# Exercice 7.1

Ecrivez un algorithme qui permette de saisir un nombre quelconque de valeurs, et qui les range au fur et à mesure dans un tableau. Le programme, une fois la saisie terminée, doit dire si les éléments du tableau sont tous consécutifs ou non.

Par exemple, si le tableau est :

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 12 | 13 | 14 | 15 | 16 | 17 | 18 |

ses éléments sont tous consécutifs. En revanche, si le tableau est :

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 9 | 10 | 11 | 15 | 16 | 17 | 18 |

ses éléments ne sont pas tous consécutifs.

tab EST\_DU\_TYPE LISTE

i EST\_DU\_TYPE NOMBRE

trier EST\_DU\_TYPE CHAINE

trier PREND\_LA\_VALEUR "vrai"

POUR i ALLANT\_DE 0 A 5

DEBUT\_POUR

LIRE tab[i]

FIN\_POUR

POUR i ALLANT\_DE 0 A 4

DEBUT\_POUR

SI (tab[i] != tab[i+1]-1) ALORS

DEBUT\_SI

trier PREND\_LA\_VALEUR "faux"

FIN\_SI

FIN\_POUR

AFFICHER trier

# Exercice 7.2

Ecrivez un algorithme qui trie un tableau dans l’ordre décroissant.

Vous écrirez bien entendu deux versions de cet algorithme, l'une employant le tri par sélection, l'autre le tri à bulles.

tab EST\_DU\_TYPE LISTE

i EST\_DU\_TYPE NOMBRE

temp EST\_DU\_TYPE NOMBRE

flag EST\_DU\_TYPE CHAINE

tab[0] PREND\_LA\_VALEUR 10

tab[1] PREND\_LA\_VALEUR 5

tab[2] PREND\_LA\_VALEUR 15

tab[3] PREND\_LA\_VALEUR 2

flag PREND\_LA\_VALEUR "vrai"

TANT\_QUE (flag == "vrai") FAIRE

DEBUT\_TANT\_QUE

flag PREND\_LA\_VALEUR "faux"

POUR i ALLANT\_DE 0 A 2

DEBUT\_POUR

SI (tab[i] > tab[i+1] ) ALORS

DEBUT\_SI

temp PREND\_LA\_VALEUR tab[i]

tab[i] PREND\_LA\_VALEUR tab[i+1]

tab[i+1] PREND\_LA\_VALEUR temp

flag PREND\_LA\_VALEUR "vrai"

FIN\_SI

FIN\_POUR

FIN\_TANT\_QUE

AFFICHER tab[0]

AFFICHER " , "

AFFICHER tab[1]

AFFICHER " , "

AFFICHER tab[2]

AFFICHER " , "

AFFICHER tab[3]

# Exercice 7.3

Ecrivez un algorithme qui inverse l’ordre des éléments d’un tableau dont on suppose qu'il a été préalablement saisi (« les premiers seront les derniers… »)

tab EST\_DU\_TYPE LISTE

i EST\_DU\_TYPE NOMBRE

temp EST\_DU\_TYPE NOMBRE

tab[0] PREND\_LA\_VALEUR 10

tab[1] PREND\_LA\_VALEUR 5

tab[2] PREND\_LA\_VALEUR 15

tab[3] PREND\_LA\_VALEUR 2

POUR i ALLANT\_DE 0 A tab.length/2

DEBUT\_POUR

temp PREND\_LA\_VALEUR tab[i]

tab[i] PREND\_LA\_VALEUR tab[3-i]

tab[3-i]PREND\_LA\_VALEUR temp

FIN\_POUR

# Exercice 7.4

Ecrivez un algorithme qui permette à l’utilisateur de supprimer une valeur d’un tableau préalablement saisi. L’utilisateur donnera l’indice de la valeur qu’il souhaite supprimer. Attention, il ne s’agit pas de remettre une valeur à zéro, mais bel et bien de la supprimer du tableau lui-même ! Si le tableau de départ était :

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 12 | 8 | 4 | 45 | 64 | 9 | 2 |

Et que l’utilisateur souhaite supprimer la valeur d’indice 4, le nouveau tableau sera :

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 12 | 8 | 4 | 45 | 9 | 2 |

tab[0] PREND\_LA\_VALEUR 10

tab[1] PREND\_LA\_VALEUR 5

tab[2] PREND\_LA\_VALEUR 15

tab[3] PREND\_LA\_VALEUR 2

AFFICHER "quel element du tableau voulez vous supprimer : "

LIRE indice

POUR i ALLANT\_DE indice A tab.length -2

DEBUT\_POUR

tab[i] PREND\_LA\_VALEUR tab[i+1]

FIN\_POUR

Supprimer tab[tab.length -1]

AFFICHER tab[0]

AFFICHER " , "

AFFICHER tab[1]

AFFICHER " , "

AFFICHER tab[2]

AFFICHER " , "

AFFICHER tab[3]

# Exercice 7.5

Ecrivez l'algorithme qui recherche un mot saisi au clavier dans un dictionnaire. Le dictionnaire est supposé être codé dans un tableau préalablement rempli et trié.

dictionnaire EST\_DU\_TYPE LISTE

i EST\_DU\_TYPE NOMBRE

mot\_cherche EST\_DU\_TYPE CHAINE

flag EST\_DU\_TYPE CHAINE

debut EST\_DU\_TYPE NOMBRE

fin EST\_DU\_TYPE NOMBRE

fin PREND\_LA\_VALEUR dictionnaire.length

début PREND\_LA\_VALEUR 0

flag PREND\_LA\_VALEUR « faux »

lire mot\_cherche

TANT\_QUE (fin-debut > 1) FAIRE

DEBUT\_TANT\_QUE

SI (dictionnaire[(fin+debut)/2] > mot\_cherche) ALORS

DEBUT\_SI

debut = (fin+debut)/2 +1

FIN\_SI

SINON

DEBUT\_SINON

SI (dictionnaire[(fin+debut)/2] < mot\_cherche) ALORS

DEBUT\_SI

fin = (fin+debut)/2 -1

FIN\_SI

SINON

DEBUT\_SINON

flag PREND\_LA\_VALEUR « vrai »

FIN\_TANT\_QUE

AFFICHER flag

# Exercice 7.6

Écrivez un algorithme qui fusionne deux tableaux (déjà existants) dans un troisième, qui devra être trié.

Attention ! On présume que les deux tableaux de départ sont préalablement triés : il est donc irrationnel de faire une simple concaténation des deux tableaux de départ, puis d'opérer un tri : comme quand on se trouve face à deux tas de papiers déjà triés et qu'on veut les réunir, il existe une méthode bien plus économique (et donc, bien plus rationnelle...)

tab1 EST\_DU\_TYPE LISTE

tab2 EST\_DU\_TYPE LISTE

tab3 EST\_DU\_TYPE LISTE

i EST\_DU\_TYPE NOMBRE

ii EST\_DU\_TYPE NOMBRE

iii EST\_DU\_TYPE NOMBRE

i PREND\_LA\_VALEUR 0

ii PREND\_LA\_VALEUR 0

iii PREND\_LA\_VALEUR 0

tab1[0] PREND\_LA\_VALEUR 5

tab1[1] PREND\_LA\_VALEUR 10

tab1[2] PREND\_LA\_VALEUR 15

tab1[3] PREND\_LA\_VALEUR 20

tab2[0] PREND\_LA\_VALEUR 2

tab2[1] PREND\_LA\_VALEUR 8

tab2[2] PREND\_LA\_VALEUR 9

tab2[3] PREND\_LA\_VALEUR 25

TANT\_QUE (i <= 3 & ii <=3) FAIRE

DEBUT\_TANT\_QUE

SI (tab2[ii]<tab1[i]) ALORS

DEBUT\_SI

tab3[iii] PREND\_LA\_VALEUR tab2[ii]

AFFICHER tab3[iii]

AFFICHER " , "

ii PREND\_LA\_VALEUR ii+1

iii PREND\_LA\_VALEUR iii+1

SI (tab2[ii]>tab1[i]) ALORS

DEBUT\_SI

tab3[iii] PREND\_LA\_VALEUR tab1[i]

AFFICHER tab3[iii]

AFFICHER " , "

i PREND\_LA\_VALEUR i+1

iii PREND\_LA\_VALEUR iii+1

FIN\_SI

FIN\_SI

FIN\_TANT\_QUE