

Leçon 3 : Les Tableaux en C++

Table des matières



I - 1- Les Tableaux à une dimension	3
II - Exercice	5
III - 2- Les Tableaux à deux dimensions	6
IV - Exercice	8
V - 3- Les chaînes comme des tableaux	10
VI - Exercice	12

1- Les Tableaux à une dimension



🔑 Définition : 1.1- Définition

Encore appelé *vecteur*, le *tableau à une dimension* est composé d'une *seule ligne* et *plusieurs colonnes*.

1.2- Déclaration

Type `nom_tableau[dim];`

dim représente la taille du tableau, c'est-à-dire le nombre d'éléments pouvant y être contenu.

En d'autres termes, le compilateur réserve la *taille (dim)* en mémoire pour ranger les éléments du *tableau*.

Les *indices* du tableau partent de 0 (*zéro*) à *dim-1* cela signifie que pour un *tableau de 10 éléments* nous aurons des indices qui vont de 0 à 9.

👉 Exemple : 1.3- Exemple

Soit le tableau *tab* de type entier représenté ci-dessous et comprenant 5 élément

Pour la déclaration on aura :

`int tab[5];`

21	-12	14	9	-5
----	-----	----	---	----

NB : L'indice commence toujours par 0 en langage donc on aura :

`tab[0]` a pour valeur 21

`tab[1]` a pour valeur -12

`tab[2]` a pour valeur 14

`tab[3]` a pour valeur 9

`tab[4]` a pour valeur -5

1.4-Application

Écrire un programme en langage C++ permettant de renseigner 10 nombres pairs strictement supérieurs à zéro dans un tableau et de l'afficher.

```
1 #include <iostream> //permet d'utiliser le cout
2 using namespace std; // permet d'utiliser le endl
```

```
3 main()
4
5 {
6 //Déclaration des variables
7 int i,tab[10];
8 cout<<"***** SAISIE DU TABLEAU *****"<<endl;
9 //Parcours des colonnes du tableau
10 for(i=0;i<10;i++) // permet de parcourir les colonnes du tableau en partant de 0
   à 9
11 {
12 do // cette boucle permettra de répéter la même saisie jusqu'à ce la condition
   mentionnée dans l'énoncé soit respecter
13 {
14 cout<<"Entrez un nombre pair à la position "<<i+1<<" strictement supérieur à
   zéro!"<<endl; // affiche le message entre les doubles griffes et retourne à la
   ligne suivant
15 cin>>tab[i]; // récupère la valeur saisie dans un tableau
16 }
17 while(tab[i]<=0 || tab[i]%2!=0); vérifie si la valeur saisie est inférieur ou le
   reste de la division par 2 est différent de zéro
18 }
19 cout<<"***** AFFICHAGE DU TABLEAU *****"<<endl;
20 for(i=0;i<10;i++) // permet d'afficher le dix éléments du tableau en commençant
   pas l'indice 0.
21 {
22 cout<<"| "<<tab[i];
23 }
24 }
25
```

Exercice


 II

Énoncé :

Écrire un programme qui permet de remplir un tableau de 10 éléments et ensuite d'afficher les nombres pairs de ce tableau.

Solution :

```

[ ]
[ ]
[ ]

{
    [ ] i,tab [ ] ;
for([ ] ; [ ] 10; [ ] )
{
    [ ] Entrez un nombre à la position " [ ] i+1 [ ] endl;
    [ ]
}
for([ ] ; [ ] 10; [ ] )
{
    [ ] ( [ ] )
    {
        [ ] "I " [ ] ;
    }
}
}
}

```

2- Les Tableaux à deux dimensions



Définition : 2.1- Définition

Encore appelé *matrice*, le *tableau à deux dimensions* est constitué de *plusieurs lignes* et *plusieurs colonnes*.

2.2- Déclaration

Type `nom_tableau[dim1] [dim2];`

- *dim1* représente le nombre de lignes
- *dim2* représente le nombre de colonnes
- Le premier *indice de ligne* a pour *valeur 0* et le dernier a pour *valeur dim1-1*
- Le premier *indice de colonne* a pour *valeur 0* et le dernier a pour *valeur dim2-1*.

Pour parcourir une matrice, l'on parcourt chaque ligne de la matrice et étant sur la ligne, l'on parcourt ses différentes colonnes.

Exemple : 2.3- Exemple

Soient 6 éléments contenus dans un tableau `tab` de type réel représenté ci-dessous et comprenant 2 lignes et 3 colonnes.

2.2	4.0	5.5
0.6	7.2	8.2

NB : L'indice commence toujours par 0 en langage donc on aura :

La valeur 2.2 est situé dans la *ligne* ayant pour *indice 0* et la *colonne* ayant pour *indice 0* : `tab[0][0]`.

La valeur 7.2 est situé dans la *ligne* ayant pour *indice 1* et la *colonne* ayant pour *indice 1* : `tab[1][1]`.

2.4- Application

Écrire un programme en langage C++ permettant de renseigner les 12 premiers nombres multiples de 3 dans une matrice de 3 lignes et 4 colonnes.

```
1 #include <iostream> //permet d'utiliser le cout
2 using namespace std; // permet d'utiliser le endl
3 main ()
4
5 {
6     int i,k,j,tab[3][4];
```

```
7 //***** RENSEIGNEMENT DU TABLEAU *****
8 k=1;
9 // parcours des lignes de la matrice
10 for(i=0;i<3;i++)
11 {
12 //parcours et renseignement des colonnes de chaque ligne de la matrice
13 for(j=0;j<4;j++)
14 {
15 tab[i][j]=3*k;
16 k++;
17 }
18 }
19 cout<<" ***** AFFICHAGE DU TABLEAU *****"<<endl;
20 for(i=0;i<3;i++)
21 {
22 for(j=0;j<4;j++)
23 {
24 //Affichage du contenu de chaque ligne suivi d'une tabulation
25 cout<<"\t"<<tab[i][j];
26 }
27 //retour-chariot avant l'affichage du contenu de la ligne suivante
28 cout<<endl;
29 }
30 }
31
```

Exercice

IV

Énoncé :

Écrire un programme qui permet de remplir une matrice carré de dimension 4 et afficher la somme de la diagonale de cette matrice.

Solution :

```

[ ]
[ ]
[ ]

{
    [ ] somd,i,j,tab [ ] ;
    [ ] ( [ ] ; [ ] 4; [ ] )
    {
        [ ] ( [ ] ; [ ] 4; [ ] )
    }
    {
        [ ] Entrez un nombre à la ligne " [ ] i+1 [ ] " et à la colonne "<<j+1<<endl;
        [ ]
    }
}

somd [ ]
[ ] ( [ ] ; [ ] 4; [ ] )
{
    [ ] ( [ ] ; [ ] 4; [ ] )
    {
        [ ] ( [ ] )
    }
    {
        [ ] += [ ]
    }
}
}
}

```



```
    La somme des éléments de la diagonale est "    endl;  
}
```

3- Les chaînes comme des tableaux

V

3.1- Définition

Une chaîne de caractères est un tableau constitué de caractère , chaque caractère est représenté par une colonne du tableau.

Exemple :

```
1 #include <iostream>
2 #include <string> // pour manipuler les chaîne de caractère.
3 using namespace std;
4
5 main()
6 {
7     string val;
8     val = "Bonjour";
9     cout << "UVCI vous dit " << val << "." << endl;
10
11     val[3] = 's'; //On modifie la quatrième lettre
12     val[5] = 'i'; //On modifie la sixième lettre
13
14     cout << "UVCI vous dit " << val << "!" << endl;
15
16 }
17
```

Pour connaître la taille d'une chaîne on utilise la fonction *size()* et pour ajouter des caractères en fin de chaîne on utilise *push_back()*.

Exemple :

```
1 #include <iostream>
2 #include <string> // pour manipuler les chaîne de caractère.
3 using namespace std;
4
5 main()
6 {
7     string val;
8     val = "Bonjour chers étudiants ";
9     cout << "Le nombre de caractère est : " << val.size() << endl;
10     val.push_back('d');
11     val.push_back('e');
12     val.push_back(' ');
13     val.push_back('U');
14     val.push_back('V');
15     val.push_back('C');
```

```

16     val.push_back('I');
17
18     cout << val << "!" << endl;
19
20 }
21

```

3.2- Application

Écrire un programme en langage C++ permettant de saisir un mot et affiche le nombre d'occurrence de voyelle dans ce mot.

```

1 #include <iostream>
2 #include <string> // pour manipuler les chaîne de caractère.
3 using namespace std;
4
5 main()
6 {
7     string mot;
8     int i,nbcat;
9     cout << "Entrer le mot SVP"<<endl;
10    cin>>mot;
11
12    nbcat=0;
13
14    for(i=0;i<mot.size();i++)
15    {
16        if(mot[i]=='i' || mot[i]=='a' || mot[i]=='e' || mot[i]=='u' || mot[i]=='y' || mot
17        [i]=='o')
18        {
19            nbcat++;
20        }
21    }
22
23    cout << "Le nombre de voyelle est : "<<nbcat<<endl;
24
25 }

```

Exercice

VI

Énoncé :

Écrire un programme qui permet de remplacer les caractères 'a' et 'o' d'un mot saisi par 'y' et d'afficher ensuite le résultat.

Solution :

```

[ ]
[ ]
[ ]
[ ]

{
string mot;
int i;
[ ] "Entrer le mot SVP" [ ] endl;
[ ]
for( [ ] ; i < [ ] ; [ ] )
{
[ ] ( [ ] 'a' [ ] [ ] 'o')
{
[ ]
}
}
[ ] "Le nouveau mot est : " [ ] endl;
}
```