```
#include <string>
#include <iostream>
using namespace std;

int main()
{
    // Création de quelques variables numériques :
    int nombreEntier = 5;
    float nombreDecimal = 5.5;

    // Création de quelques variables littérales :
    // Remarquez comment les guillemets simples sont utilisés
    // pour les caractères, et les guillemets doubles
    // pour les chaînes.
    string salutation = "Bonjour à tous ";
    char ponctuation = '!';
    // Création d'une variable booléenne :
    bool estVrai = false;

    // Affichage de la somme de nos variables :
    cout << nombreEntier + nombreDecimal << endl;
    cout << salutation + ponctuation << endl;
    cout << estVrai << endl;

        return 0;
}
```

Regardez ce qui se passe quand des variables de types différents sont additionnées. Celles de type numérique donnent le résultat escompté, 5 + 5,5 est égal à 10,5 et c'est bien ce que nous obtenons. Quand nous ajoutons des variables littérales, les lettres sont ajoutées aux autres. Ainsi, nous avons notre chaîne, "Bonjour à tous ", et nous y ajoutons le point d'exclamation, si bien que nous avons finalement "Bonjour à tous !". C'est pourquoi les types de données sont importants. Si les nombres avaient été enregistrés en tant que chaînes, l'addition aurait ajouté 5.5 après 5, et le résultat aurait été "55.5". C'est une bonne idée également de donner des noms judicieux aux variables. Ces noms peuvent contenir des lettres, des nombres et des caractères de soulignement (underscore), mais ne peuvent pas commencer par un nombre. Il existe aussi certains mots-clés qu'il n'est pas possible d'utiliser car ils sont réservés pour le langage.

Notez également que estVrai renvoie 0 au lieu de "false". Un booléen vaut fondamentalement 0 pour faux, ou 1 pour vrai, comme en binaire, et c'est pourquoi le programme l'affiche ainsi. La fois précédente nous avons utilisé cin pour permettre à l'utilisateur de saisir une valeur et de l'enregistrer dans une variable. Essayez de faire ça avec le code ci-dessus. Le signe = peut aussi être utilisé à tout moment pour modifier ce qui est enregistré dans une variable (c'est la raison pour laquelle elles sont appelées variables). La valeur d'une variable peut être affectée à une autre variable, par exemple :

```
int i = 4;
int j = i;
```

Par contre, il n'est pas possible de l'utiliser pour affecter une valeur à une chaîne :

```
int i = 4;
string s = i;
```

car la classe string ne possède aucune fonction membre qui permettrait cette affectation. L'affectation d'un nombre à une chaîne peut être réalisée grâce à un flux de chaîne de caractères (stringstream), cela sera abordé lors d'un prochain tutoriel.

Article d'Alex Kerr