

TP 9 Les tableaux à 2 dimensions
--

Exercice 1

Question 1/ Définissez un type tableau à deux dimensions `t_tab2dim`, le premier indice variant de 0 à 9, le deuxième de 0 à 19 (ces valeurs ne sont que des exemples, n'oubliez pas la notion de constante).

Question 2/ Écrivez une procédure `initialiser(sortie t_tab2dim : tab)` qui initialise chaque case du tableau `tab` avec la somme de l'indice de ligne et de l'indice de colonne.

Question 3/ Écrivez une procédure `afficher(t_tab2dim : tab)` qui affiche à l'écran le tableau `tab` sous forme de matrice.

Question 4/ Écrivez une fonction booléenne :
`fonction existe(t_tab2dim : tab, entier : valeur) délivre booléen`
qui retourne `true` si la valeur `valeur` est présente dans le tableau `tab` et `false` sinon.

Question 5/ Dans un `main()`, déclarez une variable `leTablo` de type `t_tab2dim`, initialisez-la, affichez son contenu et testez la fonction `existe`.

Exercice 2 : les sports

On veut calculer l'évolution du nombre de pratiquants dans différents sports en extrapolant l'évolution actuelle pour les années à venir.

On a constaté l'évolution suivante pour le cyclisme, le foot, la voile, et pour les inactifs .

Pour le cyclisme :	90 % continuent, 3 % passent au foot, 2 % passent à la voile, 5 % arrêtent le sport
Pour le foot :	6 % passent au cyclisme, 80 % continuent, 3% passent à la voile, 11 % arrêtent le sport
Pour la voile :	8 % passent au cyclisme, 2 % passent au foot, 75 % continuent, 15 % arrêtent le sport
Pour les inactifs :	5 % passent au cyclisme, 3 % passent au foot, 4 % passent à la voile, 88 % restent inactifs.

Question 1/

Le nombre de pratiquants actuels est de 35 pour le cyclisme, 25 pour le foot, 20 pour la voile et 20 pour les inactifs. Calculez le nombre de pratiquants dans chacun des sports au bout d'un an.

Question 2/

Le calcul précédent peut se mettre sous la forme du produit d'un vecteur par une matrice. Écrivez le vecteur de départ, la matrice, et le vecteur résultant du produit du vecteur de départ par la matrice.

Question 3/

a) Définissez un type `t_vecteur` et un type `matrice`, et initialisez directement dans le programme principal (sans lecture au clavier) le vecteur initial `vInitial` (des nombres de pratiquants), et la matrice d'évolution `mEvolution`.

On envisage un nombre maximum de sports possibles (par exemple 10), sachant que dans notre exercice, nous raisonnons sur 4 sports.

b) Écrivez la procédure `void afficherVecteur(t_vecteur vect, int nb)` qui affiche à l'écran les `nb` valeurs réellement contenues dans le vecteur `vect`.

c) Écrivez la procédure `void afficherMatrice(t_matrice mEvolution, int nb)` qui affiche à l'écran les `nb x nb` valeurs réellement contenues dans `mEvolution`.

d) Écrivez la procédure

`void produitVectMat(t_vecteur vInitial, t_matrice mEvolution, t_vecteur vResultat, int nb)` qui effectue le produit du vecteur `vInitial` par la matrice `mEvolution` et fournit le résultat dans le vecteur `vResultat`.

e) Écrivez la procédure `copiervecteur(t_vecteur vInitial, t_vecteur vResultat, int nb)` qui copie le vecteur `vInitial` dans le vecteur `vResultat`.

f) Complétez le programme principal pour calculer l'évolution en nombre de pratiquants des différents sports pour les 30 années à venir si l'évolution constatée se poursuit.

Exercices complémentaires

Exercice 3 : statistiques cinéma

À la sortie du cinéma de Lannion, on a interrogé les spectateurs pour connaître le nombre moyen de films vus en fonction de la catégorie socioprofessionnelle et de la tranche d'âge du spectateur.

Il y a `NbCat` catégories socioprofessionnelles (par exemple `NbCat=6`) et `NbTr` tranches d'âge (par exemple `NbTr=5`), numérotées respectivement de 1 à `NbCat` et de 1 à `NbTr`.

Pour chaque personne interrogée, il faut saisir sa catégorie socioprofessionnelle, sa tranche d'âge, et le nombre de films qu'elle a vus.

Question

Écrivez un programme, judicieusement «découpé» en procédures, qui permet :




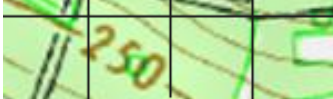
- de saisir les données de chaque personne interrogée. Une étoile "*" à la place du nom signale la fin des données.
- d'obtenir le nombre de spectateurs situés dans la tranche d'âge '*tranche*' et de catégorie socioprofessionnelle '*categorie*' ceci pour toute '*categorie*' et toute '*tranche*'.
- d'obtenir le nombre moyen de films vus par un spectateur situé dans la tranche d'âge '*tranche*' et de catégorie socioprofessionnelle '*categorie*' ceci pour toute '*categorie*' et toute '*tranche*'.

N.B. Les données venant de l'extérieur (*nom, catégorie, tranche, nombre de films vus*) ne sont pas stockées dans un tableau, mais traitées au fur et à mesure. Les saisies doivent être contrôlées (ex : vérifier que la tranche d'âge saisie est comprise entre 1 et `NbTr`)

Exercice 4

On considère le quadrillage d'une carte géographique comme un tableau à deux dimensions où chaque case contient l'altitude moyenne (nombre entier de mètres) du carré correspondant sur la carte. Pour simplifier, nous allons considérer des cartes carrées de taille TAILLE, avec TAILLE=4.

Exemple :

	0	1	2	3		0	1	2	3
0					0	2 7 5	2 7 5	2 8 0	2 8 5
1					1	2 6 5	2 7 0	2 7 0	2 7 5
2					2	2 5 5	2 6 0	2 6 5	2 7 0
3					3	2 4 5	2 5 0	2 5 5	2 6 0

Question 1/

Définir un type "carte".

Question 2/

Écrire un programme principal avec un menu qui, par l'appel de procédures ou fonctions, permettra à l'utilisateur de :

- 1) Initialiser une carte (toutes les valeurs à 0).
- 2) Saisir les valeurs d'une carte.
- 3) Afficher la somme de toutes les altitudes d'une carte (somme des valeurs du tableau).
- 4) Donner l'altitude la plus basse d'une carte (la valeur minimale des cases du tableau)
- 5) Rechercher si une altitude, saisie au clavier, est l'une des valeurs de la carte.
- 6) Afficher la moyenne des altitudes de la carte.
- 7) Quitter le programme.

Question 3/

Écrire un autre programme principal (vous mettrez le précédent en commentaires) qui à l'aide des fonctions et procédures précédentes :

- Initialisera deux cartes.
- Demandra la saisie au clavier des valeurs pour ces deux cartes.
- Affichera si oui ou non les deux cartes sont identiques. Vous écrirez une fonction **identique** qui prend en paramètres les deux cartes et qui retourne *true* si les deux cartes sont identiques et *false* sinon.
- Précisera quelle carte est la plus haute (c'est-à-dire quelle carte a la moyenne la plus grande) à l'aide d'une fonction **compare** qui prend deux cartes en paramètre et qui retourne 1 si la première carte est plus haute, -1 si c'est la seconde et 0 en cas d'égalité.