Les Pointeurs

1. Créez et initialisez des pointeurs

- **Faites afficher l'adresse d'une variable**

Pour afficher l'adresse de la variable, on doit :

1. Utiliser le symbole %p (le p du mot « pointeur ») dans le printf.
2. Envoyer à la fonction printf non pas la variable age , mais son adresse… Et pour faire cela, vous devez mettre le symbole & devant la variable age.

```
printf("L'adresse de la variable age est : %p", &age);
```

Si vous remplacez %p par %d , vous obtiendrez un nombre décimal que vous connaissez.

Je veux vous faire retenir ceci :

1. age désigne la valeur de la variable ;
2. &age désigne l'adresse de la variable.

- **Créez un pointeur et donnez-lui une valeur par défaut**

Pour créer une variable de type pointeur, on rajoute le symbole * devant le nom de la variable.

Int *monPointeur;

Notez qu'on peut aussi écrire **int* monPointeur;** . Cela revient exactement au même.

si vous voulez déclarer plusieurs pointeurs sur la même ligne, vous serez obligé de mettre l'étoile devant le nom : **int *pointeur1, *pointeur2, *pointeur3;**

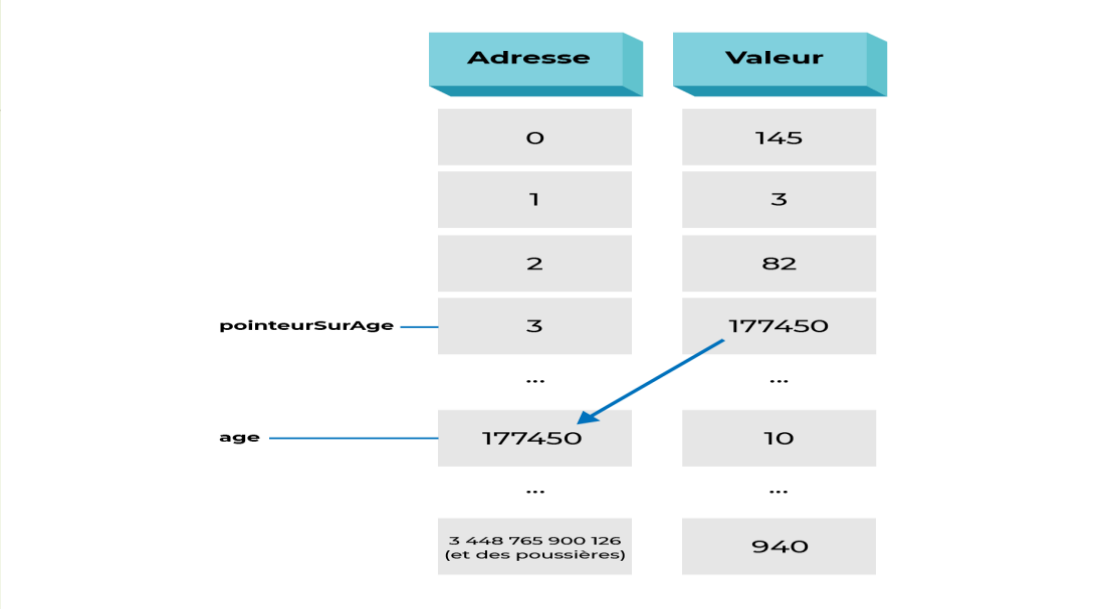
Pour initialiser un pointeur, c'est-à-dire lui donner une valeur par défaut, on n'utilise généralement pas le nombre 0 mais le mot-clé NULL (veillez à l'écrire en majuscules) **:**

Int *monPointeur = NULL;

```
int age = 10;
int *pointeurSurAge = &age;
```

1. La première ligne signifie : "Créer une variable de type int dont la valeur vaut 10".
2. La seconde ligne signifie : "Créer une variable de type pointeur dont la valeur vaut l'adresse de la variable age ".

Le schéma suivant résume ce qu'il s'est passé dans la mémoire :



1. la variable **age** a été placée à l'adresse 177450 (vous voyez d'ailleurs que sa valeur est 10) ;
2. Et le pointeur pointeurSurAge a été placé à l'adresse 3 (c'est tout à fait le fruit du hasard).

Maintenant, on a un pointeurSurAge qui contient l'adresse de la variable age.

Essayons de voir ce que contient le pointeur à l'aide d'un printf :

```
int age = 10;
int *pointeurSurAge = &age;

printf("%d", pointeurSurAge);
```

On demande la valeur de **pointeurSurAge** , et sa valeur, c'est l'adresse de la variable age (177450).

Comment faire pour demander à avoir la valeur de la variable se trouvant à l'adresse indiquée dans pointeurSurAge ?

Il faut placer le symbole * devant le nom du pointeur :

```
int age = 10;
int *pointeurSurAge = &age;

printf("%d", *pointeurSurAge);
```

Remarque: En plaçant le symbole * devant le nom du pointeur, on accède à la valeur de la variable age. Si au contraire on avait utilisé le symbole & devant le nom du pointeur, on aurait obtenu l'adresse à laquelle se trouve le pointeur.

2. **Envoyez un pointeur à une fonction**

Le gros intérêt des pointeurs (mais ce n'est pas le seul) est qu'on peut les envoyer à des fonctions pour qu'ils modifient directement une variable en mémoire,

- Envoyez un pointeur dans la fonction **triplePointeur**

```c
void triplePointeur(int *pointeurSurNombre);

int main(int argc, char *argv[])
{
    int nombre = 5;

    triplePointeur(&nombre); // On envoie l'adresse de nombre à la fonction
    printf("%d", nombre); // On affiche la variable nombre. La fonction a directement modifié la valeur de la variable car elle connaissait son adresse

    return 0;
}

void triplePointeur(int *pointeurSurNombre)
{
    *pointeurSurNombre *= 3; // On multiplie par 3 la valeur de nombre
}
```

- Alternative : ajoutez un pointeur dans la fonction main

```c
void triplePointeur(int *pointeurSurNombre);

int main(int argc, char *argv[])
{
    int nombre = 5;
    int *pointeur = &nombre; // pointeur prend l'adresse de nombre

    triplePointeur(pointeur); // On envoie pointeur (l'adresse de nombre) à la fonction
    printf("%d", *pointeur); // On affiche la valeur de nombre avec *pointeur

    return 0;
}

void triplePointeur(int *pointeurSurNombre)
{
    *pointeurSurNombre *= 3; // On multiplie par 3 la valeur de nombre
}
```

Remarque: Dans le chapitre "Déclarez des variables", nous avons vu comment récupérer la saisie d'un utilisateur, et nous avons utilisé des **pointeurs** sans vraiment le savoir.

C'était en fait en appelant la fonction **scanf** . En effet, cette fonction lit ce que l'utilisateur a entré au clavier et renvoie le résultat.

Pour que la fonction puisse modifier directement le contenu de votre variable afin d'y placer la valeur tapée au clavier, elle a besoin de l'adresse de la variable :

```
int nombre = 0;
scanf("%d", &nombre);
```

La fonction travaille avec un pointeur sur la variable **nombre**, et peut ainsi modifier directement le contenu de nombre.

Comme on vient de le voir, on pourrait créer un pointeur qu'on enverrait à la fonction **scanf** :

```
int nombre = 0;
int *pointeur = &nombre;
scanf("%d", pointeur);
```

- **Tableau**

Notez que **tableau** est un pointeur, on peut utiliser le symbole * pour connaître la première valeur **:**

```
int tableau[4];

printf("%d", *tableau);
```

Il est aussi possible d'obtenir la valeur de la seconde case avec *(tableau + 1) (adresse de tableau + 1). Les deux lignes suivantes sont donc identiques :

```
tableau[1] // Renvoie la valeur de la seconde case (la première case étant 0)
*(tableau + 1) // Identique : renvoie la valeur contenue dans la seconde case
```

Remarque:

En clair, quand vous écrivez **tableau[0]** , vous demandez la valeur qui se trouve à l'adresse **tableau + 0** case (c'est-à-dire, dans notre cas, la valeur 10 associée à l'adresse 1600). Si vous écrivez **tableau[1]** , vous demandez la valeur se trouvant à l'adresse **tableau + 1** case (c'est-à-dire, dans notre cas, la valeur 23 associée à l'adresse 1601).