Tableaux

Immaculée Conception
ENSEIGNEMENT PRIVÉ

Mathieu DOMER
BTS CIEL-IR

Qu’est-ce qu’un tableau ?

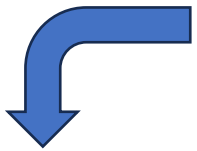
Un tableau est un ensemble de données du **même type** stockées de manière **contiguë** en mémoire et accessibles via **une seule variable**.

- Un tableau de 4 caractères (`char`)

- Un tableau de 5 entiers de 4 octets (`int`)

Déclaration et initialisation

```
type nom [taille];
float notes [3];
int nbCubesParEtages [30], decimales_pi[100000];
```



Constante !

```
int tab [3] = {-7, 64, 0};
float tab[7] = {0.5, 0, -2.90, 0.85, 2.30, 3, 8.2};
char couleur [4] = {'B', 'L', 'E', 'U'};
```

Quelques règles

- La taille du tableau est fixée à la création et ne peut être changer après. Soyez prévoyant !
- Le nombre de valeurs entre accolades ne doit pas être supérieur au nombre d'éléments du tableau (sa taille).
- Les valeurs entre accolades doivent être des constantes (l'utilisation de variables provoquera une erreur du compilateur).
- Si le nombre de valeurs entre accolades est inférieur au nombre d'éléments du tableau, les derniers éléments sont initialisés à 0.
- Il doit y avoir au moins une valeur entre accolades.
- Si la taille n’est pas spécifiée, la taille du tableau est fixée par le nombre d’éléments entre accolades.

Déclaration et initialisation

Que donne les instructions suivantes ?

```
int tab [2] = {-7, 64, 0};
int a = 0, tab [3] = {-7, 64, a};
int tab [6] = {-7, 64, 0};
int tab [3] = { 0 };
int tab [3] = { };
int tab [] = {-7, 64, 0};
```

Accès aux données

Accès à la valeur stockée à l'indice `n` du tableau `tab` :

```
tab[n]
```

L'indice de la première cellule est égal à 0.

L'indice de la dernière cellule est égal à taille - 1.

Accès aux données

```c
int tab[9] = {3, -7, 268, 4, 8, 15, 16, 23, 42};
```

Compléter :

```c
printf("%d",    ?    );
```

pour afficher : 3 `tab[0]`

pour afficher : 268 `tab[2]`

pour afficher : 31 `tab[5] + tab[6]`

Accès aux données

```c
int tab[6] = {4, 8, 15, 16, 23, 42}, i = 3;
```

Quelle instruction pour obtenir les résultats suivants ?

Remplacer 23 par 0 :

```c
tab[4] = 0;
```

Stocker une saisie dans la 3ème cellule :

```c
scanf("%d", &tab[2]);
```

Intervertir la première et la dernière cellule :

```c
i = tab[0];
tab[0] = tab[5];
tab[5] = i;
```

Accès aux données

```
int tab[6] = {4, 8, 15, 16, 23, 42};
```

Additionner toutes les valeurs du tableau :

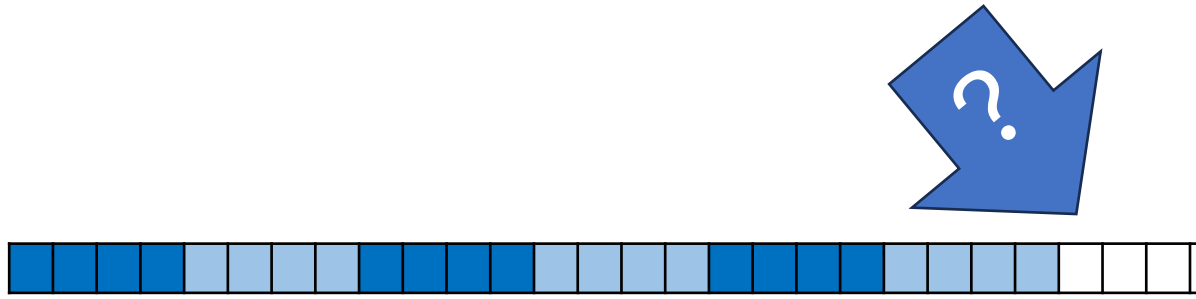
```
int somme = 0;

for (int i = 0; i < 6; i++) {

    somme = somme + tab[i];

}
```

Accès aux données

```
int tab[6] = {4, 8, 15, 16, 23, 42}, i = 3;
```

Qu'affiche :

```
printf("%d", tab[5]);     42
printf("%d", tab[i]);     16
printf("%d", tab[6]);     ?
printf("%d", tab);        ?
```



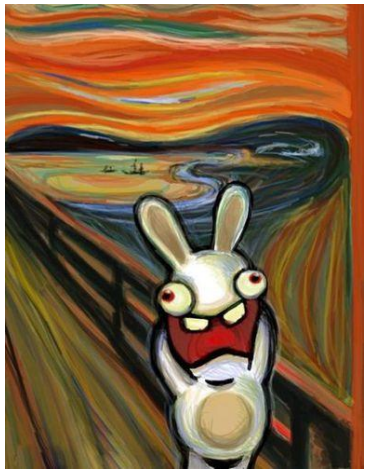
Afficher un tableau

Il faut afficher les éléments 1 à 1.

```c
int tab[6] = {4, 8, 15, 16, 23, 42};

for (int i = 0; i < 6; i++) {

    printf("%d\n", tab[i]);

}
```

Accès aux données

```
printf("%p", tab);          0000002eda5ffb14
```



Accès à la valeur stockée à l'indice `n` du tableau `tab` :

```
*(tab + n)
```

Accès aux données

```c
int tab[9] = {3, -7, 268, 4, 8, 15, 16, 23, 42};
```

Compléter :

```c
printf("%d",    ?    );
```

pour afficher : 3 `*tab`

pour afficher : 268 `*(tab + 2)`

pour afficher : 31 `*(tab + 5) + *(tab + 6)`

Accès aux données

```c
int tab[6] = {4, 8, 15, 16, 23, 42}, i = 3;
```

Quelle instruction pour obtenir les résultats suivants ?

Remplacer 23 par 0 :

```c
*(tab + 4) = 0;
```

Stocker une saisie dans la 3ème cellule :

```c
scanf("%d", tab + 2);
```

Intervertir la première et la dernière cellule :

```c
i = *tab;
*tab = *(tab + 5);
*(tab + 5) = i;
```

Accès aux données

```
int tab[6] = {4, 8, 15, 16, 23, 42};
```

Additionner toutes les valeurs du tableau :

```
int somme = 0;

for (int i = 0; i < 6; i++) {

    somme = somme + *(tab + i);

}
```

Accès aux données v2.0 !

```
int tab[6] = {4, 8, 15, 16, 23, 42};
```

Additionner toutes les valeurs du tableau :

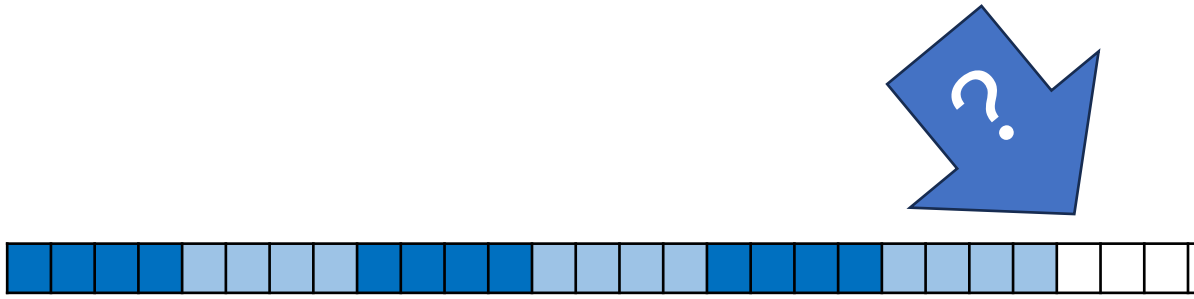
```
int somme = 0;

for (int * p = tab; p < tab + 6; p++) {

    somme = somme + *p;

}
```

Accès aux données

```
int tab[6] = {4, 8, 15, 16, 23, 42}, i = 3;
```

Qu'affiche :

```
printf("%d", *(tab + 5)); 42
printf("%d", *(tab + i)); 16
printf("%d", *(tab + 6)); ?
```



Afficher un tableau

Il faut afficher les éléments 1 à 1.

```c
int tab[6] = {4, 8, 15, 16, 23, 42};

for (int i = 0; i < 6; i++) {

    printf("%d\n", *(tab + i));

}
```

Afficher un tableau v2.0 !

Il faut afficher les éléments 1 à 1.

```c
int tab[6] = {4, 8, 15, 16, 23, 42};

for (int * p = tab; p < tab + 6; p++) {

    printf("%d\n", *p);

}
```