# LAB 3

**Long Vo, Haru Chu, Nicholas Krouse**

1.

a. *A description of the objectives/concepts explored in this assignment including why you think they are important to this course and a career in CS and/or Engineering.*

The objective in this lab is to overload math operations and handle multiple classes and files. This concept is important since it is applied into the professional field. In the professional field, we would not only just work on our own file or program but also, we would need to think of how to connect with other files/classes from other programmers in a smooth way. The overload concept in this lab is a good example of things that will not work in ways that we want it to be unless we make it do. The basic math is not applicable to complex numbers, and we need to prepare it by overloading it with updated information. It would apply the same in the professional CS field, there would be more overload and override we would attempt to face in the long term.

Our rationale for the member functions we included is as follows. For the Private data members, we decided to include a and b as the co-efficient of the real and imaginary numbers, respectively. After that, we decided that we would include r, the distance from origin to a point on the imaginary and real plane, and angle, the angle in degrees between the real plane and the point. We included this because we realized that you can easily convert between cartesian coordinates and polar coordinates, and that even if someone inputs a value in the cartesian system, they might still want to know r and angle, and vice versa. After that, we made our usual constructors, including one for cartesian and another for polar, and the normal setters and getters for the private data. Then we decided that if we wanted the ability for the user to calculate r or theta after changing the a and b values, that we should include a function that calculates them called calculateR() and calculateAngle(). Finally, we decided that we wanted to overload the operators for +, -, *, /, and ==. This is because since these are technically representations of numbers, albeit imaginary ones, that they should be able to have normal arithmetic operations done on them.

b. *Why you designed the class the way you did initially, what changes you made because of each task and what considerations you consider important when designing classes.*

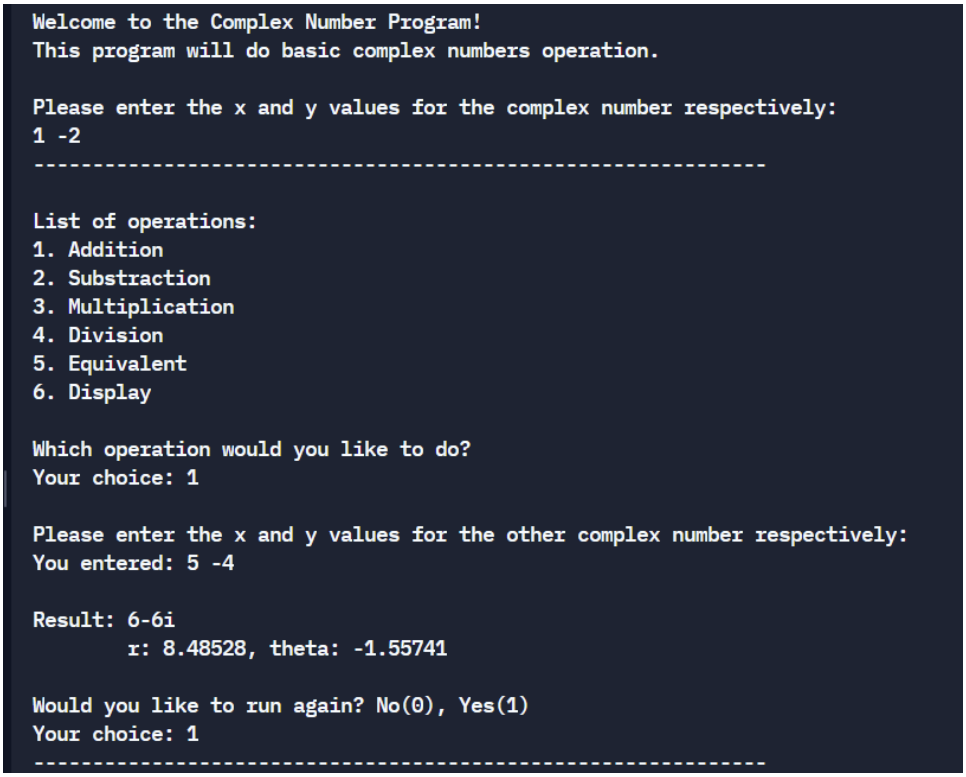
We first created the class Complex to store the default constructor, a constructor to calculate the complex number, and a constructor to determine whether the attributes were in polar complex plane form. After that, we declared the getters and setters accordingly as we moved through the tasks. We changed the functions' attributes and parameters several times to optimize the algorithm for each task, and in the end, we were able to complete the requirements. One of the major changes to note though was the inclusion of the calculateR and calculateAngle functions to the .h file. When we first introduced the idea of including both cartesian and polar coordinates, we realized that if we changed the a and b values, that without some sort of calculation the r and

angle values would stay constant when they should not. Therefore, we decided to make those two functions so that we would not have to constantly copy the code.

When designing classes, it is important to determine the necessary constructors, getters, and setters beforehand and adjust them accordingly as we go through the requirements of the tasks.

2. Screenshots of output

Program was coded and executed on Repl.it.



```
List of operations:
1. Addition
2. Substraction
3. Multiplication
4. Division
5. Equivalent
6. Display

Which operation would you like to do?
Your choice: 2

Please enter the x and y values for the other complex number respectively:
You entered: 1 10

Result: 5-16i
        r: 16.7631, theta: -0.0584739

Would you like to run again? No(0), Yes(1)
Your choice: 1
------------------------------------------------------------

List of operations:
1. Addition
2. Substraction
3. Multiplication
4. Division
5. Equivalent
6. Display

Which operation would you like to do?
Your choice: 3

Please Enter the number you want to multiply by:
You entered: 5

Result: 25-80i
        r: 83.8153, theta: -0.0584739
```

```
Would you like to run again? No(0), Yes(1)
Your choice: 1
------------------------------------------------------------

List of operations:
1. Addition
2. Substraction
3. Multiplication
4. Division
5. Equivalent
6. Display

Which operation would you like to do?
Your choice: 6

Result: 25-80i
        r: 83.8153, theta: -0.0584739

Would you like to run again? No(0), Yes(1)
Your choice: 1
------------------------------------------------------------

List of operations:
1. Addition
2. Substraction
3. Multiplication
4. Division
5. Equivalent
6. Display

Which operation would you like to do?
Your choice: 4

Please Enter the number you want to divide by:
You entered: 3

Result: 8.33333-26.6667i
        r: 27.9384, theta: -0.0584739
```

```
Would you like to run again? No(0), Yes(1)
Your choice: 1
------------------------------------------------------------

List of operations:
1. Addition
2. Substraction
3. Multiplication
4. Division
5. Equivalent
6. Display

Which operation would you like to do?
Your choice: 5

Please enter the x and y values for the other complex number respectively:
You entered: 8.33333 -26.6667

Result: Non-equivalent.

Would you like to run again? No(0), Yes(1)
Your choice: 0

Program end!
```