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 6
Les Tableaux

« *Si on ment à un compilateur, il prendra sa revanche.* » - Henry Spencer.

Bonne nouvelle ! Je vous avais annoncé qu'il y a avait en tout et pour tout quatre structures logiques dans la programmation. Eh bien, ça y est, on les a toutes passées en revue.

Mauvaise nouvelle, il vous reste tout de même quelques petites choses à apprendre…

1. Utilité des tableaux

Imaginons que dans un programme, nous ayons besoin simultanément de 12 valeurs (par exemple, des notes pour calculer une moyenne). Evidemment, la seule solution dont nous disposons à l'heure actuelle consiste à déclarer douze variables, appelées par exemple Notea, Noteb, Notec, etc. Bien sûr, on peut opter pour une notation un peu simplifiée, par exemple N1, N2, N3, etc. Mais cela ne change pas fondamentalement notre problème, car arrivé au calcul, et après une succession de douze instructions « Lire » distinctes, cela donnera obligatoirement une atrocité du genre :

```
Moy ← (N1+N2+N3+N4+N5+N6+N7+N8+N9+N10+N11+N12)/12
```

Ouf ! C'est tout de même bigrement laborieux. Et pour un peu que nous soyons dans un programme de gestion avec quelques centaines ou quelques milliers de valeurs à traiter, alors là c'est le suicide direct.

Cerise sur le gâteau, si en plus on est dans une situation on l'on ne peut pas savoir d'avance combien il y aura de valeurs à traiter, là on est carrément cuits.

C'est pourquoi la programmation nous permet **de rassembler toutes ces variables en une seule**, au sein de laquelle chaque valeur sera désignée par un numéro. En bon français, cela donnerait donc quelque chose du genre « la note numéro 1 », « la note numéro 2 », « la note numéro 8 ». C'est largement plus pratique, vous vous en doutez.

Un ensemble de valeurs portant le même nom de variable et repérées par un nombre, s'appelle un tableau, ou encore une variable indicée.
Le nombre qui, au sein d'un tableau, sert à repérer chaque valeur s'appelle – ô surprise – l'indice.
Chaque fois que l'on doit désigner un élément du tableau, on fait figurer le nom du tableau, suivi de l'indice de l'élément, entre parenthèses.

Retour Haut de Page

2. Notation et utilisation algorithmique

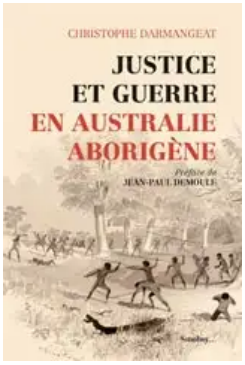
Dans notre exemple, nous créerons donc un tableau appelé Note. Chaque note individuelle (chaque emplacement du tableau Note) sera donc désignée Note[0], Note[1], etc. Eh oui, attention, les indices des tableaux, par convention, commencent généralement à 0, et non à 1.

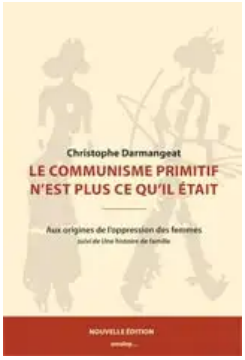
Un tableau doit être déclaré comme tel, en précisant le nombre et le type de valeurs qu'il contiendra (la déclaration des tableaux est susceptible de varier d'un langage à l'autre. Certains langages réclament le nombre d'éléments, d'autre le plus grand indice... C'est donc là aussi une affaire de conventions).

Très loin de l'informatique, pas tout près de l'économie :

Mon blog, la Hutte des Classes
À propos d'anthropologie sociale, de préhistoire et de marxisme.

Et mes livres...





En nous calquant sur les choix les plus fréquents dans les langages de programmations, nous déciderons ici arbitrairement et une bonne fois pour toutes que :

- les "cases" sont numérotées à partir de zéro, autrement dit que le plus petit indice est zéro.
- lors de la déclaration d'un tableau, on précise la plus grande valeur de l'indice (différente, donc, du nombre de cases du tableau, puisque si on veut 12 emplacements, le plus grand indice sera 11). Au début, ça déroute, mais vous verrez, avec le temps, on se fait à tout, même au pire.

```
Tableau Note[11] en Entier
```

On peut créer des tableaux contenant des variables de tous types : tableaux de numériques, bien sûr, mais aussi tableaux de caractères, tableaux de booléens, tableaux de tout ce qui existe dans un langage donné comme type de variables. Par contre, hormis dans quelques rares langages, on ne peut pas faire un mixage de types différents de valeurs au sein d'un même tableau.

L'énorme avantage des tableaux, c'est qu'on va pouvoir les traiter en faisant des boucles. Par exemple, pour effectuer notre calcul de moyenne, cela donnera par exemple :

```
Tableau Note[11] en Numérique
Variables Moy, Som en Numérique
Début
Pour i ← 0 à 11
  Ecrire "Entrez la note n°", i
  Lire Note[i]
i Suivant
Som ← 0
Pour i ← 0 à 11
  Som ← Som + Note[i]
i Suivant
Moy ← Som / 12
Fin
```

NB : On a fait deux boucles successives pour plus de lisibilité, mais on aurait tout aussi bien pu n'en écrire qu'une seule dans laquelle on aurait tout fait d'un seul coup.

Remarque générale : l'indice qui sert à désigner les éléments d'un tableau peut être exprimé directement comme un nombre en clair, mais il peut être aussi une variable, ou une expression calculée.

Dans un tableau, la valeur d'un indice doit toujours :

- **être égale au moins à 0** (dans quelques rares langages, le premier élément d'un tableau porte l'indice 1). Mais comme je l'ai déjà écrit plus haut, nous avons choisi ici de commencer la numérotation des indices à zéro, comme c'est le cas en langage C et en Visual Basic. Donc attention, Truc[6] est le septième élément du tableau Truc !
- **être un nombre entier** Quel que soit le langage, l'élément Truc[3,1416] n'existe jamais.
- **être inférieure ou égale au nombre d'éléments du tableau** (moins 1, si l'on commence la numérotation à zéro). Si le tableau Bidule a été déclaré comme ayant 25 éléments, la présence dans une ligne, sous une forme ou sous une autre, de Bidule[32] déclenchera automatiquement une erreur.

Je le re-re-répète, si l'on est dans un langage où les indices commencent à zéro, il faut en tenir compte à la déclaration :

```
Tableau Note[13] en Numérique
```

...créera un tableau de 14 éléments, le plus petit indice étant 0 et le plus grand 13.

> **LE GAG DE LA JOURNEE**
> Il consiste à confondre, dans sa tête et / ou dans un algorithme, l'**indice** d'un élément d'un tableau avec le **contenu** de cet élément. La troisième maison de la rue n'a pas forcément trois habitants, et la vingtième vingt habitants. En notation algorithmique, il n'y a aucun rapport entre i et truc[i].

Holà, Tavernier, prépare la cervoise !

Retour Haut de Page

3. Tableaux dynamiques

Il arrive fréquemment que l'on ne connaisse pas à l'avance le nombre d'éléments que devra comporter un tableau. Bien sûr, une solution consisterait à déclarer un tableau gigantesque (10 000 éléments, pourquoi pas, au diable les varices) pour être sûr que « ça rentre ». Mais d'une part, on n'en sera jamais parfaitement sûr, d'autre part, en raison de l'immensité de la place mémoire réservée – et la plupart du temps non utilisée, c'est un gâchis préjudiciable à la rapidité, voire à la viabilité, de notre algorithme.

Aussi, pour parer à ce genre de situation, a-t-on la possibilité de déclarer le tableau sans préciser au départ son nombre d'éléments. Ce n'est que dans un second temps, au cours du programme, que l'on va fixer ce nombre via une instruction de redimensionnement : **Redim**.

Notez que **tant qu'on n'a pas précisé le nombre d'éléments d'un tableau, d'une manière ou d'une autre, ce tableau est inutilisable**.

Exemple : on veut faire saisir des notes pour un calcul de moyenne, mais on ne sait pas combien il y aura de notes à saisir. Le début de l'algorithme sera quelque chose du genre :

```
Tableau Notes[] en Numérique
Variable nb en Numérique
Début
Ecrire "Combien y a-t-il de notes à saisir ?"
Lire nb
Redim Notes[nb-1]
…
```

Cette technique n'a rien de sorcier, mais elle fait partie de l'arsenal de base de la programmation en gestion.

Retour Haut de Page