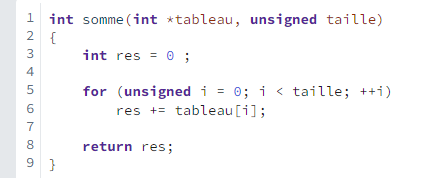
**Travaux pratiques 5 programmation en langage C – Tableaux et Fonctions**

## Exercice 1

##### Ecrire une fonction qui calcule la somme de tous les éléments d’un tableau de int.

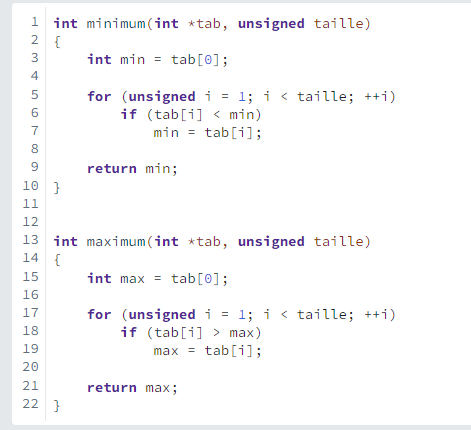
Corrigé.



## Exercice 2

##### Ecrire deux fonctions : une qui retourne le plus petit élément d’un tableau de int et une qui renvoie le plus grand élément d’un tableau de int.

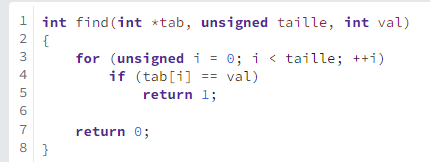
Corrigé.



## Exercice 3

##### Ecrire une fonction qui teste la présence d’une valeur dans un tableau de int. Celle-ci retournera 1 si un ou plusieurs éléments du tableau sont égaux à la valeur recherchée, 0 sinon.Pour cet exercice, vous utiliserez les coupures suivantes :

Corrigé.



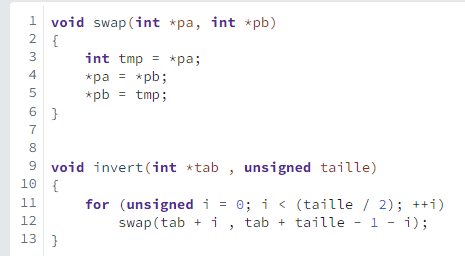
## 

## Exercice 4

Ecrire une fonction qui inverse le contenu d’un tableau (le premier élément devient le dernier, l’avant-dernier le deuxième et ainsi de suite).

Utilisez le swap (permutation)

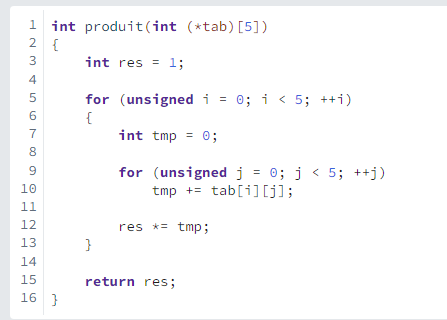
Corrigé.



## Exercice 5

Ecrire une fonction qui calcule le produit de la somme des éléments de chaque ligne d’un tableau de int à deux dimensions (ce tableau comprend cinq lignes et cinq colonnes).

Corrigé.



## Exercice 6

##### Triangle de Pascal

Les triangles de Pascal sont des objets mathématiques. Voici un exemple

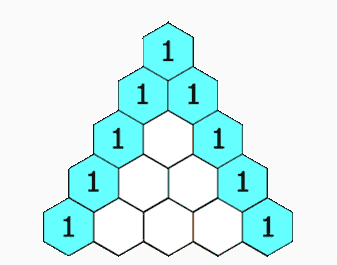
1

1 1

1 2 1

1 3 3 1

1 4 6 4 1



Corrigé.

La première chose que nous allons faire est donc définir un tableau à deux dimensions (nous fixerons la taille des dimensions à dix) dont tous les éléments sont initialisés à zéro. Ensuite, nous allons demander à l’utilisateur d’entrer la taille du triangle qu’il souhaite obtenir (celle-ci ne devra pas être supérieure aux dimensions du tableau).

Passons à la fonction de création du triangle de Pascal. Celle-ci devra mettre en œuvre l’algorithme suivant.

N = taille du triangle de Pascal fournie par l’utilisateur

Mettre la première case du tableau à 1

Pour i = 1, i < N, i = i + 1

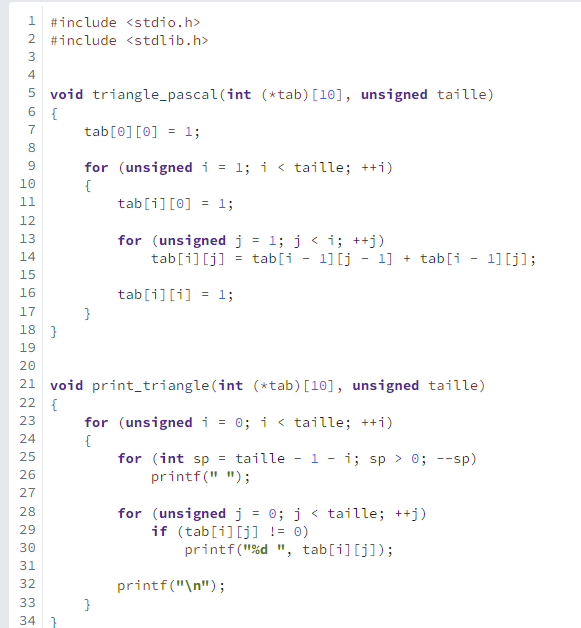
Mettre la première case de la ligne à 1

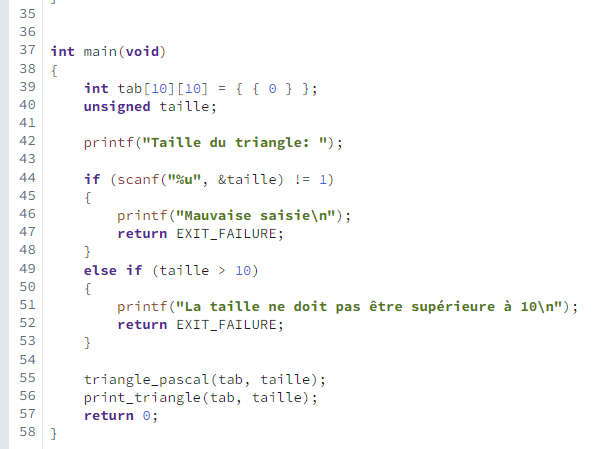
Pour j = 1, j < i, j = j + 1

La case [i,j] prend la valeur [i - 1, j - 1] + [i - 1, j]

Mettre la dernière case de la ligne à 1

Enfin, il vous faudra écrire une petite fonction pour afficher le tableau ainsi créé.





## Exercice 7

Implémenter, en C, l’algorithme de Tri suivant:

Algorithm TRi\_select(T, n)

For i = 1 to n-1 do

Min =i

For j = i+1 to n do

If(T[j] < T[min]) then

min =j

End If

End For

if(min <> i) then

Temp := T[min]

T[min]:= T[j]

T[j] := Temp

endif

End For

End

## Exercice 8

Ecrire un programme C qui calcule et affiche le produit scalaire de deux vecteurs à n entiers

P = T1[i] x T2[i]

|  |
| --- |
| 2 |
| 34 |
| 12 |
| 3 |
| 5 |

T1

|  |
| --- |
| 12 |
| 4 |
| 34 |
| 4 |
| 6 |

T2

## 

## Exercice 9

Ecrire un programme C qui calcule et affiche le produit d’une matrice nxm par un vecteur de dimension m

C[i] = x V[j]

C

|  |
| --- |
|  |
|  |
|  |
| C[i] |
|  |

=

M

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
| M[i,1] | M[i,2] | M[i,3] |
|  |  |  |

x

V

|  |
| --- |
| V[1] |
| V[2] |
| V[3] |

C[2] = M[2][1]xV[1]+M[2][2]xV[2]+M[2][3]xV[3]

## Exercice 10

Ecrire un programme C qui calcule et affiche le produit d’une matrice nxm par une matrice mxt. le résultat est une matrice n xt.

C[i][j] = A[i][k] x B[k][j] pour i de 1 à n et j de 1 à m

C

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | **C[i][j]** |  |  |
|  |  |  |  |

=

A

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
| **A[i][1]** | **A[i][2]** | **A[i][3]** | **A[i][4]** | **A[i][5]** |
|  |  |  |  |  |

x

B

|  |  |  |  |
| --- | --- | --- | --- |
|  | **B[1][j]** |  |  |
|  | **B[2][j]** |  |  |
|  | **B[3][j]** |  |  |
|  | **B[4][j]** |  |  |
|  | **B[5][j]** |  |  |