# EN3085 Assessed Coursework 1

You should solve the problems independently from other students and submit only your own work. **Submit** your solution **on blackboard** using the **Coursework Answer Sheet** provided **by** **18:00,** **20/03/17 (week 9).**

In section one of the Coursework Answer Sheet, explain the structure of your class in terms of memory usage and “user friendliness”, how you tested it and any identified issues. Then, from Microsoft Visual C++, **copy and paste all your code in** the relevant section of **the Coursework Answer Sheet**. **Add a screen shot** of the window showing what is printed on the screen when running your program (for example using Microsoft Snipping Tool).

All developed code should be included in the 3 files provided: **Complex.h**, **Complex.cpp** and **Main.cpp** and these completed files should also be **uploaded together** with the **Coursework Answer Sheet** to facilitate our testing of your program.

1. Create a class **Complex** to allow programmers to use complex numbers in their programs. Anobject of this class should: [5]
   1. Be able to store a complex number in the most efficient way.
   2. Make use of *constructor* and *destructor* functions. The *constructor* should take three arguments: a character indicating if the last two arguments should be interpreted as real and imaginary parts or magnitude and angle, and two real numbers defining the value of the complex number. The constructor should also initialise the members of the object appropriately, including for the case when it receives only some or no arguments.
   3. Be able to return individual element of the complex number in both rectangular (real and imaginary) and polar form (magnitude and angle).
   4. Allow modification of the stored complex number by enabling the modification of individual elements of the complex number (both in rectangular and polar form).
   5. Include a function to print on the screen the complex number, either in rectangular or polar form.
2. Using the class **Complex** developed inquestion 1, write a main function which will: [5]
   1. Read all complex numbers recorded in the provided text file “**ComplexList.txt**” and create for each of them an object of type complex. Each data line in the text file consist of one of the following:

* a letter ‘p’ followed by a magnitude and an angle (in radians) or
* a letter ‘r’ followed by the real and imaginary parts of the complex number.
  1. Store in a vector all complex objects created from the text file (using the vector class).
  2. After all complex numbers have been read and stored, print them on the screen in both rectangular and polar forms.

1. Expand the class Complex with the following overloaded member operators: binary operators +, -, \*, and / and a unary operator -. [5]

The operators should return complex numbers only in rectangular form and provide the standard functionality for arithmetic operations on complex numbers. Using those operators, in the main function:

* 1. Calculate the sum of all complex number stored in the vector.
  2. Demonstrate the use of all operators using the following equation: A= - B\*(C-D)/E.
  3. Print on the screen the complex number resulting from the equation above in both rectangular and polar form.

When relevant you should make use of all the “best practice” concepts covered in the lecture (e.g. inline, const).

##### **Marking scheme**

**(1) Source code. Maximum** marks will be awarded for:

1. *A correctly functioning program*. The program should operate according to the specification.
2. *An efficient program and elegant algorithms*. Try to develop algorithms, which are efficient in terms of the amount of data, which needs to be stored (e.g. minimum number of variables used), and the speed in which they operate.
3. *Appropriate use of classes and functions*. Try to keep your function main() as short as possible.
4. *A user-friendly program*. When your program runs, the messages displayed on the screen should be easy to understand and succinct. The user should be in no doubt as to what they must do to use the program.
5. *A well commented program*. The judicious use of commenting is essential if somebody else is to easily understand your program. Start your programs with a short comment specifying the name of the program, your name, and what your program is designed to do. Supply your functions with pre- and post-conditions.
6. *A well laid out program*. Indent your program code. Indent your comments as well. Use blank lines to space out blocks of code.

**(2) Documentation.** Marks will be awarded for:

* *A well documented program*. Include a short description of the problem-solving phase and the implementation of your program, as well as an account of its *limitations and possible improvements*.
* *A well tested program.* You must supply and explain the choice of values, which you used to test your program and the answers, which the program gave. You should also document any particular cases for which your program fails, or better add code to your program to make sure that they do not occur.

Dr Y Hicks