Fiche 1 : Les tableaux

>>> Création, lecture et enregistrement de données

Description :

Le but de cet article n'est pas de faire un cours sur les tableaux mais simplement de donner un bref rappel sur leur création et leur manipulation.
Attention : il est nécessaire d'avoir déjà acquis les bases de la programmation en C afin d'apprécier au mieux ces exemples.

Fiche 1 : Les tableaux

>>> Création, lecture et enregistrement de données

Sommaire :

I) Les tableaux simples
1) Définir un tableau
2) Lire les cases d'un tableau
II) Les tableaux à 2 dimensions
1) Définir un tableau bidimensionnel
2) Initialiser un tableau bidimensionnel
3) Lecture du tableau bidimensionnel
III) Conclusion

I) Les tableaux simples

1) Définir un tableau

Pour définir un tableau, il suffit de le déclarer en indiquant le nombre de cases que l'on souhaite. Ainsi, un tableau de 10 cases sera créé comme suit :

```
int nomDuTableau[10];
```

2) Lire les cases d'un tableau

Là encore il n'y a rien de compliqué, il suffit d'utiliser la forme suivante :

```
nomDuTableau[numéroDeLigne];
```

(avec *numéroDeLigne* étant égale à la case que vous souhaitez voir dans un tableau "vertical").

/ ! N'oubliez pas qu'il faut compter les cases d'un tableau à partir de zéro !

Dans le cas où on n'écrit que le nom du tableau, on se retrouve face à un pointeur et ce dernier, lors de son utilisation, nous renverrait l'adresse de la première case.

De ce fait, on peut aussi utiliser les pointeurs pour lire un tableau grâce à l'indication * devant le nom de ce dernier. En effet, le code

```
printf("%d", *nomDuTableau);
```

permet, par exemple, d'afficher la première valeur de notre tableau. Pour afficher la valeur d'une des cases suivantes, il faut utiliser le pointeur et ajouter le numéro de la case. Ainsi, pour afficher la dernière case du tableau (la neuvième donc, puisqu'on part de zéro), on aura le code suivant :

```
printf("%d", *(nomDuTableau + 9)
```

II) Les tableaux à 2 dimensions

1) Définir un tableau bidimensionnel

La définition d'un tableau à deux dimensions se fait en indiquant le nombre de lignes et le nombre de colonnes :

```
int nomDuTableau[nombreDeLignes][nombreDeColonne];
```

2) Initialiser un tableau bidimensionnel

L'initialisation d'un tableau à deux dimensions peut se faire de plusieurs manières :

- en indiquant la liste des composants du tableau entre accolades (définissant chaque ligne) lors de la déclaration :

```
int nomDuTableau[2][3] = {{29, 56, 22} {53, 72, 35}};
```

Ceci permet d'obtenir le tableau suivant :

29	56	22
53	72	35

- en utilisant les coordonnées de la case que l'on souhaite remplir :

```
nomDuTableau[numeroLigne][numéroColonne];
```

Par exemple, pour remplacer le nombre 53 dans le tableau ci-dessus, on utilisera l'instruction suivante :

```
nomDuTableau[2][1] = 50;
```

On aura alors le résultat suivant :

29	56	22
50	72	35

3) Lecture du tableau bidimensionnel

La lecture d'un tableau à deux dimensions est un peu moins simple à pratiquer. En effet, pour se faire, nous aurons besoin non pas d'une, mais de deux boucles : la première permettant de créer ou de changer de ligne à initialiser, la seconde permettant de créer ou initialiser une colonne.

Voici le code de base qui réalise cette demande (à partir du tableau créé précédemment) :

```
int i =0;
int j= 0;

for (i<=1) // tant que i inférieur ou égal à 1
{
for (j<=2) // tant que j inférieur ou égal à 2
{
printf("| %d |", nomDuTableau[i][j]); // affichage du contenu de la case au coordonnées [i] et [j]
j++; // ajouter 1 à j pour passer à la colonne suivante
}
printf("-----------------------------"); // affichage pour la mise en forme
printf("\n"); // passage à la ligne suivante sur la console
i++; // ajouter 1 à i pour passer à la ligne suivante dans le tableau
}
```

Dans l'exemple ci-dessus 1 et 2 correspondent respectivement au nombre de lignes-1 et au nombre de colonnes-1 puisque dans les deux cas on partira de zéro pour les compter.

III) Conclusion

Dans cet article, vous avez pu revoir succinctement les tableaux, leurs initialisations et les manières de les lire ainsi que d'y inscrire des données.