# Worksheet 1


[Notes](#)
[Directions](#)
[Questions](#)


---

## Notes

- The previous iteration of WS1 is archived here.

## Directions

- Create a working C++ program to complete this assignment.
- Some questions, beginning with a comment (//) do not require code and may be answered using comments or printing to stdout (cout).
- Unless they have a return value, all functions should be void type.
- Start with the template below and use these commands to compile and execute:
    - **g++ ps0.cpp**
    - **./a.exe**
- Template

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

void q1a(){
    // answer question 1a in this function
}

// etc...

int main(){

   // call your functions in main
   q1a();
   q1b( "Mr. B" );

   // etc...

    return 0;
}
```

# Questions

1. Basics
    a. Write a function that declares and initializes three variables of type int, float, and char. Use sizeof to output the size (in bytes) of each.
    b. Repeat 1a with the following types: long, long long, double, and long double.
    c. // what range of values can be represented by each of the following types: int, unsigned int, long, long long?
    d. Write a function that declares an int, and initializes it with a char value (e.g. 'A'). Output the value. // why does it have that value?
    e. Write a function declares and initializes a string and prints its value to standard output (cout).
    f. Write a function with one parameter (string name) and prints the value to stdout.
    g. Write a function with two parameters (int a, int b) and returns the sum of the two. Output the returned value in main.
2. Control flow
    a. Write a function with two parameters (int a, int b) and compares them. Return: -1 if a<b, 0 if a==b, 1 if a>b.
    b. Write a function with one parameter (int a) and return true (the bool, not the string) if a is positive and even.
    c. Write a function that continually takes user input until that input is the string "quit".
    d. Write a function that takes one string parameter and uses a loop to output each letter, one at a time, with a comma separating them.
    e. Same thing as d, but go over the string backwards.
    f. Same thing as d, but stop if you get to the letter 'x'.
3. String and file processing
    a. Write a function that tokenizes a string, printing each token separated by commas.
    b. Create a file "input.txt" in the same directory as your program, and write a function that reads and prints all of the lines in the file.
    c. Write a function that reads all of the lines in a file and tokenizes them, until you run out of input or reach a line with "quit" or "exit" as the first token.
4. Classes & objects
    a. Create the Rectangle class that is defined in the notes.
    b. Implement the 3 extra methods: toString, setDimensions, and getArea.
    c. Create an instance of Rectangle using the default constructor and setDimensions.
    d. Create another Rectangle object using the 2-argument constructor.
    e. Compare the Rectangles using getArea and use toString to print the one with greater area.