Introduction au langage C

6 janvier 2010

Table des matières

Chapitre 1

Les opérateurs

1.1 Définition

Les opérateurs sont des symboles qui permettent de manipuler des variables, c'est à dire effectuer des opérations, les évaluer, etc.

On distingue plusieurs types d'opérateurs :

1. les opérateurs de calcul ;
2. les opérateurs d'assignation ;
3. les opérateurs d'incrémentation ;
4. les opérateurs de comparaison ;
5. les opérateurs logiques ;
6. les opérateurs bit à bit ;
7. les opérateurs de décalage de bit.

1.1.1 Les opérateurs de calcul

Les opérateurs de calcul permettent de modifier mathématiquement la valeur d'une variable.

Opérateur	**Dénomination**	**Effet**	**Exemple**	**Résultat avec x=7**
+	Opérateur d'addition	Ajoute deux valeurs	x+3	10
-	Opérateur de soustraction	soustrait deux valeurs	x-3	4
*	Opérateur de multiplication	multiplie deux valeurs	x*3	21
/	Opérateur de division	divise deux valeurs	x/3	2.3333333
=	Opérateur d'affectation	affecte une valeur à une variable	x=3	Met la valeur 3 dans la variable x

1.1.2 Les opérateurs d'assignation

Ces opérateurs permettent de simplifier des opérations telles que ajouter une valeur dans une variable et stocker le résultat dans la variable. Une telle opération s'écrirait habituellement de la façon suivante, par exemple : `x=x+2` . Avec les opérateurs d'assignation il est possible d'écrire cette opération sous la forme suivante : `x+=2` . Ainsi, si la valeur de `x` était `7` avant opération, elle sera égale à `9` après ... Les autres opérateurs du même type sont les suivants :

Opérateur	**Effet**
+=	additionne deux valeurs et stocke le résultat dans la variable (à gauche)
-=	soustrait deux valeurs et stocke le résultat dans la variable
*=	multiplie deux valeurs et stocke le résultat dans la variable
/=	divise deux valeurs et stocke le résultat dans la variable

L'opérateur d'affectation = renvoie aussi une valeur, qui est celle de la variable après affectation. Cela permet notamment de faire des affectations en cascade :

```
a = b = c = 1;
```

ce qui correspond à :

```
a = (b = (c = 1));
```

1.1.3 Les opérateurs d'incrémentation

Ce type d'opérateur permet de facilement augmenter ou diminuer d'une unité une variable. Ces opérateurs sont très utiles pour des structures telles que des boucles, qui ont besoin d'un compteur (variable qui augmente de un en un). Un opérateur de type `x++` permet de remplacer des notations lourdes telles que `x=x+1` ou `x+=1`.

Opérateur	**Dénomination**	**Effet**	**Syntaxe**	**Résultat avec x=7**
++	Opérateur d'incrémentation	Augmente d'une unité la variable	x++	8
-	Opérateur de décrémentation	Diminue d'une unité la variable	x–	6

1.1.4 Les opérateurs logiques (booléens)

Ce type d'opérateur permet de vérifier si plusieurs conditions sont vraies :

Opérateur	**Dénomination**	**Effet**	**Syntaxe**
||	OU logique	Vérifie que toutes les conditions sont réalisées	((condition1)||(condition2))
&&	ET logique	soustrait deux valeurs	((condition1)&&(condition2))
!	NON logique	Inverse l'état d'une variable booléenne (retourne la valeur 1 si la variable vaut 0, 0 si elle vaut 1)	(!condition)

1.1.5 Les opérateurs de comparaison

Opérateur	**Dénomination**	**Effet**	**Exemple**	**Résultat avec x=7**
== **A ne pas confondre avec le signe d'affectation** =	Opérateur d'égalité	Compare deux valeurs et vérifie leur égalité	x==3	Retourne 1 si x est égal à 3, sinon 0
<	Opérateur d'infériorité stricte	Vérifie qu'une variable est strictement inférieure à une valeur	x<3	Retourne 1 si x est strictement inférieur à 3, sinon 0
>	Opérateur de de supériorité stricte	Vérifie qu'une variable est strictement supérieure à une valeur	x>3	Retourne 1 si x est strictement supérieur à 3, sinon 0
>=	Opérateur de supériorité	Vérifie qu'une variable est supérieure ou égale à une valeur	x>=3	Retourne 1 si x est supérieur ou égal à 3, sinon 0
<=	Opérateur d'infériorité	Vérifie qu'une variable est inférieure ou égale à une valeur	x<=	Retourne 1 si x est inférieur ou égal à 3, sinon 0
!=	Opérateur de différence	Vérifie qu'une variable est différente d'une valeur	x!=3	Retourne 1 si x est différent de 3, sinon 0

1.1.6 Les opérateurs bit à bit

Ce type d'opérateur traite ses opérandes comme des données binaires, plutôt que des données décimales, hexadécimales ou octales. Ces opérateurs traitent ces données selon leur représentation binaire mais retournent des valeurs numériques standard dans leur format d'origine. Les opérateurs suivants effectuent des opérations bit à bit, c'est-à-dire avec des bits de même poids.

Opérateur	Dénomination	Effet	Syntaxte	Résultat
&	ET bit à bit	Retourne 1 si les deux bits de même poids sont à 1	9 & 12 (1001 & 1100)	8 (1000)
|	OU bit à bit	Retourne 1 si l'un ou l'autre des deux bits de même poids est à 1 (ou les deux)	9 | 12 (1001 | 1100)	13 (1101)
	OU bit à bit exclusif	9 12 (1001 1100)	5 (0101)	

1.1.7 Les opérateurs de décalage de bit

Ce type d'opérateur traite ses opérandes comme des données binaires d'une longueur de 32 bits, plutôt que des données décimales, hexadécimales ou octales. Ces opérateurs traitent ces données selon leur représentation binaire mais retournent des valeurs numériques standard dans leur format d'origine. Les opérateurs suivants effectuent des décalages sur les bits, c'est à dire qu'ils décalent chacun des bits d'un nombre de positions vers la gauche ou vers la droite. La première opérande désigne la donnée sur laquelle on va faire le décalage, la seconde désigne le nombre de décalages.

Opérateur	Dénomination	Effet	Syntaxte	Résultat
«	Décalage à gauche	Décale les bits vers la gauche (multiplie par 2 à chaque décalage). Les zéros qui sortent à gauche sont perdus, tandis que des zéros sont insérés à droite	6 « 1 (110 « 1)	12 (1100)
»	Décalage à droite avec conservation du signe	Décale les bits vers la droite (divise par 2 à chaque décalage). Les zéros qui sortent à droite sont perdus, tandis que le bit non nul de poids plus fort est recopié à gauche	6 » 1 (0110 » 1)	3 (0011)

1.2 Les priorités

Lorsque l'on associe plusieurs opérateurs, il faut que le compilateur sache dans quel ordre les traiter, voici donc dans l'ordre décroissant les priorités de tous les opérateurs :

Priorité des opérateurs

```
++++++++++++++ () []
++++++++++++   -- ++ ! ~ -
++++++++++++   * / %
++++++++++     + -
+++++++++      << >>
++++++++       < <= >= >
+++++++        == !=
++++++         &
+++++          ^
++++           |
+++            && ||
++             ?:
+              = += - = *= /= %= <<= >>= &= ^= |=
```