TD 5
Deug S2
Tableaux

Dans cette séance de travaux dirigés, nous introduisons les structures de données "tableaux". Nous allons travailler sur des tableaux de taille N (N est une constante définie dans le programme).

Rappels

- **Définition de constante symbolique**
  `#define N 5` est une instruction du précompilateur. Toute apparition de N hors d'une chaîne de caractères sera remplacée par `5`.

- **Tableau**
  Un tableau en C est une suite indicée d'éléments du même type. Les indices commencent impérativement à l'indice 0. Une variable tableau est déclarée par :
  `type_des_éléments Nom[taille] ;`

1. **Accès aux cases d'un tableau**
   Écrire un programme qui déclare un tableau de N entiers et met la valeur -1 dans toutes les cases. Quel est l'intérêt d'utiliser une constante symbolique pour définir la taille du tableau ?
   ATTENTION C'est au programmeur de vérifier que les indices utilisés ne dépassent pas la taille du tableau.
   Tableaux et fonctions
   L'indication qu'un paramètre est un tableau se fait uniquement par les `[]` : par exemple `void affiche(int suite[], ,)`. Il peut être nécessaire de transmettre à la fonction la taille du tableau.
   L'affectation d'une case d'un tableau transmis en paramètre d'une fonction change définitivement la valeur de la case. Le passage du tableau en paramètre est toujours un passage par adresse. En fait le C identifie le tableau et l'adresse de son premier élément.

2. **Lecture, affichage d'un tableau**
   - Écrire une fonction `Affichage` prenant en argument un tableau `T` de N entiers et qui effectue l'affichage de ce tableau.
   - Écrire une fonction `Lecture` qui prend un tableau `T` de N entiers en argument et qui remplit le tableau `T` avec N entiers lus au clavier.

3. **Calcul d'une moyenne**
   - Écrire une fonction `Moyenne` prenant en argument un tableau `T` de N réels et qui calcule la moyenne des éléments de ce tableau.

- En déduire une fonction `SuperieurMoy` qui compte le nombre d'éléments supérieurs à la moyenne des éléments du tableau `T` donné en argument.

4. **Modification dans un tableau**
   Écrire une fonction `Modif` prenant en argument un tableau `T` et un entier `i` et qui remplace la valeur `T[i]` par `T[i]+2` (que faut-il vérifier ?). On effectuera un affichage du tableau `T` après l'appel à `Modif`.

5. **Tableau à deux dimensions**
   Définir un tableau carré à deux dimensions ($N \times N$) de caractères. Saisir les coordonnées d'une case de ce tableau (indices de ligne et de colonne), et placer `*` dans cette case. Placer des `+` dans les diagonales du tableau passant par ce point, et des blancs partout ailleurs. Faire afficher le tableau ligne par ligne.

6. **Triangle de Pascal**
   Les coefficients binomiaux formant le triangle de Pascal sont liés par les relations de récurrence suivantes :

$$C_n^p = \begin{cases} 1 & \text{si } n = p \text{ ou } p = 0, \\ 0 & \text{si } p > n, \\ C_{n-1}^p + C_{n-1}^{p-1} & \text{si } p < n. \end{cases}$$

   Nous voulons afficher le triangle de Pascal jusqu'à l'ordre N. Nous proposons de résoudre ce problème en utilisant trois structures différentes :

   - un tableau à deux dimensions de taille $N \times N$ (la case `t[i][j]` contient C_i^j) ;
   - deux tableaux de taille N ;
   - un tableau de taille N.

   Dans chaque cas, décrire la méthode utilisée et écrire la fonction `AffichePascal` qui réalisent l'affichage du triangle de Pascal.