C++

Variables

Introduction

Avant de voir ce qu'est une variable, il est important de comprendre ce qu'elle représente d'un point de vue physique dans un ordinateur. Celui-ci dispose de mémoire vive (RAM), dans laquelle on réserve de l'espace pour exécuter un programme et stocker ses variables. La RAM a une taille définie (par exemple 32 Go), ce qui correspond à une certaine quantité d'octets disponibles.

Pour comprendre ce que cela signifie, il faut revenir à la base : le binaire. L'unité la plus simple est le bit, qui ne peut valoir que 0 ou 1. Comme un seul bit ne permet de représenter que deux valeurs possibles, on en regroupe plusieurs. Ainsi, 8 bits forment un octet, qui peut représenter 256 combinaisons différentes.

Définition

Une variable est une association entre un nom et un espace mémoire. Pour bien comprendre ce concept, il faut savoir que chaque type de variable occupe une taille différente en mémoire.

Une fois créée, une variable permet de stocker une donnée dans l'espace mémoire qui lui est réservé. Cette donnée peut être modifiée au cours de l'exécution du programme. Les variables sont indispensables au bon fonctionnement d'un programme, car elles permettent de conserver et de manipuler des informations essentielles. Par exemple, dans un jeu vidéo, on pourra avoir une variable pour représenter la vie du joueur et sa valeur évoluera à mesure que le jeu progresse.

Types de variables

Comme dit précédemment il existe différent type de variables, voici quelques uns des plus courants :

- Integer (représente un nombre entier)
- float (représente un nombre à virgule flottante)
- bool (représente un état binaire pouvant être soit à true ou à false)
- char (représente un seul caractère orthographique)

Types de variables

Ces variables n’occupent pas toutes la même quantité de mémoire : leur taille est directement liée à leur utilisation.

- Par exemple, un int (entier) doit pouvoir représenter des nombres relativement grands. Sur de nombreux systèmes, il occupe donc 4 octets, ce qui permet de représenter environ 4,2 milliards de valeurs différentes .
- En revanche, un char (caractère) ne sert qu’à stocker un seul symbole (par exemple 'A' ou '%'). Il n’a donc besoin que d’1 octet, ce qui suffit à coder 256 caractères possibles (en ASCII).

Déclaration

A présent il est temps de voir comment utiliser ces variables dans un programme informatique, pour déclarer un variable il faut lui associer un nom et sans que cela soit obligatoire une valeur par défaut.

```
int UninitNumber;
int InitNumber = 87;
```

Si l’on conseille souvent de donner une valeur par défaut à une variable, ce n’est pas sans raison.

En effet, lorsqu’une variable est créée, elle est associée à un espace mémoire. Si aucune valeur n’y est explicitement placée, alors le programme utilisera simplement le contenu déjà présent dans cette portion de mémoire.

Opération

Une fois une variable créée, il est possible de la manipuler grâce à différents opérateurs.

- Les opérateurs arithmétiques (+, -, *, /, %) permettent de réaliser des calculs.
- L'opérateur d'affectation (=) permet d'attribuer une valeur à une variable.
- Il existe aussi d'autre opérateur que l'on verra plus en détails dans les cours futur tel que les opérateurs de logiques et de comparaison.

Il est important de comprendre qu'un opérateur retourne toujours un résultat. Ce résultat peut être immédiatement utilisé, ou bien stocké dans une variable, à condition que le type corresponde.

Opération

Voici un exemple de l'utilisation de l'opérateur arithmétique + et de l'opérateur d'assignation = :

```
int NumberA = 50;
int NumberB = 87;

int NumberC = NumberA + NumberB;
```

Ici l'opération est entre deux integers et va retourner un integer, il est donc possible de directement stocker ce résultat dans une nouvelle variable

Différent types

Comme vu précédemment, il existe différents types de variables, chacun ayant son usage. Il n'est donc pas toujours possible de passer directement d'un type à un autre (par exemple d'un int vers un bool). Pour cela, il faut intervenir explicitement en programmation et appliquer une conversion.

Certains types sont convertibles, mais la conversion peut modifier la valeur initiale

Ici on déclare une variable de type float dans laquelle on stocke 3,75. La conversion en int est autorisé cependant la valeur sera tronqué (arrondi à l'entier inférieur), en effet ici la valeur qui ressortira de Y sera 3. La partie décimale est perdue

```
float X = 3.75;
int Y = X;
```

Modulo

En plus des quatre opérateurs arithmétiques de base (+, -, *, /), il existe un cinquième opérateur : le modulo (%).

Contrairement à ce que son symbole pourrait laisser penser, il ne représente pas un pourcentage. Le modulo permet de récupérer le reste de la division euclidienne.

```
int ResultA = 7 % 3;
int ResultB = 10 % 2;
```

Ici, ResultA sera égal à 1, car il représente le reste de la division de 7 par 3. ResultB, quant à lui, sera égal à 0, car 10 divisé par 2 ne laisse aucun reste.

Le modulo peut être très utile notamment pour détecter un nombre pair ou pour répéter un cycle.

Incrémentation et décrémentation

Un dernier point important concernant les opérateurs arithmétiques est la notion d'incrémentation et de décrémentation. Écrire une opération comme x = x + 1 peut sembler un peu lourd. Pour simplifier, le langage C++ propose l'opérateur ++ pour ajouter 1 à une variable, et -- pour soustraire 1.

```
int X = 0;
X = X + 1;
```

=

```
int X = 0;
X++;
```

Bien que n'ayant rien à voir ces deux syntaxes donnent exactement le même résultat.

Exercice

Déclarez deux variables A et B et ensuite écrivez un programme qui permet d'échanger leur valeurs.