

# LA PROGRAMMATION EN LANGAGE C

## INTRODUCTION SÉANCE 2

**SMI/ S3**

Pr S.ELFILALI

Une variable  
les variables typées  
Les caractéristiques d'une variable  
Les types de variable  
Les types numériques entiers  
Les types rationnels  
Déclaration d'une variable  
Initialisation  
Déclaration d'une constante  
Identificateurs  
Incrémentatation / décrémentation  
Préfixe et postfixe  
Conversion des types

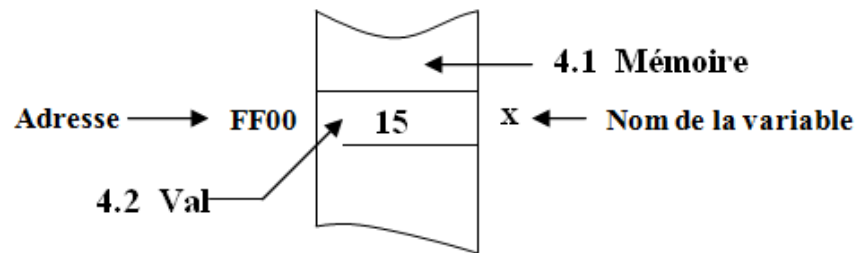
# UNE VARIABLE ?

Une variable est un objet

Repéré par son  
nom

Pouvant  
contenir des  
données

Pourront être  
modifiées lors de  
l'exécution du  
programme.



# LES VARIABLES TYPÉES

Les variables en langage C sont typées,

les données possèdent un type

elles sont stockées dans la mémoire

occupent un nombre d'octets

dépendant du type de donnée stockée.

# LES CARACTÉRISTIQUES D'UNE VARIABLE

## Nom

- Unique pour chaque variable
- Commence toujours par une lettre
- Différenciation minuscule-majuscule

## type

- Conditionne le format de la variable en mémoire
- Peut être soit un type standard ou un type utilisateur

## Valeur

- Peut évoluer pendant l'exécution
- initialisation grâce à l'opérateur d'affectation

## Adresse

# LES TYPES DE DONNÉES

## Types de variable

<b>Char</b>	→	caractères
<b>Int</b>	→	entiers
<b>short [int]</b>	→	entiers courts
<b>long [int]</b>	→	entiers longs
<b>Float</b>	→	nombres décimaux
<b>Double</b>	→	nombres décimaux de précision supérieure
<b>long double</b>	→	nombres décimaux encore plus précis
<b>unsigned int</b>	→	entier non signé

# TYPES NUMÉRIQUES ENTIERS

Type de donnée	Taille (en octets)	Plage de valeurs acceptée
char	1	-128 à 127
unsigned char	1	0 à 255
short int	2	-32768 à 32767
unsigned short int	2	0 à 65535
int	2	-32768 à 32767
unsigned int	2	0 à 65535
long int	4	-2 147 483 648 à 2 147 483 647

# LES TYPES RATIONNELS

Type de donnée	Taille (en octets)	Plage de valeurs acceptée
float	4	$-3.4 \times 10^{-38}$ à $3.4 \times 10^{38}$
double	8	$1.7 \times 10^{-308}$ à $1.7 \times 10^{308}$
long double	10	$3.4 \times 10^{-4932}$ à $3.4 \times 10^{4932}$



# DÉCLARATION D'UNE VARIABLE

Les déclarations permettent de :

réserver de l'espace en mémoire  
pour les variables.

```
Type  nom_de_la_variable [= valeur] ;
```

# DÉCLARATION D'UNE VARIABLE

Type    nom\_de\_la\_variable [= valeur] ;

## Exemple:

int nb;

float pi = 3.14;

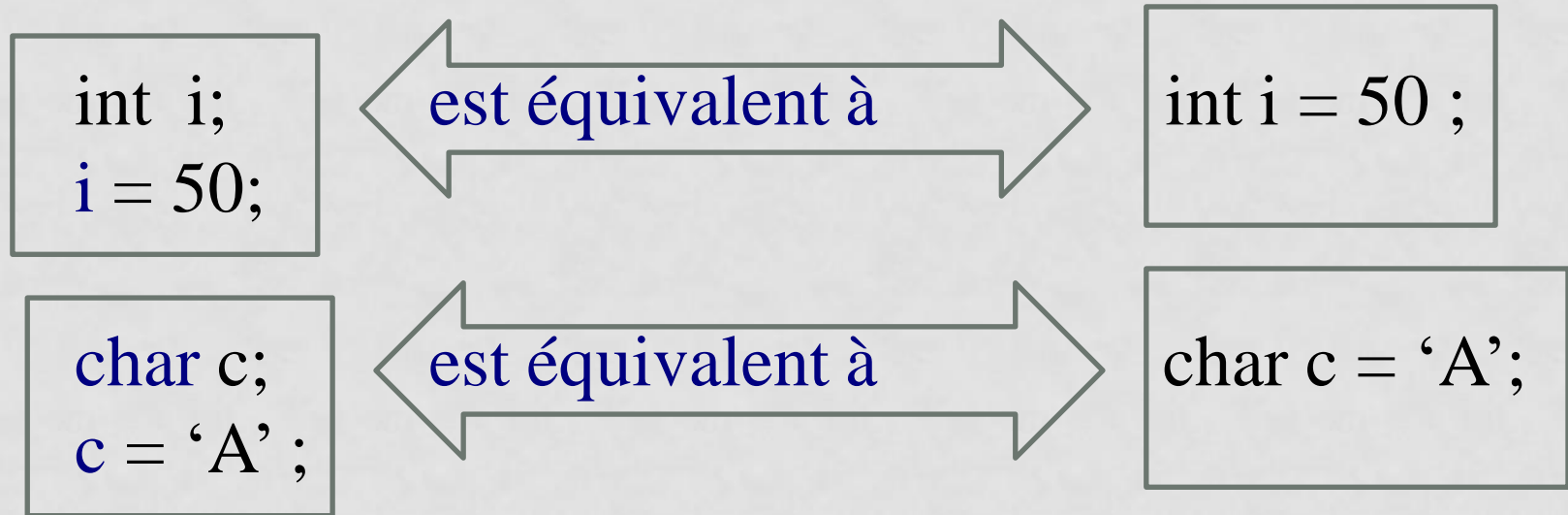
char c = 'a';

long i,j,k;

double r = 6.2879821365;

# LES INITIALISATIONS

C permet l'initialisation des variables dans la zone de déclaration



# LES DECLARATIONS DE CONSTANTES

## 1ere méthode :

définition d'un symbole à l'aide de la **directive** de compilation **#define**.



Le compilateur ne réserve pas de place en mémoire

Syntaxe :      **#define** **identificateur** texte(valeur)

## Exemple:

```
#define      PI  3.14159
void main()
{
    float perimetre, rayon = 8.7;
    perimetre = 2*rayon*PI;
    ....
}
```

# LES DECLARATIONS DE CONSTANTES

2ème méthode :

**déclaration** d'une variable, dont la valeur sera constante pour tout le programme.

➤ **Le compilateur réserve de la place en mémoire**

## Exemple

```
void main()
{
    const float PI = 3.14159;
    const int JOURS = 5;
    float perimetre, rayon = 8.7;
    perimetre = 2*rayon*PI;
    ....
    JOURS = 3;
    ....
}
```

*Le compilateur réserve de la place en mémoire (ici 4 octets).*

*/\*ERREUR!\*/ On ne peut changer la valeur d'une const.*

# IDENTIFICATEURS

**Les identificateurs nomment les objets C (fonctions, variables ... )**

- **C'est une suite de lettres ou de chiffres.**
- **Le premier caractère est obligatoirement une lettre.**
- **Le caractère \_ (souligné) est considéré comme une lettre.**

# IDENTIFICATEURS

**Le C distingue les minuscules des majuscules.**

## **Exemples :**

**abc, Abc, ABC sont des identificateurs valides et tous différents.**

## **Identificateurs valides :**

<b>xx</b>	<b>z2</b>	<b>somme_5</b>	<b>_position</b>
<b>Noms</b>	<b>surface</b>	<b>fin_de_fichier</b>	<b>VECTEUR</b>

## **Identificateurs invalides :**

**5eme**  
**x#z**  
**no-commande**  
**taux change**

**commence par un chiffre**  
**caractère non autorisé (#)**  
**caractère non autorisé (-)**  
**caractère non autorisé (espace)**

# IDENTIFICATEURS

**Un identificateur ne peut pas être un mot réservé du langage :**

auto	double	int	struct
break	else	long	switch
case	enum	register	typedef
char	extern	return	union
const	float	short	unsigned
continue	for	signed	void
default	goto	sizeof	volatile
do	if	static	while

**RQ : Les mots réservés du langage C doivent être écrits en minuscules.**



# INCRÉMENT ET DÉCREMENT

- C a deux opérateurs spéciaux pour incrémenter (ajouter 1) et décrémenter (retirer 1) des variables entières
  - ++      **increment**    :     $i++$  ou  $++i$  est équivalent à  $i += 1$  ou  $i = i + 1$
  - --      **decrement**
- Ces opérateurs peuvent être :
  - préfixés (avant la variable)
  - postfixés (après)

**“i” vaudra 6**

**“j” vaudra 3**

**“i” vaudra 7**

```
int  i = 5, j = 4;  
    i++;  
    --j;  
    ++i;
```

# PRÉFIXE ET POSTFIXE

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

```
int main(void)  
{
```

```
    int    i, j = 5;
```

```
    i = ++j;  
    printf("i=%d, j=%d\n", i, j);
```

```
    j = 5;  
    i = j++;
```

```
    printf("i=%d, j=%d\n", i, j);
```

```
    return 0;
```

```
}
```

équivalent à:

1. j++;
2. i = j;

équivalent à:

1. i = j;
2. j++;

i=6, j=6  
i=5, j=6

