Sommaire chapitre 1

- Structure d'un programme C
- Les variables et leur type
- Entrée/sortie : `printf` et `scanf`
- **Opérateurs :**
  - **arithmétiques et conversion de type**

Opérateurs arithmétiques

operateur	signification	exemples	
+	addition	x=3+2;	/*constantes*/
		y+z;	/*variables*/
		x+y+2;	/*les deux*/
-	soustraction	3-2;	/*constantes*/
		int x = y - z;	/*variables*/
		y-2-z	/*les deux*/
*	multiplication	int x=3*2;	/*constantes*/
		int x=y*z;	/*variables */
		x*y*2;	/*les deux*/

- (-) est aussi un opérateur unaire (avec un seul opérande) :

```
int x = -i;    /*x = opposé de la variable i*/
```

Opérateurs arithmétiques

operateur	signification	exemples
/	division	double x=3/2; /*x=1.0 (quotient div. entière)*/
		double x=3.0/2.0; /* x=1.5 (div. float)*/
%	modulo (reste de la division entière)	int x=3%2; /* x=1*/
		int y=7; int x=y%10; /*x=7*/

- une opération
- n'est possible qu'entre variables/constantes de même type
- à un résultat dépendant du type

Note: pas d'opérateur « puissance » en C mais une fonction `pow()` existe.

Opérateurs arithmétiques

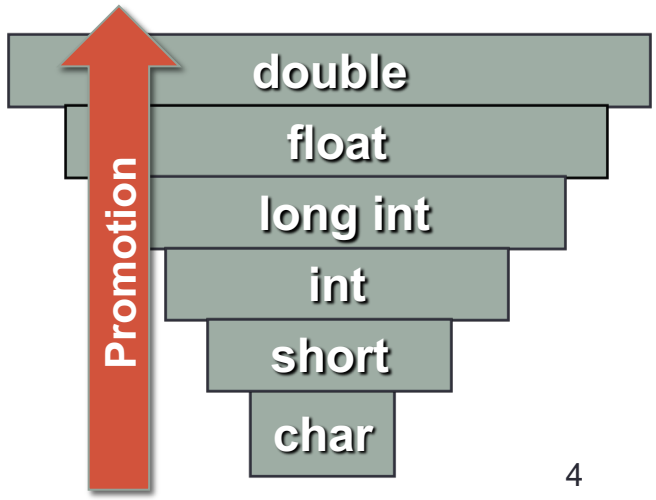
Conversion implicite de types :

1. si un opérateur mélange des types, le plus petit type est promu vers la plus grande précision

Exemple :

```
int x =2;
double f = 3.0/x;
```

```
  /* x promu en double   */
  /* f= 3.0/(double) x;  */
  /* f=3.0/2.0;          */
  /* f=1.5;              */
```



Opérateurs arithmétiques

Conversion implicite de types :

2. lors d'une affectation, conversion dans le type de destination.

Exemples :

```
int x = 3.0/2;      /*entier 2 promu en double */
                     /* x= 3.0/2.0;*/
                    /*x= 'troncature de 1.5';
                                        x=1;*/

float f = 2/3;      /*-> float f = (entier) 0 */
                     /*conversion : f =0.0 */
```

Opérateurs arithmétiques : de conversion

Conversion explicite de types :

- le programmeur peut forcer une conversion de type
- via l'opérateur de conversion ou cast : (`type`)

Exemples :

```
float f;
f= (float)2/3;      /*idem f = ((float)2)/3;
                            f = 2.0/3;
              [conv. explicite] ici f = 0.667;  */


printf (" %d", (int) 3.001 ); /*affiche 3*/
```

Opérateurs d'incrémentation

Incrémenter et décrémenter une variable est un opération commune :

`x++` est un raccourci pour `x=x+1;`
`x--` est un raccourci pour `x=x-1;`

Ainsi

`y = x++;` est un raccourci pour `y=x; x=x+1;`
`x` est évalué **avant** son incrémentation.

(`y = ++x;` est un raccourci pour `x=x+1; y=x;`
`x` est évalué **après** son incrémentation)