# CS 1428
# Fall 2018
# Gentry Atkinson
# Lab 7

## Introduction:

Recall that at the beginning of every C++ program you type the line "**int main ()**" or something similar. Whether or not you knew it, you've been defining a function called main, saying that it will return an int value, and saying that it will not be passed any values. This is because every C++ program must have a **main** function but it can include many more.

Functions let us separate out small sections of code so that we can invoke those lines with a single name rather than having to retype them. Every function needs 3 things:

1. A unique name or more accurately a unique signature. It's possible for two functions to have the same name if they have different parameters but this is something we'll study more later. For now every function should have a unique name.
2. A return type. This can be of any variable type or the **void** type which tells the compiler that the function will not return a value. A function can only return one value in C++.
3. A parameter list. These are values that are given to the function that the function will be able to process.

The purpose of today's lab is to familiarize you with the definition and invocation of functions.

## Directions:

**1-** Launch Code::Blocks and start a new file. Name it your_last_name_lab7.cpp.
**2-** Include the standard header for this lab:

```
//Your Name
//CS1428 Fall 2018
//Lab 7
```

**3-** Include the iostream standard library and declare some functions that we will be using in this lab. Start your main function:

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

```
int findLargest(int list [], int size);
float findAverage(int list[], int size);
void copyArray(int from[], int to[], int size);


int main() {
```

**4-** Create the following variables inside of your **main** function:

1. An integer array called **firstArray** with the values: {3, 9, 8, 8, 7, 4, 5, 1, 11}
2. An integer array called **secondArray** with default values and the same size as the first array.
3. An integer called **arraySize** whose value is equal to the size of **firstArray** or **secondArray.**

**5-** Below your **main** function, copy the following line of code to start defining the **findLargest** function:

```
int findLargest(int list [], int size) {

}
```

Implement the following pseudo-code to fill in the body of **findLargest**

```
//DECLARE AN INTEGER VARIABLE CALLED largest WITH VALUE 0
//DECLARE AN INTEGER VARIABLE CALLED counter WITH VALUE 0
//ENTER A LOOP WHICH WILL CONTINUE WHILE counter < size
//IF list[counter] > largest THEN largest = list[counter]
//EXIT LOOP
//RETURN largest
```

**6-** Now that you're familiar with how to fill in the body of a function you should be able to implement the **findAverage** function on your own. The most common way to do this is to use a loop to add up all the values in **list**, divide the total by the number of values in **list,** and then return the result. Be aware that your code will want to use integer division rather than floating point so try using **static_cast<float>** to cause the compiler to treat your values as floats rather than ints.

**7-** Debug the following poorly written code to implement the **copyArray** function, which should copy all of the values in the **from** array into the **to** array:

```
void copyArray(int from[], int to[], int size){
        four(int i = 1; i < size; size++) {
                to[j] == from[j];
        }
        return i;
}
```

**8-** Copy the following code into your **main** function:

```
cout << “Largest value: ” << findLargest(firstArray, size) << endl;
cout << “Average value: ” << findAverage(firstArray, size) << endl;

copyArray(firstArray, secondArray, size);

for(int i = 0; i < size; i++) cout << secondArray[i] << “ ” << endl;
```

**9-** What value would you expect to be returned from **findLargest(firstArray, size - 1)**? Print your answer with a single **cout** statement in your **main** function.