EE208 Laboratory Session 5

Bryan Hennelly

27 February 2018

1 Lab objectives

In this lab you will start to work with classes in C++ for the first time. We have only just started to learn about them in the last lecture. You will need to read over that last lecture and you will also have to read ahead into the coming lectures; the slides are available on moodle for you. Tomorrow's lecture will cover the fundamentals of using classes, including access specifiers (public, private, etc.) as well as contructors and destructors. You will need to read ahead and teach yourself these concepts in order to complete the exercised below. The due date is 5pm Monday 5th March.

Learning Outcomes
From a C++ perspective you will be able to implement simple C++ classes and use instances of those classes (objects) within your programs. In this lab you will create and use classes for the first time, you will employ contructors and destructors and you will overload operators such as + and *.

2 Questions:

For each of the problems given below write a C++ program that provides a solution. You will just need one filename for this exercise as each probelm builds on the same code.

Remember:

- **Comment your code.**
- **Use proper indentation for function and control structure.**

Exercise 4.1: Have a quick look at your solution to last weeks exercise on Complex numbers; this week you will be doing something very similar, except this time you will be using C++ classes. In particular define a new class called `Complex`.

There should be two `public` data members, called `REAL` and `IMAG`, both of which should be type `float`.

Provide `public` member functions for each of the following:

Getting the modulus of the `Complex` number using the following prototype:
`float Modulus(void)`

Getting the angle, which you should be familiar with from polar co-ordinates, of the `Complex` number, using the following prototype:
`float Argument(void)`

Changing the complex number to be equal to it's own conjugate, using the following prototype:
`void Conjugate(void)`

In the main function you should request the user to input values for a single Complex number and then print out all the information you can about the complex number using the methods listed above.

Exercise 4.2: Take your solution to Exercise 4.1 and change it such that the two data members `REAL` and `IMAG` are now both `private` but leave al the methods as `public`. Now, since you cant access these values directly from the main function anymore you must provide a constructor function that enables an object of this class to be initialised when it is declared. You should also define a default constructor that initialises both values to be set equal to zero. You should also provide two new `public` member functions to :

Reset the values of the real and imaginary values of the `Complex` number, using the following prototype:
`void setvalues(double x, double y)`

You will also need to provide a `print` function to output a complex number to the screen (e.g. $2+4i$). The function should have the following prototype:
`void print()`

Demonstrate your new constructors, and your two new methods in action in the main function.

Exercise 4.3: You should now provide two new `public` member functions to overload the operators + and * to add and multiply two Complex numbers, respectively. The function prototypes should look like:
`Complex operator+(Complex)`
`Complex operator*(Complex)`

Demonstrate your new operators in action in the main function.

Exercise 4.4: Create a new class called `OneD_Complex_matrix`. This class should contain two private data members, a pointer to type `Complex` and an integer N. You should create a constructor with the following prototype
`OneD_Complex_matrix(int N)`
This constructor should take in a single paramater and use this to dynamically allocate memory for a 1D matrix of Complex numbers, it should then set all of the real values and imaginary values to be zero. Create a public method to set the values to random values and create a second public method to convert the matrix into its conjugate. The function prototypes should look like:
`void set_random_values()`
`void conjugate()`
Demonstrate objects of your new class in action in the main function.

Finally you should add a destructor to delete the allocated memory.