ECE2036: Week 2_A – Introduction to C++ Classes (D&D Chapter 3)

Today we will introduce the idea of C++ classes, which is a primary software component used in object oriented programming (OOP). The goal with these techniques is to minimize *errors*, improve coding *efficiency*, and to streamline code *evolution*.

Today we will consider a story of a simple programmer who wanted to create a library for users to use complex numbers in C++. We will roughly compare the techniques between basic "structured programming" and more advanced "object oriented programming" to provide motivation for understanding the complexity of C++ classes and "instantiating" objects.

```
//Starter program – same file with no constructors

#include <iostream>
using namespace std;

class ComplexNumber
{
public: // access specifier that makes items public

void setReal (float xr)
{   real = xr;  }

void setImag (float yi)
{   imag = yi;   }

void setComplexNumber(float xr, float yi)
{ //good programming practice to reuse code when possible
setReal(xr);
setImag(yi);
}

float getReal()
{   return(real);   }

float getImag()
{   return(imag);   }


private: //access specifier that encapsulates data

float real;
float imag;

};  //don't forget the semicolon here!

int main()
{

ComplexNumber num1;

num1.setReal(3);
num1.setImag(4);

//display number to terminal
cout << num1.getReal() << " +j" << num1.getImag() <<endl;

//example of nesting functions
num1.setReal(num1.getImag()); // overwrites the real data

//print result
cout << "The real part of num1 is " << num1.getReal() << endl;
}
```