

Travaux Dirigés 3 Tableaux unidimensionnels

Notes de cours et consigne :

- Un tableau unidimensionnel est déclaré par :
`var <identificateur>: tableau (<taille>) <type de base> ;`
 où **<taille>** est une constante entière positive indiquant le nombre d'éléments du tableau.
- Un élément est désigné par `<identificateur>[<indice>]` ou `<identificateur>(<indice>)`.
- <indice>** est une expression qui peut prendre une valeur entière positive entre 1 et **<taille>** (et non pas entre 0 et **<taille>-1** comme en C).

Exercice 1 : Réaliser un algorithme en suivant les étapes suivantes :

- Ecrire un algorithme qui lit un tableau de 9 notes puis calcule et affiche leur moyenne.
- Modifier l'algorithme afin de saisir un tableau `tabCoef` contenant les coefficients respectifs des 9 modules. Ensuite donner la nouvelle moyenne en tenant compte de ces coefficients.

Exercice 2 : Soit une liste de N nombres entiers contenus dans un tableau. Ecrire l'algorithme qui détermine le plus petit parmi ces nombres.

Traduire en C.

Exercice 3 : Ecrire un algorithme qui permet d'inverser l'ordre des éléments d'un tableau.

Rem : résoudre cet exercice avec deux tableaux puis avec un seul (utiliser pour ce 2^{ème} cas la boucle tantque puis pour, avec deux indices puis avec un seul).

Traduire en C.

Exercice 4 : Soit un tableau `vect` composé de n nombres entiers. On veut écrire un algorithme qui permet de scinder `vect` en deux tableaux `vectPos` et `vectNeg` contenant respectivement les nombres positifs et les nombres négatifs.

Exemple :

Pour le tableau `vect` suivant :

-1	6	3	-5	-2	0
----	---	---	----	----	---

votre algorithme produira les tableaux :

6	3	Pour <code>vectPos</code>
---	---	---------------------------

et

-1	-5	-2	-1	Pour <code>vectNeg</code>
----	----	----	----	---------------------------

Un programmeur novice a proposé le traitement suivant (après déclaration et lecture de `vect`):

```
Pour i ← 1 à n faire
    si vect(i)>0 alors vectPos(i) ← vect(i) ;
    sinon vectNeg(i) ← vect(i) ;
fPour ;
```

1. Que va réaliser un tel traitement pour l'exemple précédent (i.e. `vect=(-1,6,3,-5,-2,0)`) ? Est ce le résultat escompté ?

2. Écrire un algorithme qui réalise la scission de façon juste.

Exercice 5 : La suite (U_n) est définie par:

$$\begin{cases} U_1 = 2023 \\ U_i = \frac{1000}{U_{i-1}} + \frac{U_{i-1}}{1000} & \text{si } i > 1 \end{cases}$$

1. En utilisant les tableaux, écrire un algorithme qui demande à l'utilisateur un nombre entier n ($1 \leq n \leq 100$), puis calcule et affiche la valeur du n^{ème} terme de la suite (U_n) .

2. Comparer votre solution a celle n'utilisant pas de tableaux (exercice 10 TD2)

Traduire en C.

Exercice 6 : On se propose de vérifier si une valeur VAL donnée est un élément d'un tableau T de nombres entiers.

- Écrire un algorithme de recherche séquentielle de VAL dans T (Afficher l'index de VAL dans T).
- Supposons que le tableau initial T soit trié par

ordre croissant. Utiliser une approche dichotomique pour la recherche de VAL dans T.

Exercice 7 : Soit VECT un tableau d'entiers. Ecrire un algorithme qui insère, dans ce tableau une valeur VAL à la $k^{\text{ième}}$ position.

Exercice 8 : Etant donné deux tableaux V1 et

V2, triés dans l'ordre croissant :

a. Ecrire un algorithme qui fusionne ces deux vecteurs en un vecteur V3 trié dans le même ordre.

b. Modifier l'algorithme afin de supprimer les doublons éventuels. Traduire en C.

Exercices supplémentaires

Exercice 9 : Que produit l'algorithme suivant ?

```
Var nb :Tableau (5) Entier ; i : Entier ;
Debut
  Pour i ← 1 a 5 faire
    nb(i) ← i * i ;
  fPour ;
  Pour i ← 1 a 5 faire
    Ecrire ( nb(i))
  fPour ;
Fin.
```

Peut-on simplifier cet algorithme tout en ayant le même résultat ?

Exercice 10 : Que produit l'algorithme suivant ?

```
Var suite :Tableau (7) Entier ;
  i:Entier ;
Debut
  suite(1) ← 1 ;
  suite(2) ← 1 ;
  Pour i ← 3 a 7 faire
    suite(i) ← suite(i-1) + suite(i-2) ;
  fPour ;
  Pour i ← 1 a 7 faire
    Ecrire(suite(i)) ;
  fPour ;
Fin.
```

Exercice 11 :

Soient vect1 et vect2 deux vecteurs de même taille. Ecrire l'algorithme qui détermine :

- leur somme.
- leur produit scalaire.

Exercice 12 : Écrire un algorithme qui remplit un tableau avec les N premiers entiers naturels, puis ajoute 1 à toutes les valeurs de rang pair de ce tableau et retranche 1 à toutes les valeurs de rang impair et enfin affiche l'intégralité des tableaux, initial et résultant.

Exercice 13 :

Ecrire un algorithme qui détermine le nombre d'occurrences de chaque élément d'un tableau d'entiers.

Exercice 14 : Soient deux vecteurs vectA et vectB ayant le même nombre d'éléments. Ecrire un algorithme qui permet de construire, à partir de ces vecteurs, deux vecteurs vMax et vMin définis par : vMax(i) = maximum entre vectA(i) et vectB(i) et vMin(i) = minimum entre vectA(i) et vectB(i). On comptabilisera au fur et à mesure les cas d'égalités.

Exercice 15 : Soit vect un tableau de nombres entiers triés dans l'ordre croissant. Écrire un algorithme qui insère un nombre nbr dans ce tableau, tout en conservant le tri.

Exercice 16 : (ETLD 2013/2014) On considère un tableau d'entiers positifs vect de taille n. Écrire un algorithme ou un programme C qui détermine le deuxième plus grand écart entre deux éléments consécutifs de ce tableau.

Exemple : Soit vect=(1,8,3,10,9,5,8,13,7,4) un tableau de 10 entiers. Le deuxième plus grand écart est celui entre 13 et 7 et il vaut 6.

NB : L'écart entre deux nombres est la valeur absolue de leur soustraction. i.e. l'écart entre 4 et 5 est égal à l'écart entre 5 et 4 et cet écart vaut 1.

Exercice 17 : (ETLD 2013/2014 spécial) On considère un tableau d'entiers Vect de taille n. Une *tranche* d'un tableau est une séquence d'éléments consécutifs de celui-ci. Une tranche d'ordre k est une séquence contenant entre 1 et k éléments. La valeur d'une tranche est la somme de ses éléments.

Écrire un algorithme ou un programme C qui détermine la tranche d'ordre trois de ce tableau ayant la plus petite valeur.

- **NB** : Une tranche d'ordre 3 est une séquence d'éléments consécutifs ayant au plus 3 éléments.

Exemple : Soit Vect= $(-20, 8, -11, 10, -9, -15, 8, 13, -7, 4)$ un tableau de 10 entiers.

La tranche $(-9, -15)$ est celle ayant la plus petite valeur $((-9) + (-15) = -24)$ parmi les tranches d'ordre 3. La tranche $(-20, 8, -11)$ est une tranche d'ordre 3. Sa valeur est $(-20 + 8 - 11 = -23) > -24$.

Exercice 18 : (ETLD 2022/2023 remplacement) :

T est un tableau de n entiers ($2 \leq n \leq 50$). On appelle sous-tableau de **T**, une séquence d'éléments consécutifs de celui-ci définie par le couple d'entier (s, t) où s est l'indice dans **T** du premier élément de la séquence, et t est la taille de la séquence.

Écrire un algorithme ou un programme C qui :

1. Remplit le tableau **T** par des valeurs données par l'utilisateur.
2. Lit les informations définissant un sous-tableau de **T**, puis ramène les éléments du sous-tableau à la fin en conservant leur ordre.
3. Affiche le tableau **T** résultat.

Exemple : En supposant des indices de tableau commençant à partir de 1,

Pour **T** =

8	3	4	2	7	1	9	-5
---	---	---	---	---	---	---	----

 et $(s, t) = (2, 5)$, on obtient, **T** =

8	9	-5	3	4	2	7	1
---	---	----	---	---	---	---	---