

La Programmation en langage C

Chapitre 6: les Tableaux

La cible : SMI S3

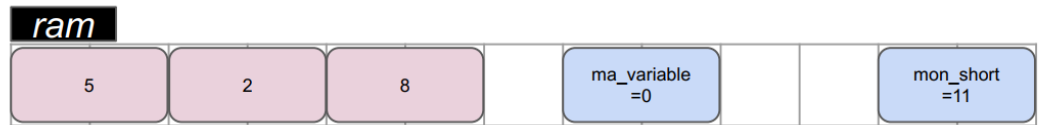
Pr S.ELFILALI

sanaa.elfilali@etu.univh2c.ma

Les tableaux

tableau de 3 cases:

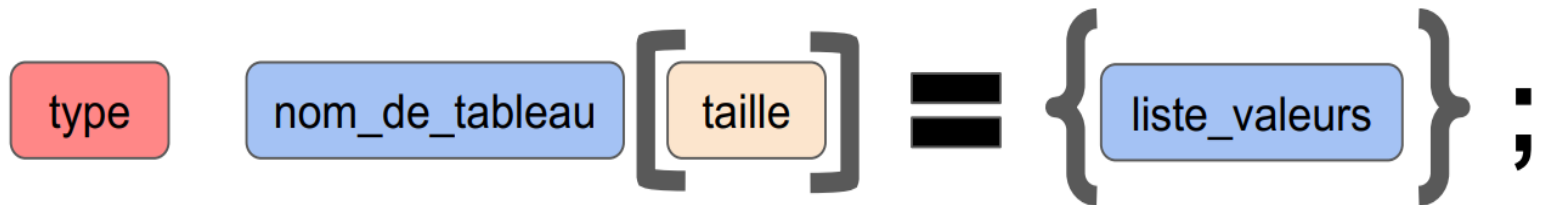
5	2	8
---	---	---



Toutes les cases d'un tableau sont contiguës en mémoire

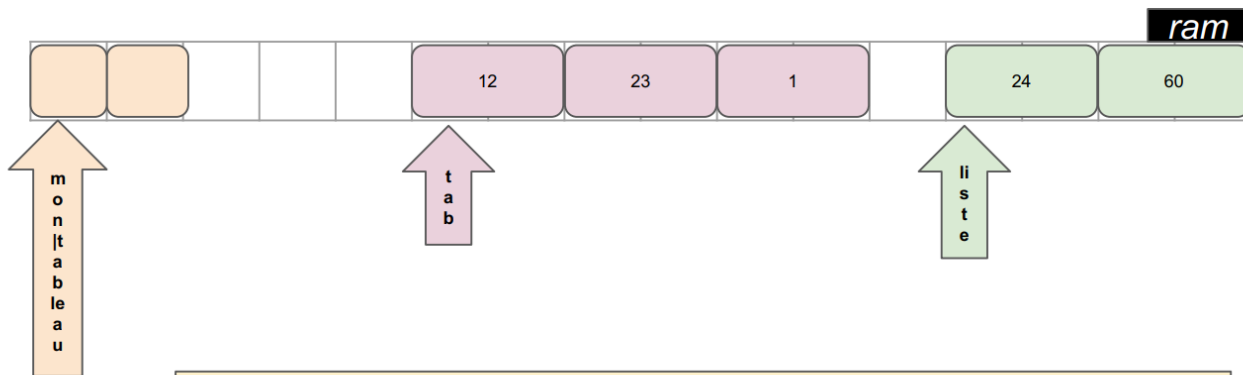
Toutes les cases d'un tableau sont du même type

Déclaration d'un tableau



Exemple:

```
char mon_tableau[2];  
short tab[3] = {12, 23, 1};  
short liste[] = {24, 60}
```



Taille du tableau fixé à la déclaration et ne peut pas varier

Déclaration d'un tableau

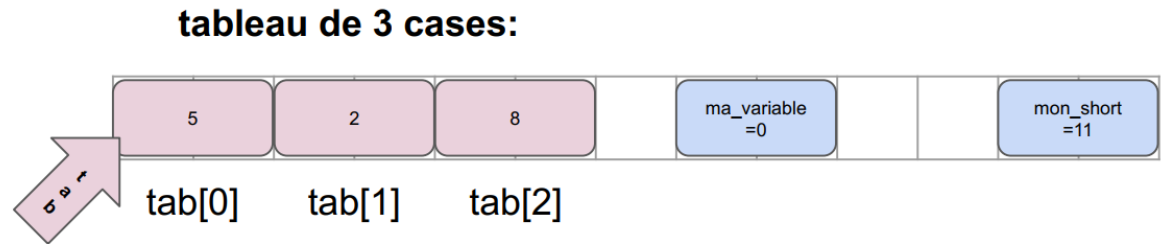
Exemple sur codeblocks

```
int main()
{
int tirage_loto_boules[5] = {1.2.3.4.5};
int tirage_loto_etoiles[] = {6.7};
char initiales_gagnant[2];
return 0;

}
```

Parcourir un tableau

ID:	0	1	2
	5	2	8



1ere case d'une tableau a pour ID 0

Toutes les cases d'un tableau sont du même type

Toutes les cases d'un tableau sont contiguës en mémoire

Parcourir un tableau

Exemple sur codeblocks

Initialisation du
tableau sans boucle

```
# define Taille_Tableau 3
int main()
{
//declarer le tableau
    int i;
    int mon_tableau[Taille_Tableau];
// init du tableau
    mon_tableau [0]= 0;
    mon_tableau [1]= 1;
    mon_tableau [2]= 2;

// affichage du tableau
    for(i=0;i<Taille_Tableau;i++)
    {
        printf("mon_tableau[%d] =%d \n",i, mon_tableau[i] );
    }
    return 0;
}
```

Initialisation du
tableau avec la boucle

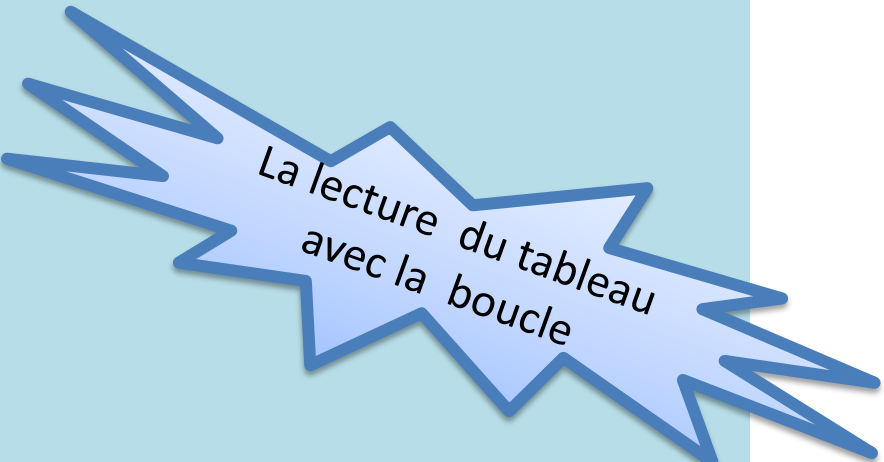
```
#include <stdlib.h>

# define Taille_Tableau 3
int main()
{
//declarer le tableau
    int i;
    int mon_tableau[Taille_Tableau];
// init du tableau
    for(i=0;i<Taille_Tableau;i++)
    {
        mon_tableau[i]=i;
    }
// affichage du tableau
    for(i=0;i<Taille_Tableau;i++)
    {
        printf("mon_tableau[%d] =%d \n",i, mon_tableau[i] );
    }
    return 0;
}
```

Parcourir un tableau

Exemple sur codeblocks

```
# define Taille_Tableau 3
int main()
{
//declarer le tableau
    int i;
    int mon_tableau[Taille_Tableau];
// lecture du tableau : la saisie
    for(i=0;i<Taille_Tableau;i++)
    {
        printf("> ");
        scanf("%d",&mon_tableau[i]);
    }
// affichage du tableau
    for(i=0;i<Taille_Tableau;i++)
    {
        printf("mon_tableau[%d] =%d \n",i, mon_tableau[i] );
    }
    return 0;
}
```

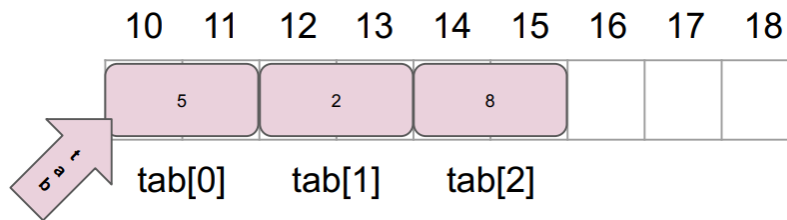


La lecture du tableau
avec la boucle

```
> 12
> 56
> 25
mon_tableau[0] =12
mon_tableau[1] =56
mon_tableau[2] =25

Process returned 0 (0x0)   execution time : 8.344 s
Press any key to continue.
```

Tableaux et pointeurs



`tab` (adresse du tableau)

`*tab` (contenue de la première case du tableau)

`*(tab+n-1)` (contenue de la la case `n` du tableau)

Exemple:

`tab[0] -> 5 // *(tab+0)->5`

`tab[1] -> 2 // *(tab+1)->2`

`tab[2] -> 8 // *(tab+2)->8`

`*tab + 1 => 5 + 1 => 6`

Tableaux et pointeurs

Exemple sur codeblocks

```
# define Taille_Tableau 3
int main()
{
//declarer le tableau
    int i;
    int mon_tableau[Taille_Tableau];
// lecture du tableau : la saisie
    for(i=0;i<Taille_Tableau;i++)
    {
        printf("> ");
        scanf("%d", (mon_tableau+i));
    }
// affichage du tableau
    for(i=0;i<Taille_Tableau;i++)
    {
        printf("mon_tableau[%d] =%d \n", i, *(mon_tableau+i));
    }
    return 0;
}
```

```
> 25
> 14
> 19
mon_tableau[0] =25
mon_tableau[1] =14
mon_tableau[2] =19
```

Tableaux et adresses : Exercice

- 1- Créez un tableau de 5 caractères et initialisez le
- 2- Pour chaque cases, afficher l'index, le contenu et l'adresse
- 3- Faire la même chose mais en utilisant la

Aide :

- 1- `type nom[taille] = {liste_valeurs};`
- 2- Il faut utiliser `%p` pour afficher une adresse mémoire dans un `printf`.
- 3- `tab[i] = *(tab+i)`

```
tab[0] = C (0060FEF9)
tab[1] = O (0060FEFA)
tab[2] = D (0060FEFB)
tab[3] = E (0060FEFC)
tab[4] = R (0060FEFD)
```

Tableaux et adresses : Exercice

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
    char tab[] ={'C','O','D','E','R'};
    int i;
    // annotation simple

    for(i=0;i<5;i++)
    {
        printf("tab[%d] = %c (%p)\n", i,tab[i], &tab[i]);
    }
    return 0;
}
```

```
tab[0] = C (0019FF34)
tab[1] = O (0019FF35)
tab[2] = D (0019FF36)
tab[3] = E (0019FF37)
tab[4] = R (0019FF38)
```

Tableaux et adresses : Exercice

Annotation simple

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
    char tab[] = {'C','O','D','E','R'};
    int i;
    // annotation simple

    for(i=0;i<5;i++)
    {
        printf("tab[%d] = %c (%p)\n", i, tab[i], &tab[i]);
    }
    return 0;
}
```

```
tab[0] = C (0019FF34)
tab[1] = O (0019FF35)
tab[2] = D (0019FF36)
tab[3] = E (0019FF37)
tab[4] = R (0019FF38)
```

Tableaux et adresses : Exercice

Annotation pointeur

```
#include <stdio.h>
#include <stdlib.h>

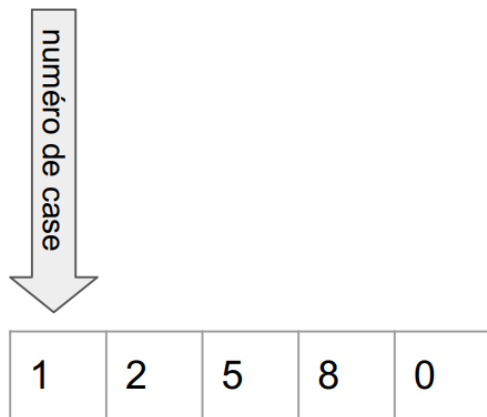
int main()
{
    char tab[] = {'C','O','D','E','R'};
    int i;

    // Annotation Pointeur
    for(i=0;i<5;i++)
    {
        printf("(tab + %d) = %c (%p)\n", i,*(tab+i),(tab+i));
    }

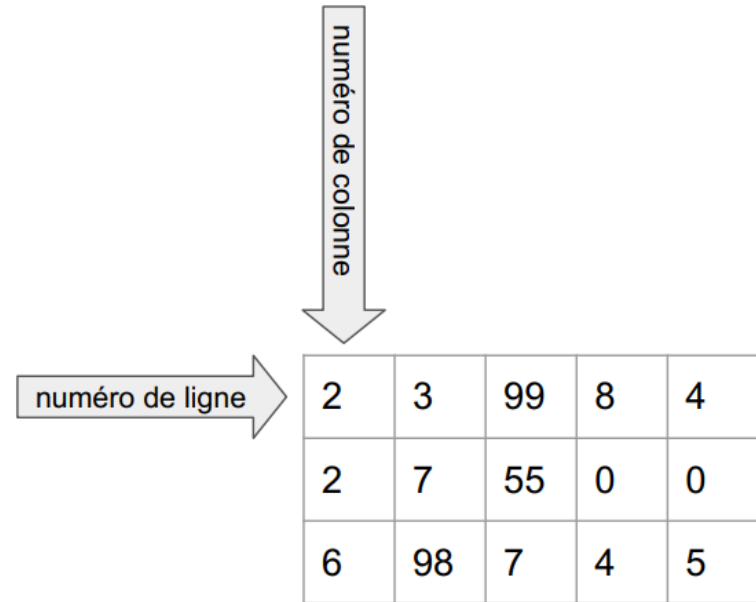
    return 0;
}
```

```
*(tab + 0) = C (0019FF34)
*(tab + 1) = O (0019FF35)
*(tab + 2) = D (0019FF36)
*(tab + 3) = E (0019FF37)
*(tab + 4) = R (0019FF38)
```

Les tableaux multidimensionnels

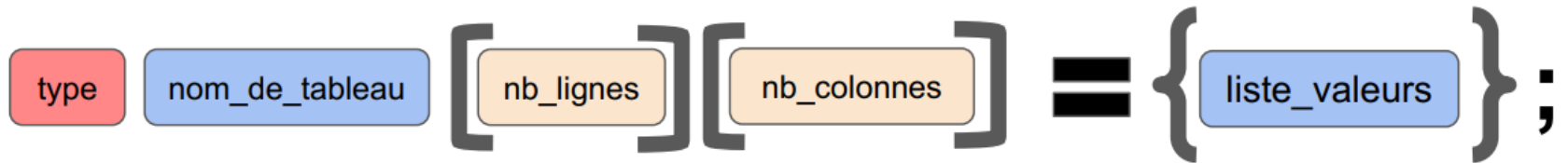


1D



2D

Déclaration tableaux 2D

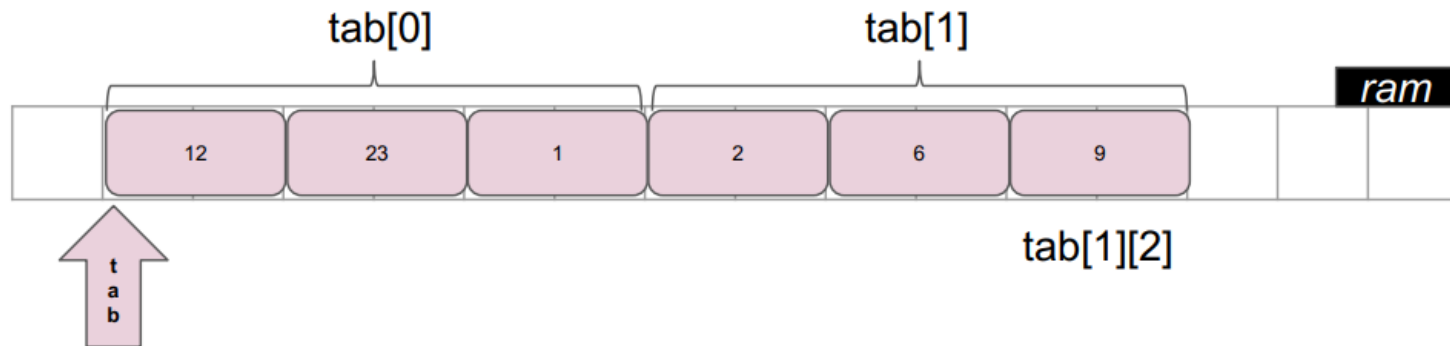


Exemple:

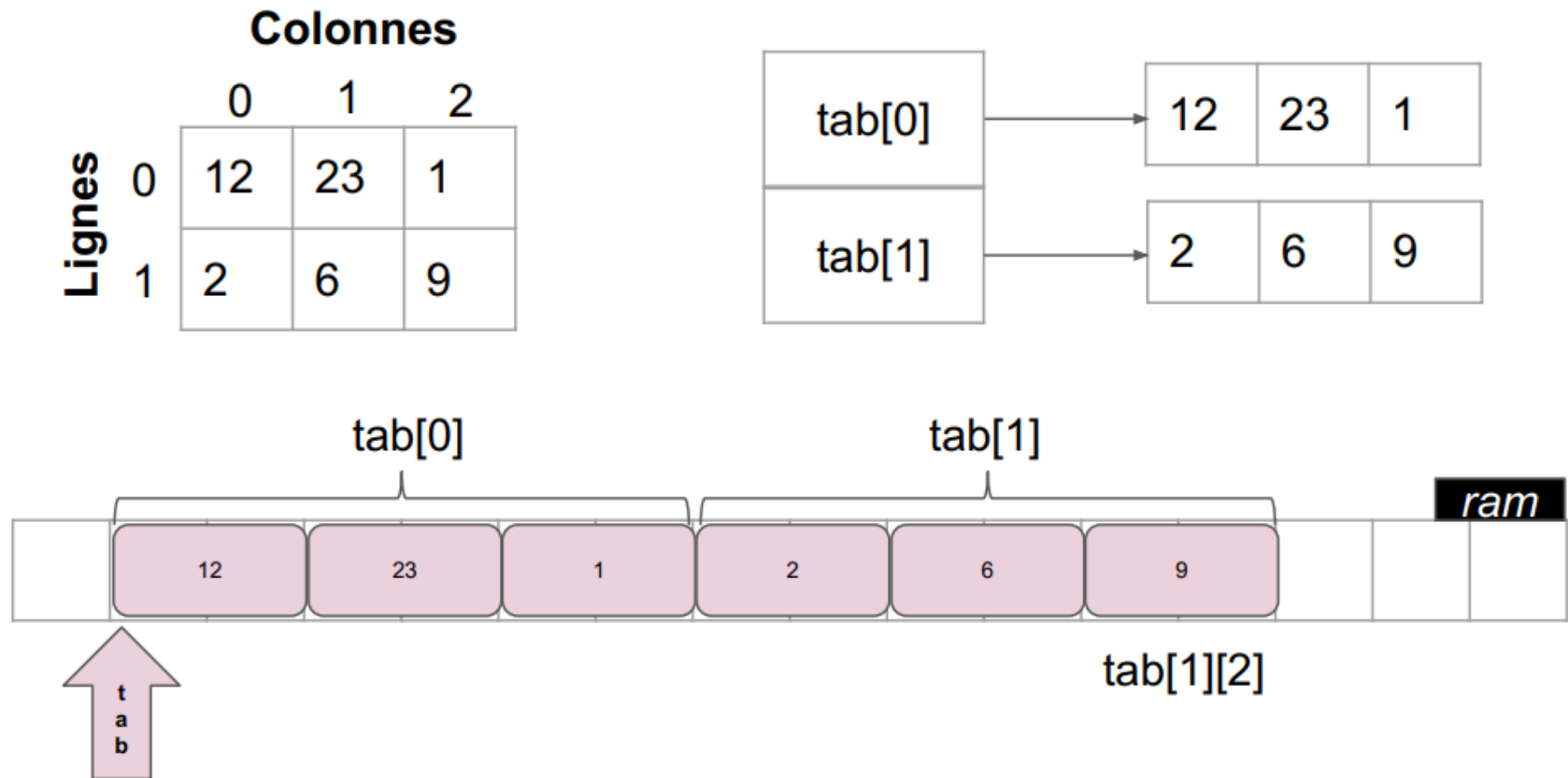
```
short tab[2][3] = { {12, 23, 1}, {2, 6, 9} };
```

```
tab[1][2] //retourne la valeur 9
```

		Colonnes		
		0	1	2
Lignes	0	12	23	1
	1	2	6	9



Tableaux 2D = tableau de tableaux



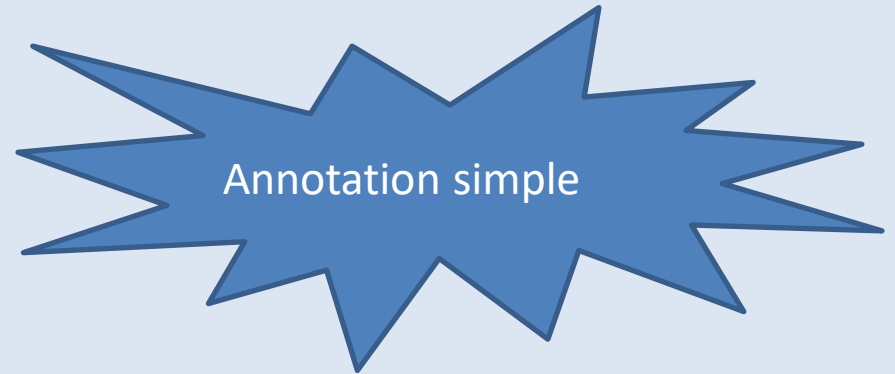
Les tableaux multidimensionnels

Exercices

Exemple sur codeblocks : créer un tableau à deux dimensions

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
    int ligne;
    int colonne;
    int tab[2][3]={{1,2,3},{4,5,6}};
    //Annotation simple
    for(ligne=0;ligne<2;ligne++)
    {
        for (colonne=0;colonne<3;colonne++)
        {
            printf("tab[%d][%d] = %d\n",ligne,colonne,tab[ligne][colonne]);
        }
    }
    return 0;
}
```



Les tableaux multidimensionnels

Exercices

Exemple sur codeblocks : créer un tableau à deux dimensions

```
#include <stdio.h>
#include <stdlib.h>
```

```
int main()
```

```
{
```

```
    int ligne;
```

```
    int colonne;
```

```
    int* p_ligne;
```

```
    int tab[2][3]={1,2,3},{4,5,6}};
```

```
    for(ligne=0;ligne<2;ligne++)
```

```
    {
```

```
        p_ligne = *(tab+ligne);
```

```
        for (colonne=0;colonne<3;colonne++)
```

```
        {
```

```
            printf("tab[%d][%d] = %d\n",ligne,colonne,*(p_ligne +colonne));
```

```
        }
```

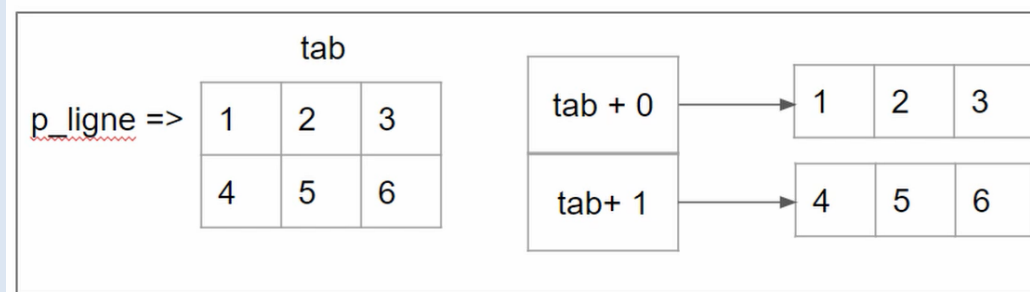
```
    }
```

```
    return 0;
```

```
}
```

Annotation pointeur

```
tab[0][0] = 1
tab[0][1] = 2
tab[0][2] = 3
tab[1][0] = 4
tab[1][1] = 5
tab[1][2] = 6
```



```
*(*(tab+ligne)) = *(p_ligne)
```

Mini projet

Travaux Pratiques Programmation en langage C

SMI_ S3

TP : les Tableaux
MASTERMIND



Pr S.ELFIALI
2020-2021

Mini projet

Nous allons créer un petit jeu de type mastermind. Pour ceux qui ne connaissent pas, ce jeu de société se joue à deux, une des personnes crée un code secret composé de 4 couleurs. Puis le 2em joueur doit découvrir cette combinaison. Vous pouvez retrouver les règles du jeu ici: <https://www.regles-de-jeux.com/regle-du-mastermind/>

Nous allons simplifier un peu les règles pour ce TP.

- Nous allons utiliser les initiales des couleurs pour lire la saisie utilisateur et stocker le code secret parmi ('R','V','B','J','O')
- Le code sera constitué de 4 couleurs.
- Le code ne sera pas choisi par un joueur mais écrit en dure dans le programme (ex {'R','V','B','J'})
- Le nombre maximal de tentatives via une constante (ex: 3)

```
...
Donnez un code de 4 couleurs differentes et sans espaces parmi : {'R', 'V', 'B', 'J', 'O'}
> RVJO
Tentative 1/3
Couleurs mal placees: 1
Couleurs bien placees: 2
```