

Les types

Types de base

Il existe 4 types de base

- Caractere **char**
- Entier **int**
- Réel **float**
- Reel double precision **double**

Le type d'une donnée determine

- La place memoire (sizeof())
- Les operations legales
- Les bornes

Types de base

Nom	Taille	Constante	Min	Max
char	1 octet	'a', 97	-128 (SCHAR_MIN)	+127 (SCHAR_MAX)
unsigned char	1 octet	'a', 97	0	255 (UCHAR_MAX)
short	2 octets	-125, 312	-32768 (SHRT_MIN)	32767 (SHRT_MAX)
unsigned short	2 octets	2730	0	65535 (USHRT_MAX)
int	4 octets (parfois 2 ou 8)	-3000, 3213	-2 milliards env. (INT_MIN)	+2 milliards env. (INT_MAX)
unsigned int	4 octets (parfois 2 ou 8)	31240	0	4 milliards env. (UINT_MAX)
long	4 octets (parfois 8 ou 16)	31671	-2 milliards env. (INT_MIN)	+2 milliards env. (INT_MAX)
unsigned long	4 octets (parfois 8 ou 16)	24301	0	4 milliards env. (UINT_MAX)
float	4 octets	21.3F, 3.12e4F	-3.4e38 env. (-FLT_MAX) 1.17e-38 (FLT_MIN)	+3.4e38 env. (FLT_MAX)
double	8 octets	12.4, 125.2e5	-1.79e308 env. (-DBL_MAX) 2.22e-308 env. (DBL_MIN)	1.79e308 (DBL_MAX)

char

Là encore, deux types, codés sur 8 bits.

- char (-128 à 127)
- unsigned char (0 à 255)

Exemple: 'a' 'c' '\$' '\n' '\t' '\007'

les caractères imprimables sont toujours positifs.

le code ASCII de 'A' est 65 mais celui de '2' n'est pas 2.

Caractères spéciaux:

\n (nouvelle ligne)

\t (tabulation horizontale)

\a (sonne),

\f (nouvelle page),

Les entiers

Exemple: unsigned short maValeur;

short (short int)	16 bits	-32.768 à 32.767
int	32 bits	-2.147.483.648 à 2.147.483.647
long (long int)	32 bits	-2.147.483.648 à 2.147.483.647
unsigned short	16 bits	0 à 65.535
unsigned long	32 bits	0 à 4.294.967.295

Constantes entières:24, -24, +25, -0

0x00ff (hexadécimale)

016 (octale)

54L (de type long)

Les réels

- Trois types possibles:
 - float, 32 bits, $-3,4e38$ à $3,4e38$ (7 chiffres significatifs)
 - double, 64 bits, $-1,7e308$ à $1,7e308$ (15 chiffres signi.)
 - long double, 80 bits, $3,4e-4932$ à $1,1e4932$ (non standard)
- Par défaut, une constante réelle est de type double mais on peut forcer le type à float (532.76f) ou à long (3E-10L)
- Stocké en deux parties: mantisse(signi.) et exposant.

La déclaration de variables

Constante

Constantes

constante symbolique



constante typée



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

```
#define PI 3.14
```

```
main()
```

```
{
```

```
    double x = 2*PI;
```

```
    const int nbEtudiants = 105;
```

```
    int n;
```

```
    n = nbEtudiants;
```

```
    nbEtudiants = 12;
```

```
}
```



ERREUR

Identificateurs

- *Identificateurs corrects:* *Identificateurs incorrects:*

nom1

nom_2

_nom_3

Nom_de_variable

deuxieme_choix

mot_francais

1nom

nom.2

-nom-3

Nom de variable

deuxième_choix

mot_français

déclaration des variables simples

- **Type NomVar1, NomVar2 ,..., NomVarN;**
- **int compteur,X,Y;**
- **float hauteur,largeur;**
- **double masse_atomique;**
- **char touche;**

Initialisation des variables

- ***Initialisation***

En C, il est possible d'initialiser les variables lors de leur déclaration:

```
int MAX = 1023;
```

```
char TAB = '\t';
```

```
float X = 1.05e-4;
```

const

- En utilisant l'attribut **const**, nous pouvons indiquer que la valeur d'une variable ne change pas au cours d'un programme:

```
Const int MAX = 767;
```

```
Const double e = 2.71828182845905;
```

```
const char newline = '\n';
```

Affichage de variables: `Printf()`

- `Printf(" format", variable1,variable2,...);`

```
int r=15;
```

```
double pi=3.14159;
```

```
printf( "un cercle de rayon %d a pour périmètre %f \n", r,2*pi*r);
```

—————> un cercle de rayon 15 a pour périmètre 94.247700

Affichage de variables: **Printf()**

Format	Type de variable correspondant
%c	Caractère de type char
%d ou %i	Entier de type int
%x	Entier de type int affiché en notation hexadécimale
%u	Entier non signé de type unsigned int
%f	Nombre à virgule flottante de type float ou double
%e ou %E	Nombre à virgule flottante affiché en notation exponentielle

Formatage des nombres avec `print()`

Format	Résultat	Description
%6d	123	Largeur minimum de champ de 6 caractères
%06d	000123	Largeur minimum de champ de 6 caractères, remplissage avec des zéros
%-6d	123	Largeur minimum de champ de 6 caractères, cadrage à gauche
%6.3f	3.141	Largeur minimum de champ de 6 caractères avec 3 chiffres après la virgule

Les opérateurs particuliers de C

- **Les opérateurs d'affectation**

En pratique, nous retrouvons souvent des affectations comme:

$$i = i + 2$$

En C, nous utiliserons plutôt la formulation **plus compacte**:

$$i += 2$$

- Pour la plupart des expressions de la forme:

$$\text{expr1} = (\text{expr1}) \text{ op } (\text{expr2})$$

- il existe une formulation équivalente qui utilise un opérateur d'affectation:

$$\text{expr1} \text{ op} = \text{expr2}$$

Les opérateurs particuliers de C

- ***Opérateurs d'affectation***

+= ajouter à

-= diminuer de

***=** multiplier par

/= diviser par

%= modulo

Les opérateurs particuliers de C

- **Opérateurs d'incrémentation et de décrémentation**

Les affectations les plus fréquentes sont du type:

$I = I + 1$ et $I = I - 1$

En C, nous disposons de deux opérateurs inhabituels pour ces affectations:

- **$I++$ ou $++I$** pour l'incrémentation (augmentation d'une unité)
 $I--$ ou $--I$ pour la décrémentation (diminution d'une unité)

Opérateurs d'incrémentation et de décrémentation

- **$X = I++$** passe d'abord la valeur de I à X et ***incrémente après***
- **$X = I--$** passe d'abord la valeur de I à X et ***décrémente après***
- **$X = ++I$** ***incrémente d'abord*** et passe la valeur incrémentée à X
- **$X = --I$** ***décrémente d'abord*** et passe la valeur décrémentée à X

- ***Exemple***

Supposons que la valeur de N est égal à 5:

Incrém. postfixe:

$X = N++;$

Résultat: **$N=6$ et $X=5$**

Incrém. préfixe:

$X = ++N;$

Résultat: **$N=6$ et $X=6$**

Les priorités des opérateurs

- ***Classes de priorités***

Priorité 1 (la plus forte):

()

Priorité 2:

! ++ --

Priorité 3:

* / %

Priorité 4:

+ -

Priorité 5:

< <= > >=

Priorité 6:

== !=

Priorité 7:

&&

Priorité 8:

||

Priorité 9 (la plus faible):

= += -= *= /= %=

Les fonctions arithmétiques standard

- **#include <math.h>**

<i>Fonctions C</i>	<i>EXPLICATION</i>	<i>LANG.ALGORITHMIQUE</i>
exp(X)	fonction exponentielle	eX
log(X)	logarithme naturel	ln(X), X>0
log10(X)	logarithme à base 10	log10(X), X>0
pow(X,Y)	X exposant Y	XY
sqrt(X)	racine carrée de X	pour X>0
fabs(X)	valeur absolue de X	 X
floor(X)	arrondir en moins	int(X)
ceil(X)	arrondir en plus	fmod(X,Y)
	reste rationnel de X/Y (même signe que X)	pour X différent de 0
sin(X)	cos(X)	tan(X)
asin(X)	acos(X)	atan(X)
sinh(X)	cosh(X)	tanh(X)