Tableaux

-Lorsqu’un tableau est créé, il prend un espace contigu en mémoire : les cases sont les unes a la suite des autres.

-Toutes les cases d’un tableau sont du même type. Ainsi, un tableau de *int* contiendra uniquement des *int*, et pas autre chose.

/!\ un tableau commence a l’indice 0. Notre tableau de 4 *int* a donc les indices 0, 1, 2 et 3. Il n’y a pas d’indice 4 dans un tableau de 4 cases.

Rapport entre les tableaux et les pointeurs :

Si les valeurs:

int tableau[4];

tableau[0] = 10;

tableau[1] = 23;

tableau[2] = 505;

tableau[3] = 8;

Si on écrit juste *tableau*, on obtient un pointeur. C’est un pointeur sur la première case du tableau.

int tableau[4];

printf("%d", tableau);

Résultat, on voit l’adresse où se trouve le tableau : 1600 (exemple).

En revanche, si on indique l’indice de la case du tableau entre crochets, on obtient la valeur :

int tableau[4];

printf("%d", tableau[0]);

Résultat : 10.

De même pour les autres indices. Comme *tableau* est un pointeur, on peut utiliser le symbole \* pour connaître la première valeur :

int tableau[4];

printf("%d", \*tableau);

Résultat : 10.

Il est aussi possible d’obtenir la valeur de la seconde case avec *\*(tableau + 1)*  (adresse du tableau + 1).

Les deux lignes suivantes sont donc identiques :

tableau[1] // Renvoie la valeur de la seconde case (la première case étant 0)

\*(tableau + 1) // Identique : renvoie la valeur contenue dans la seconde case

Tableaux à taille dynamique

Parcourir le tableau pour mettre 0 a chaque case :

for (i = 0 ; i < 4 ; i++)

{

tableau[i] = 0;

}

Autre façon d’initialiser :

Elle consiste à écrire *tableau[4] = {valeur1, valeur2, valeur3, valeur4}.*

Résultat : 0 0 0 0

/!\ Vous pouvez aussi définir les valeurs des premières cases de tableau, toutes celles qui ne sont pas renseignées auront automatiquement la valeur 0.

Passage de tableaux a une fonction

Ex // pour afficher tout le contenu d’un tableau, il faut envoyer deux informations a la fonction :

Le tableau (l’adresse du tableau) et aussi (surtout) sa taille.

La fonction doit être capable d’initialiser un tableau de n’importe quelle taille. (of course)

*tableau* peut être considéré comme un pointeur. On peut donc l'envoyer à la fonction comme on l'aurait fait avec un vulgaire pointeur :

// Prototype de la fonction d'affichage

void affiche(int \*tableau, int tailleTableau);

int main(int argc, char \*argv[])

{

int tableau[4] = {10, 15, 3};

// On affiche le contenu du tableau

affiche(tableau, 4);

return 0;

}

void affiche(int \*tableau, int tailleTableau)

{

int i;

for (i = 0 ; i < tailleTableau ; i++)

{

printf("%d\n", tableau[i]);

}

}

Résultat : 10.

/!\ Notez qu'il existe une autre façon d'indiquer que la fonction reçoit un tableau. Plutôt que d'indiquer que la fonction attend un *int \*tableau*, mettez ceci :

void affiche(int tableau[], int tailleTableau)

Cela revient exactement au même, mais la présence des crochets permet au programmeur de bien voir que c'est un tableau que la fonction prend, et non un simple pointeur. Cela permet d'éviter des confusions.