1 TP6 – TABLEAUX A DEUX DIMENSIONS

Le but de ce TP est de se familiariser avec la notion de tableaux à deux dimensions et de renforcer les notions sur les sous-programmes.

Ces exercices ont pour but de travailler sur les manipulations d'indices et de valeurs de tableaux à deux dimensions ou matrices.

1.2 Initialisation et affichage d'une matrice

Il s'agit de réaliser, dans un premier temps, un programme qui permet d'initialiser une matrice et d'afficher les valeurs de la matrice. Le programme principal doit présenter un menu sous la forme suivante :

```
Voulez-vous:
(1) initialiser la matrice
(2) afficher la matrice
(3) quitter
```

Préparation

1. *Préparer la déclaration des constantes correspondant au nombre maximum de lignes et de colonnes du tableau à deux dimensions. Par exemple : LIG_MAX = 10 et COL_MAX = 15.*

Préparer la déclaration du type de la matrice correspondant au tableau à deux dimensions ayant LIG_MAX lignes et COL_MAX colonnes : t_matrice.

***NB.** Dans la suite du TP, la taille effective utilisée pour les matrices est LIG_MAX et COL_MAX. Ainsi, elle ne sera pas passée en paramètres dans les sous-programmes.*

2. *Faire la vue externe puis écrire l'algorithme du sous-programme initialiserMatrice() qui initialise à 0 toutes les cases d'une matrice qui sera renvoyée au programme principal.*
3. *Faire la vue externe puis écrire l'algorithme du sous-programme afficherMatrice() qui affiche la série de mesures sous forme d'un tableau à deux dimensions. Le sous-programme reçoit en entrée le tableau à afficher.*
4. *Faire la vue externe puis écrire l'algorithme du programme principal dont le rôle est de présenter le menu donné ci-dessus et d'appeler les deux sous-programmes précédents.*

Reprendre le travail effectué lors du TP5 en utilisant une variable de type entier pour la sélection des différents choix disponibles.

Réalisation en séance

1. *Ouvrir, dans **Code::Blocks**, un nouveau projet TP6 ainsi que le fichier modele.cpp qui doit être renommé en manipulationMatrice.cpp.*
2. *Coder le programme principal puis tester son fonctionnement en remplaçant les appels de sous-programmes par l'affichage du nom du sous-programme concerné.*
3. *Compléter le programme précédent en y intégrant les deux sous-programmes lireMatrice() et afficherMatrice().*

1.3 Manipulation des éléments de la matrice

Cette partie consiste à manipuler les indices afin de travailler sur une partie restreinte de la matrice

Relativement à la partie précédente, le menu du programme principal doit être complété à chaque ajout de traitement. En version finale, on doit disposer des choix suivants :

Voulez-vous:
(1) initialiser la matrice
(2) afficher la matrice
(3) lire une valeur
(4) lire les valeurs d'une ligne à partir d'une case de départ
(5) remplir la matrice avec des valeurs aléatoires
(6) effacer les valeurs à partir d'une case de départ et d'une direction
(7) quitter

Préparation

1. *Faire la vue externe puis écrire l'algorithme du sous-programme lireCoordonnees() qui demande à l'utilisateur les coordonnées d'une case et les renvoie au sous-programme appelant. Les numéros de ligne et de colonne saisies au clavier doivent être valides.*

Ce sous-programme est utilisé par les sous-programmes suivants si besoin.

2. *Faire la vue externe puis écrire l'algorithme du sous-programme lireVal() dont le but est d'enregistrer une valeur fournie par l'utilisateur dans la case qu'il a lui-même choisie. Ce sous-programme doit recevoir puis renvoyer la matrice.*

3. *Faire la vue externe puis écrire l'algorithme du sous-programme lireLigne() dont le but est de remplir la ligne à partir d'une case donnée, avec des valeurs fournies par l'utilisateur.*

L'utilisateur entre les coordonnées de la case initiale puis les valeurs des cases de la ligne choisie. Ce sous-programme doit recevoir puis renvoyer la matrice.

4. *Faire la vue externe puis écrire l'algorithme du sous-programme* initialiserAleatoirement() *dont le but est de remplir la matrice avec des valeurs générées aléatoirement.*

Ce sous-programme utilise la fonction rand() qui génère un nombre entier aléatoire compris entre 0 et une constante RAND_MAX. Cette fonction doit être utilisée conjointement avec la fonction srand(). Lire le document tirage_aleatoire.pdf disponible sur le serveur Moodle de l'IUT.

Réalisation en séance

1. *Compléter le programme principal de la partie précédente un y intégrant un par un les trois premiers sous-programmes. Vous devez tester et valider le fonctionnement du premier avant de passer au suivant.*
2. *Tester le sous-programme initialiserAleatoirement() sans faire appel à la fonction srand(). Répeter plusieurs fois l'appel à ce sous-programme puis l'affichage de la matrice. Que constatez-vous ?*
3. *Rajouter en début de programme principal, après les déclarations locales, l'appel au sous-programme srand() en utilisant la syntaxe présentée dans le document tirage_aleatoire.pdf. Répéter à nouveau plusieurs fois l'appel au sous-programme initialiserAleatoirement() puis l'affichage de la matrice. Que constatez-vous ? Conclure.*

1.4 Bonus : Effacement d'éléments de la matrice

Cette partie consiste à effacer les éléments en partant d'une case de départ et d'une direction choisie par l'utilisateur. Elle implique d'ajouter un nouveau menu après le choix de l'option 6 du menu principal donné dans la partie précédente.

```
Case de départ.
Donner la ligne (0..10) et la colonne (0..15) :
Donner la direction d'effacement :
(1) haut
(2) bas
(3) gauche
(4) droite
```

Préparation

1. *Étudier l'évolution des indices d'un tableau à deux dimensions à partir des coordonnées d'une case en considérant les 4 directions possibles. Pour cela donner la relation entre l'indice de la case présente et celui de la case suivante.*
2. *Écrire l'algorithme du sous-programme effacerDirection() dont le but est d'effacer les éléments du tableau à partir d'une case donnée, et en considérant la direction choisie par l'utilisateur.*

Comme les autres sous-programmes, effacerDirection() doit recevoir puis renvoyer la matrice.

Réalisation en séance

1. *Compléter le programme principal de la partie précédente un y intégrant ce sous-programme que vous testerez et validerez.*