# Creating Variables and Classes in C++

Creating variables in C++ is very similar to Java and C#. Here are a few of the most common types used:

int - An integer (Usually 32-bit). Represents whole numbers

float - A floating point number (usually 32-bit). Can represent numbers with decimals

double - A floating point number(usually twice the byte size of a float)

bool - A Boolean. Can represent true or false

char - A character. Represents a singular character (Usually 8-bit)

void - A non-type. Use for functions that do not return a value.

Copy the following code into a project and compile it. This program makes use of all of the types above and initialises them.



Make a note of the difference between an Int, a Float, and a Double. Int’s are initialised by giving it a whole number, Double’s require a decimal point, Float’s also require a decimal point but we add an ‘f’ to the end to distinguish between the two.

Unlike Java, some standard variables like String do not exist as a base type in C++. Luckily, C++ comes with a set of libraries called the Standard Template Libraries which contain templates for objects like Strings, Dynamic Arrays, Maps, etc.

To create a string in C++, you need to make sure the file you are using the string in has the proper include. In this case, we can add the following at the top of the code to allow use to use Strings:



Now all we have to do is define a String and we can use it in our code.



## Operators

Similar to Java and C#, in C++ there are operators we can use to manipulate variables to produce a new result. The most common operators are below:

* - Subtraction
* + Addition
* \* Multiplication
* / Division
* ++ Increment
* -- Decrement
* = Assignment

There are a lot of others but we will work with the ones above for now. Look into the others though and incorporate them into your exercises. Knowing all the operators is very useful skill.

Below is the code that makes use of all the above operators. A lot of them are very similar if not identical in operation to their counterparts in Java/C#.



Notice above that all the variables are of the type float. It is advisable to keep the types the same but you can add an int and a float together for example as the compiler can cast between these two types.

## Arrays

Arrays in C++ are similar to their Java/C# counterparts but with a few small differences. In C++ we can create the array in the code without the use of the new keyword. If you are comfortable with pointers you can use new to create arrays, but we will cover that later. The normal syntax for an array is as follows:

****

The first instance we create an array of five ints but we don’t initialise any values. The second example creates and array of five ints and sets up all five values using an initialiser list. The third example does exactly the same as the previous but doesn’t include and = operator. Example two and three behave identically.

To access items in the array, we can add [n] after the array name to access element at index number n. Remember arrays start from 0, so the indexes for an array of five items are 0, 1, 2, 3 & 4. If you try to access elements outside those ranges, the program may crash. An easy way to remember is that the valid indexes of an array are from zero to the size of the array minus one. Or:



## Other Useful Formatters

When working with outputs into the console, there are a couple of extra functions we can use to format the output in a way we want. The two covered here are:

* setw(n) – Sets a space of n for between the current and next printed item.
* setprecision(n) – Sets the precision of float and double variables that you output.

Both these functions require the iomanip library that you can include at the top of your code files like so:



### A Useful Note

Whenever you start a C++ project, make sure you are aware of which namespace header files store their data. If it is in the standard C++ namespace, make sure to prefix each call with std:: or add *using namespace std;* to the top of the file.



The compiler will read the file from the top to the bottom so make sure #includes and the namespaces are setup at the top.

## Exercises

* 1. Create a calculator that divides two inputs together and prints the result. Make sure to use floats and set the precision to three decimal places.
  2. Write a program that performs one of each of the calculations as shown above (addition, subtraction, multiplication, division) and prints the results onto the console. You can have the user input the data manually or hard code it.
  3. Expand the St. Ives riddle exercise from the last worksheet, to figure out and print the answer. The riddle is:

As I was going to St. Ives,   
I met a man with seven wives.   
Each wife had seven sacks,   
Each sack had seven cats,   
Each cat had seven kits.   
Kits, cats, sacks, and wives,   
How many were going to St. Ives?