# TABLEAUX ET CHAINES DE CARACTERES EN C++
## *Méthode classique compatible avec le langage C*

Dans ce chapitre, nous abordons la méthode classique, compatible avec le langage C.

Dans le prochain chapitre, nous verrons une autre solution pour créer des tableaux et chaines de caractères qui utilise la librairie standard du C++.

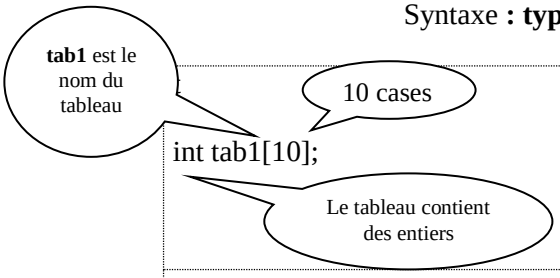
## 1. Remarques générales

- Un tableau permet de stocker un ensemble de données de même type.
- Les chaînes de caractères correspondent à une sorte de tableau particulier qui permet de stocker des caractères et dont le dernier élément est un caractère particulier appelé marque de fin de chaîne.
- En C ou C++, on peut créer des tableaux à 1, 2 ou n dimensions (dans la limite de la mémoire disponible)

## 2. Utilisation des tableaux

### *2.1. Déclaration*

Syntaxe **: type NomDuTableau[NombreDeCases]**

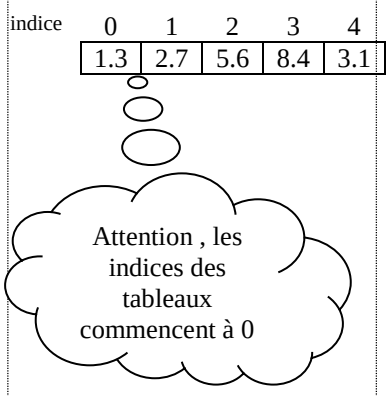



| | |
|---|---|
| int tab1[10]; | déclare un tableau de 10 entiers non initialisés<br>? ? ? ? ? ? ? ? ? ? |
| int tab[5]={1,2,5,8,3}; | déclare un tableau de 5 entiers initialisés<br>1 2 5 8 3 |
| float tab[5]={0}; | déclare un tableau de 5 réels initialisés à 0<br>0 0 0 0 0 |
| int tab[5]={0,3}; | déclare un tableau de 5 cases initialisées<br>0 3 0 0 0 |
| int tab[2][3] ; | déclare un tableau à 2 dimensions de 6 cases non initialisées |
| int tab[2][3]={{12,2,44},{6,5,3}}; | déclare un tableau à 2 dimensions de 6 cases initialisées |

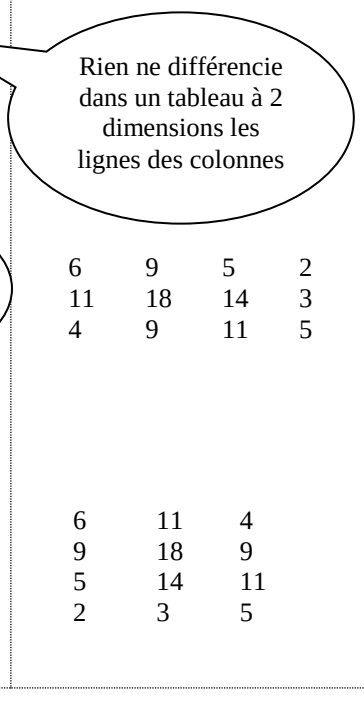
### 2.2. Accès aux composantes des tableaux

Une fois le tableau déclaré, on peut accéder facilement à ses éléments. Pour cela, il suffit de connaître leur position (leur indice).

```
double tab[5]={1.3 ,2.7 ,5.6 ,8.4 ,3.1 };

int i;

cout << " La deuxieme case contient " << tab[1] << endl;

cout <<" \nAffichage de toutes les cases " << endl;

for (i=0 ; i<5 ; i++)
    cout << "\tLa case numero "<< i <<" contient "
    << tab[i] <<endl;
```




```
int tab2[3][4]={{6,9,5,2},{11,18,14,3},{4,9,11,5}};

int i , j;

cout <<"\nAffichage dans un sens \n\n" << flush ;
//**** Affichage dans un sens *********
for (i=0 ; i<3 ; i++ )
{
    for (j=0 ; j<4 ; j++)
        cout << tab2[i][j] << "\t";

    cout << endl;
}

cout <<"\nAffichage dans l'autre sens \n\n" << flush ;


//**** Affichage dans l'autre sens *********
for (j=0 ; j<4 ; j++ )
{
    for (i=0 ; i<3 ; i++)
         cout <<  tab2[i][j] << "\t";

     cout << endl ;
}
```



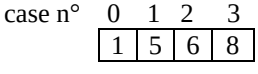

### 2.3. Quelques conseils importants

- **Ne dépassez jamais l'indice maximum d'un tableau**

int tab[4] = {1,5,6,8} ;

tab[3]=7 ; // OK

tab[4] = 8 ; // plantage car la case n°4 n'existe pas




- **Initialisez de préférence vos tableaux lors de la déclaration**

- **Attention à bien choisir le type de vos tableaux. Si vous stockez des valeurs , toutes entières sauf une, il vous faudra choisir un tableau de réels**

## 3. Les chaînes de caractères

Une chaîne de caractères est traitée comme un *tableau à une dimension de caractères* (vecteur de caractères). Il existe quand même des notations particulières et une bonne quantité de fonctions spéciales pour le traitement de tableaux de caractères.

### *3.1. Déclaration*

```
char chaine[4+1]="TOTO";
```

pour tenir compte de la marque de fin de chaîne

déclare une chaîne de 4 caractères

| indice | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| | 'T' ou 84 ou 0x54 | 'O' ou 79 ou 0x4F | 'T' ou 84 ou 0x54 | 'O' ou 79 ou 0x4F | '\0' ou 0 |

*idem que*

```
char chaine[4+1]={'T'O','T','O','\0'};
```

```
char chaine2[]="BONJOUR";
```

Ici, comme on n'indique pas la taille, le système calculera la taille de la chaîne "BONJOUR" et en déduira la taille de chaine2 (c'est à dire 7+1 ).

### *3.2. Saisie et affichage de chaînes de caractères*

Voyons sur un exemple les différentes options de saisie et d'affichage.

```
char message1[80+1];
char message2[80+1];
```

Quand vous ne connaissez pas la taille de la chaîne de caractères, il faut la surdimensionner

```
cin >> message1 ;
```

1ere solution pour saisir un texte
Convient si le texte que vous saisissez ne contient pas d'espaces

```
gets(message2);
```

2ème solution pour saisir un texte
convient dans tous les cas. **Inclure <stdio.h>**

```
cout << message1 ;
```

1ère solution pour afficher un texte sur écran en mode console
convient quelque soit la chaîne à afficher, qu'elle comprenne ou non des espaces

```
puts(message2);
```

2ème solution pour afficher un texte sur écran en mode console
convient dans tous les cas

### *3.3. Manipulation des chaînes de caractères*

Pour manipuler les chaînes de caractères, on dispose d'un ensemble de fonctions prédéfinies commençant toutes par **str** et définies dans le fichier <string.h>. Voici résumé ci-dessous les principales.

- **Copie de chaînes : Fonctions strcat , strcpy , strdup , strncat , strncpy**
- **Chercher des chaînes , des caractères individuels ou les caractères d'un jeu spécifié (strchr , strcspn , strpbrk , strrchr , strspn , strstr)**

- **Effectuer des comparaisons de chaînes (strcmp, strcmpi, stricmp, strncmp, strnicmp)**

- **Remplir une chaîne avec un caractère spécifié (strnset , strset)**

- **Diviser des chaînes en éléments (strtok)**

Voyons l'utilisation de certaines de ces fonctions (la documentation devrait vous aider à utiliser les autres).

```
#include <iostream.h>  // inclusion de fichiers d'entêtes
#include <string.h>

/*-----------------------------------------------
  Fonction principale
-----------------------------------------------*/
void main(void)
{
char chaine1[80+1]="Salut ";
char chaine2[80+1];
char chaine3[]="la compagnie";

//**** copie de la chaine1 dans la chaine2            *****
//**** Il ne faut surtout pas écrire chaine2=chaine1 *****
strcpy(chaine2 , chaine1);

//**** concaténation de chaines ******
strcat(chaine2 , "a tous ");
strcat(chaine1 , chaine3 );

cout <<"\nLa chaine1 est "<< chaine1 << flush ;
cout <<"\nLa chaine2 est "<< chaine2 << flush ;
cout <<"\nLa longueur de la chaine 1 est " << strlen(chaine1)
     << flush;
}
```

Pour finir sur les chaînes de caractères, citons deux fonctions de la bibliothèque standard (stdio.h) très utiles. Ces fonctions permettent de convertir des nombres en chaines (**sprintf**) et vis-versa (**sscanf**). L'exemple ci-dessous en illustre l'utilisation

```
#include <iostream.h>
#include <stdio.h>
#include <string.h>

void main(void)
     {
     int valeur1 , valeur2;
     int heure=14,minutes=3,secondes=25;
     char chaine2[50];
     char chaine[10]="103";

     //**** conversion d'une chaine en un nombre codé en base 10 ***
     sscanf(chaine,"%d",&valeur1);
     cout <<"\n Valeur1 vaut "<< valeur1 << flush ;

     //**** conversion d'une chaine en un nombre codé en base 16 ***
     sscanf(chaine,"%x",&valeur2);
     cout <<"\n Valeur2 vaut "<< valeur2 << endl << flush ;

     //***** Création d'une chaine à partir de nombres ****
     sprintf(chaine2 ,"Il est %dh %dmn %d secondes ",heure,minutes,
                      secondes);

     cout << chaine2 << flush ;

     }
```

## 4. Les tableaux de chaînes

Souvent, il est nécessaire de mémoriser une suite de mots ou de phrases dans des variables. Il est alors pratique de créer un tableau de chaînes de caractères, ce qui allégera les déclarations des variables et simplifiera l'accès aux différents mots (ou phrases).

Un tableau de chaînes de caractères correspond à un tableau à deux dimensions du type **char**, où ***chaque ligne contient une chaîne de caractères***.

***Déclaration***

La déclaration `char JOUR[7][8+1];`
réserve l'espace en mémoire pour 7 mots contenant 8 caractères (+1 pour la fin de chaine).

Lors de la déclaration il est possible d'initialiser toutes les composantes du tableau par des chaînes de caractères constantes:

```
char JOUR[7][8+1]= {"lundi", "mardi", "mercredi",
                    "jeudi", "vendredi", "samedi",
                    "dimanche"};
```

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | |
|---|---|---|---|---|---|---|---|---|---|---|
| JOUR: | 'l' | 'u' | 'n' | 'd' | 'i' | '\0' | | | | 0 |
| | 'm' | 'a' | 'r' | 'd' | 'i' | '\0' | | | | 1 |
| | 'm' | 'e' | 'r' | 'c' | 'r' | 'e' | 'd' | 'i' | '\0' | 2 |
| | ... | ... | ... | ... | ... | ... | ... | ... | ... | 3 … |
| | 'd' | 'i' | 'm' | 'a' | 'n' | 'c' | 'h' | 'e' | '\0' | 6 |

**Accès aux chaînes**

Il est possible d'accéder aux différentes chaînes de caractères d'un tableau, en indiquant simplement la ligne correspondante.

L'exécution des trois instructions suivantes:

```
char jour[7][9]= {"lundi", "mardi", "mercredi",
                  "jeudi", "vendredi",
                  "samedi", "dimanche"};
int i = 2;
cout <<" Aujourd'hui, c'est " << jour[i] << flush;
```

affichera la phrase: ***Aujourd'hui, c'est mercredi !***

**Accès à un caractère donné dans une chaîne**

Comme tout tableau, il est facile d'accéder à une case donnée.
Par exemple, jour[2][3] correspond au caractère '**c**' (ligne n°2 et colonne n° 3)