

# La Programmation en langage C

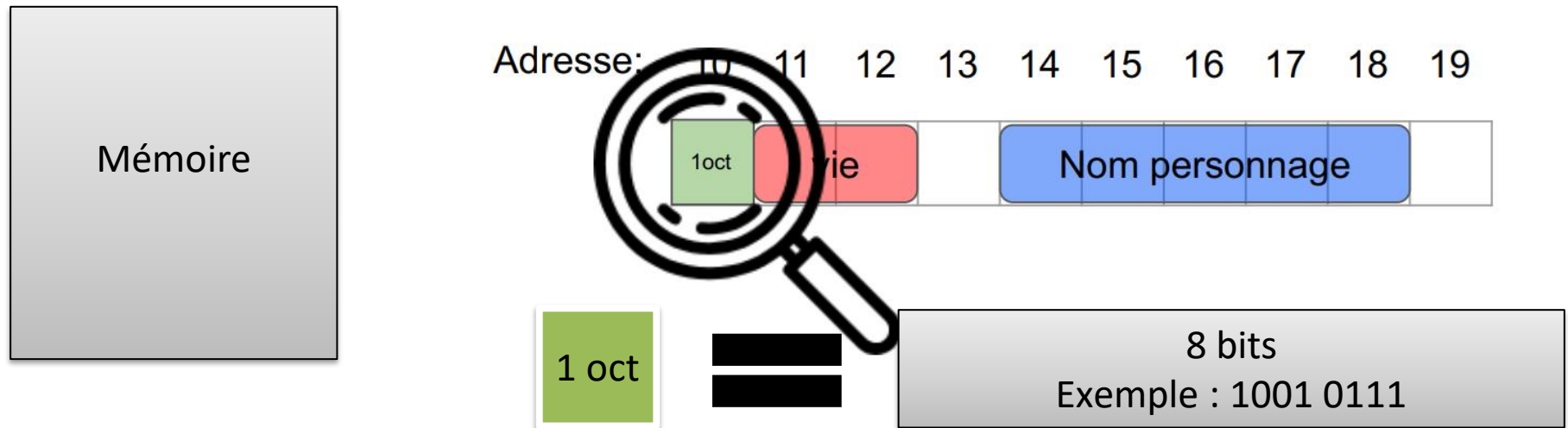
## Chapitre 4: les Pointeurs

### ***La cible : SMI S3***

Pr S.ELFILALI  
sanaa.elfilali@etu.univh2c.ma

# Rappel :

## Gestion de la mémoire et variable



- Le programme est exécuté depuis la RAM
- La mémoire est composée de cases continues et chacune de ces cases est représentée par un octet
- Un octet c'est 8 bits
- Ces cases sont identifiées de manières uniques à l'aide de leurs adresses
- Une variable est une boîte qui occupe un nombre entier de ces cases et qui permet de stocker les données en binaire
- La taille et l'interprétation dépend de leur type de la variable
- On peut accéder au contenu de la variable à l'aide de son nom qui sert comme label pour accéder à la variable

# Notion de pointeur

- Un *pointeur* est une variable contenant l'adresse d'une autre variable d'un type donné.
- Il permet donc d'accéder **indirectement** à une variable.



Adresse: 10 11 12 13 14 15 16 17 18 19



# Adresses et variables

Adresse: 10 11 12 13 14 15 16 17 18 19



```
int age_utilisateur = 25;
```

```
printf("Age utilisateur: %d\n", age_utilisateur);  
printf("Adresse: %d\n", &age_utilisateur);  
printf("Adresse: %p\n", &age_utilisateur);
```

```
Age utilisateur: 25  
Adresse: 10  
Adresse: 0x0A
```

# Adresses et variables

Exemple sur codeblocks :

Afficher le contenu et l'adresse d'une variable en décimale et en hexadécimale

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

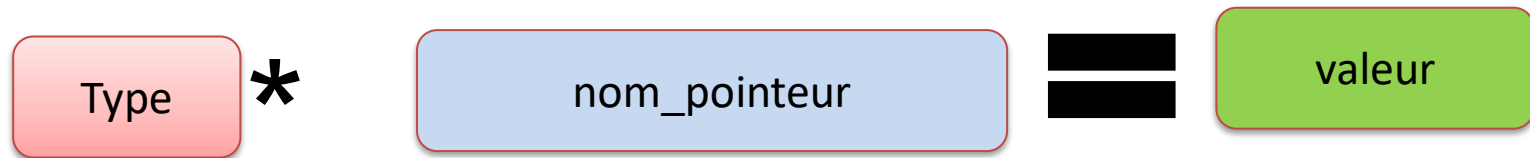
int main()
{
    int mon_int=10;
    printf("\nmon int = %d",mon_int );
    printf("\nAdresse de mon_int = %d",&mon_int);
    printf("\nAdresse de mon_int = %p",&mon_int);
    return 0;
}
```

```
mon int = 10
Adresse de mon_int = 1703736
Adresse de mon_int = 0019FF38
```

# Représentation hexadécimale

Représentation	Alphabet	Exemple
<b>Décimale (10)</b>	0, 1, 2, 3, 4, 5, 6, 7, 8, 9	12
<b>Binaire (2)</b>	0, 1	1100
<b>Hexadécimale (16)</b>	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F	C

# Déclaration d'un pointeur



## Exemple :

Char\* p\_char = 0;

Int\* p\_int = NULL;



# Déclaration d'un pointeur

- Sur codeblocks
- Reprendre l'exemple précédant et déclarer des pointeurs sur les variables déjà déclarées



# Utilisation des pointeurs

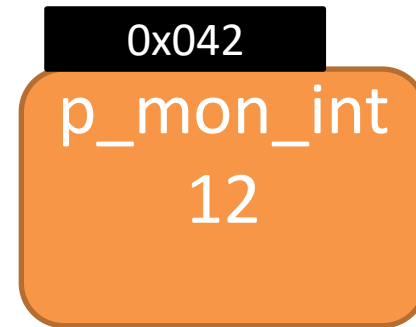


**`mon_int` (contenue de `mon_int`)**

-> 12

**`&mon_int` (adresse de `mon_int`)**

-> 0x007



**`&p_mon_int` (adresse de `p_mon_int`)**

-> 0x042

**`p_mon_int` (contenue de `p_mon_int`)**

-> 0x007

**`*p_mon_int` (contenue de 0x007)**

-> 12

# Dangers des pointeurs

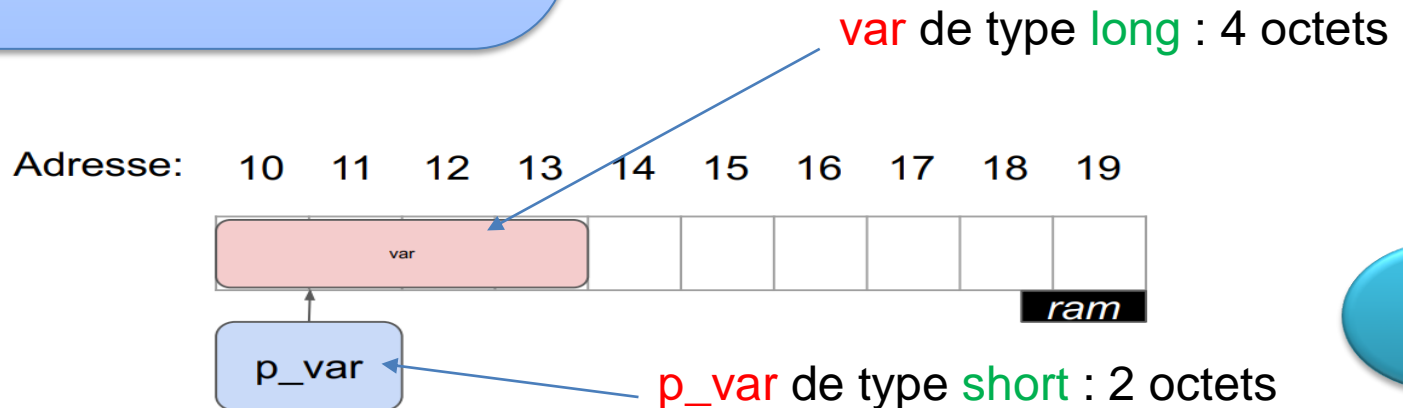
## 1. Problème de typage :

Les pointeurs doivent correspondre aux types que l'on souhaite pointer

```
int main()
{
    long var = 65321;
    short* p_var = &var;

    printf("ma variable = %ld \n", var);
    printf("ma variable = %ld \n", *p_var);
}
```

ma variable = 65321  
ma variable = -215



# Dangers des pointeurs

## 2. Pointeur NULL:

On essaie d'afficher le contenu des variables pointées par NULL

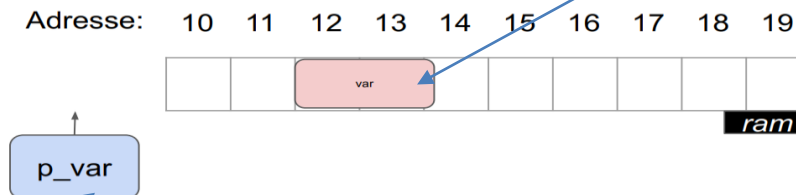
```
int main()
{
    short var = 123;
    short* p_var = NULL; // short* p_var = 0;

    printf("ma variable = %ld \n", var);
    printf("ma variable = %ld \n", *p_var);
}
```

ma variable = 123

Crash !

var de type short : 2 octets



p\_var de type short : ne point null part

# Dangers des pointeurs

## 2. Pointeur NULL:

On essaie d'afficher le contenu des variables pointées par NULL

Pour que le programme ne crashe pas :

```
int main()
{
    short var =123;
    short* p_var =NULL; // short* p_var = 0;
    printf("ma variable = %ld \n",var);
    if(p_var ==NULL)
    {
        printf("\n pointeur NULL !!!\n ");
    }
    else
    {
        printf("ma variable = %ld \n",*p_var);
    }
    return 0;
}
```

ma variable = 123

pointeur NULL !!!

# Dangers des pointeurs

## 2. Pointeur NULL:

On essaie d'afficher le contenu des variables pointées par NULL

Pour que le programme ne crashe pas :

```
int main()
{
    short var =123;
    short* p_var =&var;
    printf("ma variable = %ld \n",var);
    if(p_var ==NULL)
    {
        printf("\n pointeur NULL !!!\n ");
    }
    else
    {
        printf("ma variable = %ld \n",*p_var);
    }
    return 0;
}
```

ma variable = 123  
ma variable = 123

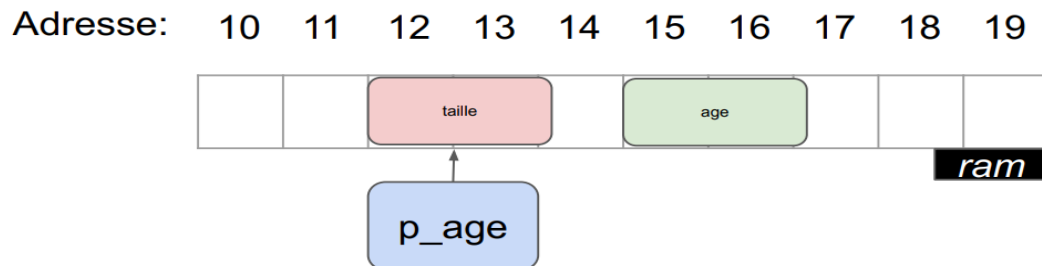
# Dangers des pointeurs

## 3. Jardinage :

On essaie de pointer sur une autre variable valide , problème difficile à trouver !!!

```
int main()
{
    int age=25;
    int taille =165;
    int* p_int =0;
    p_int=&taille;
    printf("\n donner votre age : ");
    scanf("%d",p_int);
    printf("votre age est : %d ans ",age);
    return 0;
}
```

donner votre age : 30  
votre age est : 25 ans



# Tester nos connaissances

- Jeux kahoot
- <https://create.kahoot.it/details/pointeur/823599f7-6485-4658-825d-523c3bec364c>

# EXERCICE

1. Créer une variable de type char
2. Afficher le contenu de la variable, sa taille en mémoire et son adresse

```
Informations sur ma variable:  
type: char  
taille: 1 octet  
contenu: A  
adresse: 0060FF0F
```

Aide : pour afficher une adresse il faut utiliser

**%p**



- `int main()`
- `{`
- `char lettre = 'A';`
- `printf("\n Informations sur ma vaiabale : \n ");`
- `printf("\t type : char \n ");`
- `printf("\t taille : %d \n ", sizeof(lettre));`
- `printf("\t contenu : %c \n ",lettre);`
- `printf("\t adresse : %p \n",&lettre );`
- `return 0;`
- `}`

```
Informations sur ma vaiabale :
    type       : char
    taille     : 1
    contenu    : A
    adresse    : 0019FF3B

Process returned 0 (0x0)   execution time : 0.206 s
Press any key to continue.
```

FIN