendu.

Exercice 8.1
Exercice 8.2
Exercice 8.3
Exercice 8.4
Exercice 8.5
Exercice 8.6
Exercice 8.7

Retour Haut de Page

3. Tableaux à n dimensions

Si vous avez compris le principe des tableaux à deux dimensions, sur le fond, il n'y a aucun problème à passer au maniement de tableaux à trois, quatre, ou pourquoi pas neuf dimensions. C'est exactement la même chose. Si je déclare un tableau Titi[2, 4, 3, 3], il s'agit d'un espace mémoire contenant 3 x 5 x 4 x 4 = 240 valeurs. Chaque valeur y est repérée par quatre coordonnées.

Le principal obstacle au maniement systématique de ces tableaux à plus de trois dimensions est que le programmeur, quand il conçoit son algorithme, aime bien faire des petits gribouillis, des dessins immondes, imaginer les boucles dans sa tête, etc. Or, autant il est facile d'imaginer concrètement un tableau à une dimension, autant cela reste faisable pour deux dimensions, autant cela devient l'apanage d'une minorité privilégiée pour les tableaux à trois dimensions (je n'en fais malheureusement pas partie) et hors de portée de tout mortel au-delà. C'est comme ça, l'esprit humain a du mal à se représenter les choses dans l'espace, et crie grâce dès qu'il saute dans l'hyperespace (oui, c'est comme ça que ça s'appelle au delà de trois dimensions).

Donc, pour des raisons uniquement pratiques, les tableaux à plus de trois dimensions sont rarement utilisés par des programmeurs non matheux (car les matheux, de par leur formation, ont une fâcheuse propension à manier des espaces à n dimensions comme qui rigole, mais ce sont bien les seuls, et laissons les dans leur coin, c'est pas des gens comme nous).

Retour Haut de Page