CS 111

Program Layout, cin, cout, Variables, Arithmetic

Program Layout

- For most of this course, your programs will be based on the following template:

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

int main(){
        // write stuff
        return 0;
}
```

Program Layout

- Proper indentation makes code more readable

```
int main(){
        // indent code inside curly braces
        return 0;
}
```

- // ← Backslashes are used for comments

- Comments are text strings that are not read by the compiler. They are helpful notes to explain the code to yourself or people reading it.

Variables

- In order to receive input from a user, the computer must set aside memory to store the input
- The computer needs to know what type of information it needs to store
  - E.g., the number 1 is a different type of data to store than the text string “Queens College”
- You can think of variables as boxes that you create to put data in

Receiving Input

- Once you have created a variable, you can put data into it
- You can either put the data into it yourself, or ask a user to provide the input
- If you ask the user to provide the input, you will use the cin keyword

```
int num;  // declare integer variable num
cin >> num;  // cin obtains input from the user and stores it in num
```

Printing to the Monitor

- Printing to the monitor allows the program to interact with the user
  - Tell the user what type of input you need
  - Show the user the results of the executed program
  - Debugging

- Accomplish this by using the cout keyword

```
int num;
cout << “Please enter a number: “;
cin >> num;
```

Variable types

- int – whole numbers
  - e.g., 21
- double – numbers with a decimal component
  - e.g. 3.14
- string – sequences of characters enclosed in double quotes
  - e.g. “CS111” or “Queens College”
- char – a single character enclosed in single quotes
  - e.g. ‘1’ or ‘a’ or ‘&’
- bool – true/false
  - e.g. 1 or true or 0 or false

Arithmetic Operations

- Addition (+)
  - x + y
- Subtraction (-)
  - x - y
- Multiplication (*)
  - x * y
- Division (/)
  - x / y
- Remainder (%) (aka mod)
  - x % y

Be mindful of Order of Operations
e.g. **3 + 4 * 5** is different from **(3 + 4) * 5** (why?)

Integer Division is another thing to be mindful until you learn to think more like a computer.
e.g. **7/3 = 2** when both **7** and **3** are stored as integers

Integer Division

- C++ requires some extra consideration when a certain level of precision is needed in the result of division
- For instance, given integers 7 and 2, the result of dividing 7 by 2 in a C++ program would be 3, when the precise result is 3.5
- You need to signal to C++ that the result requires greater precision by setting up the division to include a double rather than just ints
- Given integers 7 and 2, if you want 3.5 as the result when dividing you need to divide 7 by 2.0, or 7.0 by 2

Pseudocode

- For the first few labs, I will provide pseudocode to help guide you through writing some of the programs.
- According to Wikipedia, “pseudocode is a plain language description of the steps in an algorithm or another system.”
- It is often structured like typical programs, but is meant to be read by a human rather than a machine.

Pseudocode

- From a pseudocode representation, you can translate the pseudocode into a computer program that can be read by a computer.
- It is not necessary to follow the pseudocode outline.
- When pseudocode is provided it is meant to help provide some guidance regarding an approach to a solution.

Lab 3.1 Solution Pseudocode

```
Function Main
        // This program demonstrates arithmetic operations
        Declare integer x
        Declare integer y
        Ask the user for a number
        Obtain and store user input into x
        Ask the user for a number
        Obtain and store user input into y
        Output the sum of x and y
        Output the result of subtracting y from x
        Output the product of x and y
        Output the average of x and y
        Output the remainder when x is divided by y
```

Lab 3.3 Solution Pseudocode

Function Main

    // Obtains a four-digit number and prints it in reverse
    Declare integer num
    Ask the user for a four-digit number
    Obtain and store user input into num
    Print num mod 10 to the monitor
    Set num equal to num divided 10
    Repeat the above two statements two more times
    Print num to the monitor