Variables et expressions

Lorsque l'on écrit des expressions en C++, elles sont constituées de deux éléments principaux : Les **opérateurs** et les **données**. Les données peuvent être des valeurs immédiates, par exemple : 5, 56, 3.1415, -985, ou alors être des variables.

Les 5 types de données de base

Il existe en C++ 5 types de données : **char** (caractère), **double** (nombre à virgule flottante en double précision), **float** (nombre à virgule flottante en simple précision), **int** (entier) et **void** (sans valeur).
En plus de ces 5 types de données, il existe le **bool** (booléen qui peux valoir vrai (1) ou faux (0)).

La taille et la plage de données de ces types peuvent varier selon l'environnement et le compilateur utilisé mais la taille du char est toujours d'un octet.

Les variables et leurs caractéristique

Type	Plage de valeurs	Taille en bits
char	de –127 à 127	8
unsigned char	de 0 à 255	8
double	10 chiffres après la virgule	64
long double	10 chiffres après la virgule	80
float	6 chiffres après la virgule	32
int	de –32 767 à 32 767 ou de –2 147 483 645 à 2 147 483 645	16 ou 32
unsigned int	de 0 à 65 535 ou de 0 à 4 294 967 295	16 ou 32
short int	de –32 767 à 32 767	16
unsigned short int	de 0 à 65 535	16
long int	de –2 147 483 645 à 2 147 483 645	32
unsigned long int	0 à 4 294 967 295	32

Les identificateurs

Les identificateurs peuvent être le nom d'une variable, d'une fonction, ou de tout étiquette définie par l'utilisateur.
Ils peuvent être composés de **lettres**, de **chiffres** et de **traits de soulignements**.
Ils ne doivent pas commencer par un trait de soulignement et de doivent pas être un mot clé.

Identificateurs

Corrects	Incorrects
Compteur	2compteur
vArIaBlE	variable !
essai123	123_essai
compteur_numero_127	compteur.numero.127

Le C++ respecte la **casse**, c'est à dire que l'identificateur *variable* sera différent de *Variable* et différent de *VARIABLE*.

Les variables

Une variable est un emplacement mémoire qui contient une valeur qui peux être modifiée par le programme. Contrairement au Basic, le C++ oblige la déclaration des variables avant leur utilisation selon la syntaxe suivante :

t*ype nom_de_la_variable ;*

Plusieurs variables du même type peuvent être déclarées en même temps.

```
int age ;
double nombre1, nombre2, nombre2, resultat ;
unsigned short int variable1, variable2 ;
```

La portée d'une variable est définie par l'endroit ou celle-ci est déclarée : Si elle est déclarée dans une fonction, elle ne sera connue que dans cette fonction, elle peux être connue de tout le programme si elle est déclarée en dehors de toute fonction.

Modificateurs de variable

const

Lorsqu'une variable est préfixée du mot clé *const,* elle doit être initialisée lors de sa création mais sa valeur ne peux pas changer au cours de l'exécution du programme.

```
const long int x = 123 ;
```

La valeur de *x* est fixée à 123 et ne plus être changée.

volatile

Il s'agit d'une variable dont la valeur peux changer sans instruction explicite, par exemple si une variable est affectée à l'heure système.

extern

Le C++ permet la compilation d'un programme répartit dans plusieurs fichier. Le mot clé *extern* permet de rendre une variable accessible dans plusieurs fichier.
La variable doit être déclarée normalement dans le premier fichier et déclarée avec le mot clé *extern* dans tous les fichiers où elle sera utilisée.

Fichier 1

```
int a, b, c;
```

Fichier x

```
extern int a, b, c;
```

register

Le fait de faire précéder le type de la variable du mot clé *register* demande au compilateur de stocker la variable dans le registre du processeur plutôt que dans la RAM. Le temps d'accès est donc considérablement plus court. Si un nombre trop important de variables register sont déclarées, le compilateur les stocke automatiquement dans la mémoire traditionnelle.

static

Si une variable est déclarée *static,* elle gardera sa valeur d'un appel à l'autre mais sa portée ne changera pas.

Initialisations des variables

Les variables peuvent être ou non initialisées lors de leur déclaration. Il faut toujours initialiser une variable avant de l'utiliser car la valeur qu'elle prend lors de sa déclaration est imprévisible.

```
int a, b, c ;
double pi = 3.14159265 ;
short premier = 1, second = 2, troisieme = 3, dernier ;
```

Opérateurs

1) Opérateur d'affectation

L'opérateur d'affectation à la syntaxe suivante :

nom_de_variable = expression ;
nom_de_variable = valeur ;

La cible, le membre de gauche, doit être une variable, il ne peux pas s'agir d'une valeur ou d'une fonction.

```
int a, b = 5 ;
a = b;
b = -2;
```

L'affectation multiple est autorisée.

```
int a, b, c;
a = b = c = 0;
```

Le C++ permet un certain nombre de *conversions de types* lorsque les deux opérandes ne sont pas de même type. Il peux entraîner des pertes d'informations notamment des arrondis.

2) Opérateurs arithmétiques

Les opérateurs sont : + (addition), - (soustraction ou changement de signe), * (multiplication), / (division), % (modulo), -- (décrémentation), ++ (incrémentation).

L'opérateur ++ augmente de un la variable concernée, l'opérateur – la diminue de 1. La position de ces opérateurs influe sur le résultat.

Exemple :

Dans ce cas, l'incrémentation est effectuée avant l'affectation, *b* vaut donc 11 ;

```
a = 10;
b = ++a;
```

Dans ce cas, l'incrémentation est effectuée après l'affectation, *b* vaut donc 10.

```
a = 10;
b = a++;
```

3) Opérateurs raccourcis

Il existe des opérateurs qui effectuent une affectation et une opération. Ils sont : +=, -=, *=, /=, %=.

a += *2* est équivalent à *a* = *a* + *2*
a *= *4* est équivalent à *a* = *a* * *4*