

Problem 1:

Using good coding practices, design your own **ComplexVector** class for vectors whose elements are complex numbers. Provide an overloading implementation for term by term addition, subtraction, multiplication, division, and the stream output operator. Such that if **v1** and **v2** are **ComplexVector** objects, then one can compute **v1 + v2**, etc.

Hint : you may want to create a **Complex** class separately.

Complex numbers are expressed in the form $a+bi$, where a is the real part, b is the imaginary part, and i is the imaginary unit satisfying $i^2 = -1$. Print your complex numbers using this representation. When adding (or subtracting) two complex numbers, add (or subtract) their real parts and imaginary parts separately. The multiplication of two complex numbers is defined by,

$$(a + bi)(c + di) = (ac - bd) + (bc + ad)i, \quad (1)$$

while division is given by,

$$\frac{a + bi}{c + di} = \frac{ac + bd}{c^2 + d^2} + \frac{bc - ad}{c^2 + d^2}i. \quad (2)$$

Using your newly defined **ComplexVector** class, write a **recursive** function to print the first 6 terms in the sequence from equation (3) to a file titled “ComplexSequence.txt”.

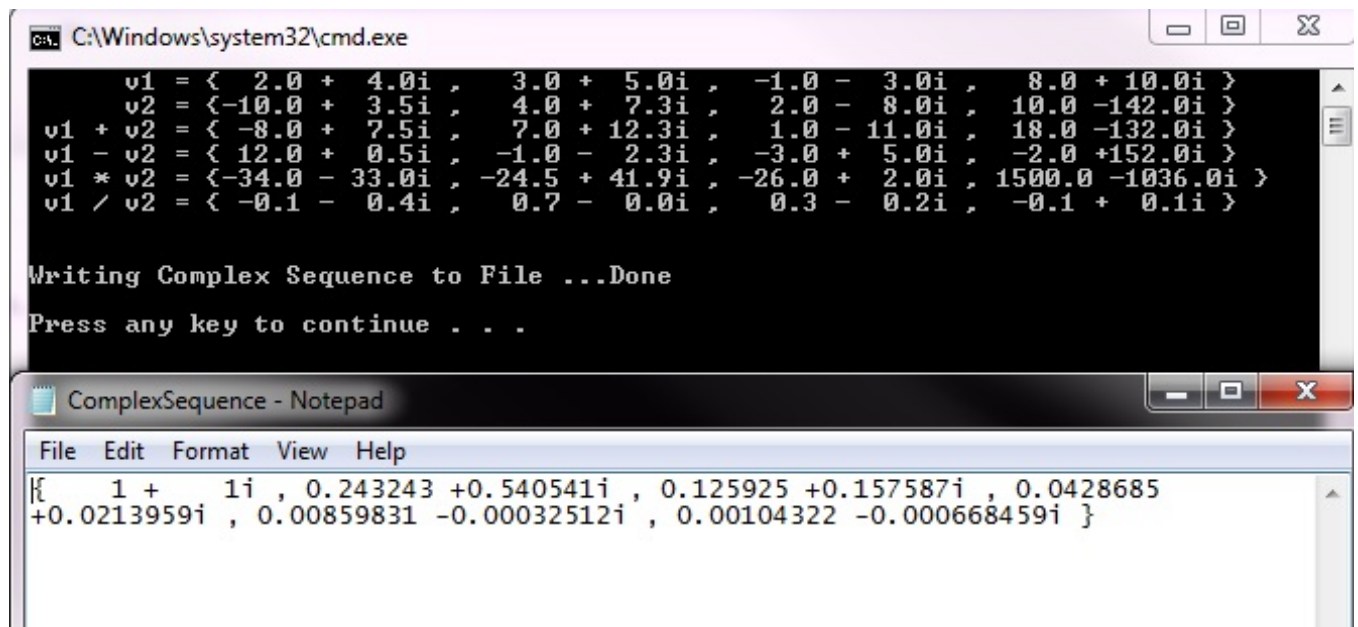
$$f_{n+1} = \frac{(2 + 3i)n}{7 + 5n^2i} f_n, \quad (3)$$

$$f_1 = 1 + i \quad (4)$$

Using the vectors **v1** and **v2** ,

v1 = { 2 + 4 i, 3 + 5 i, -1 - 3 i , 8 + 10 i }
v2 = { -10 + 3.5 i, 4 + 7.3 i, 2 - 8 i, 10 -142 i},

write a main function that checks **v1 + v2**, **v1 - v2**, **v1 * v2**, and **v1 / v2** by printing them to the console. Figure 1 shows the sample outputs.



The screenshot shows a Windows command prompt window titled "C:\Windows\system32\cmd.exe" and a Notepad window titled "ComplexSequence - Notepad".

The command prompt displays the following calculations:

```

v1 = { 2.0 + 4.0i , 3.0 + 5.0i , -1.0 - 3.0i , 8.0 + 10.0i }
v2 = { -10.0 + 3.5i , 4.0 + 7.3i , 2.0 - 8.0i , 10.0 -142.0i }
v1 + v2 = { -8.0 + 7.5i , 7.0 + 12.3i , 1.0 - 11.0i , 18.0 -132.0i }
v1 - v2 = { 12.0 + 0.5i , -1.0 - 2.3i , -3.0 + 5.0i , -2.0 +152.0i }
v1 * v2 = { -34.0 - 33.0i , -24.5 + 41.9i , -26.0 + 2.0i , 1500.0 -1036.0i }
v1 / v2 = { -0.1 - 0.4i , 0.7 - 0.0i , 0.3 - 0.2i , -0.1 + 0.1i }

```

Below the calculations, it says "Writing Complex Sequence to File ...Done" and "Press any key to continue . . .".

The Notepad window shows the following text:

```

{ 1 + 1i , 0.243243 +0.540541i , 0.125925 +0.157587i , 0.0428685
+0.0213959i , 0.00859831 -0.00032512i , 0.00104322 -0.000668459i }

```

Figure 1: Sample output.

Good Coding Practices:

- think about cross-platform. Don't use Windows or Mac only commands. For example, `pause == cin.get()` twice, write many `\n` vs. `system(clear)` or `system('cls')`.
- passing objects by reference `&` or `const &` when possible
- using field initializer list when possible in all constructors

Instructions for submission:

- Name your files exactly `hw4.cpp`, `Complex.h`, `Complex.cpp`, `ComplexVector.h`, and `ComplexVector.cpp`.
- You may not use `#include "stdafx.h"`.
- Add code description in the comment at the beginning of the file. A sample description may look like:

```

/*
    PIC 10B 2A, Homework 1
    Purpose: Tic-tac-toe game
    Author: Hanqin Cai
    Date: 10/10/2019
*/

```

- Submit your header files and source codes to BruinLearn in separate files. Only `.h` and `.cpp` files should be uploaded.