Problem 4

姓名	學號	日期
吳宇昕	B10831020	2022/11/8

> OOP挑戰題，將過去作業四的complex struct matrix改以class方式呈現

sorce code and replit

終端機輸出

程式可以在終端機執行時輸入四個command line argument浮點數作為隨機複數實部與虛部的上下限，也可以執行程式後依提示設定。下圖以輸入command line argument為例。

```
(base) PS D:\NTUST_Not_Sync\EngineeringProgramming\code\EngineeringProgramming\midterm\CODE\bin> .\Problem4.exe -5.0 -10 15.0 17.0
Student ID: B10831020
      real imaginary
  min  -10   15
  max   -5   17

Complex matrix m1
 Row ID|                                Content                                |        Row avg
      0|        (-6.85,15.8)         (-9.13,16)          (-9.88,16)       (-8.52,16.5)|     (-7.84,16)
      1|         (-6.9,16.5)       (-8.58,16.3)        (-7.52,15.9)       (-8.92,16.6)|   (-7.61,16.1)
      2|        (-8.34,16.6)       (-9.12,16.8)        (-5.79,15.9)       (-8.66,16.6)|   (-8.16,16.2)
      3|        (-7.68,16.9)       (-7.58,16.9)        (-5.19,16.9)       (-8.08,16.6)|   (-7.29,16.4)
Sum of the matrix excluding the diagonal = (-2.2e+002,4.5e+002)


Complex matrix m2
 Row ID|                                Content                                |        Row avg
      0|          (-8.4,15)       (-6.69,15.5)        (-8.61,15.8)       (-7.63,15.5)|   (-7.17,15.8)
      1|        (-6.37,15.6)       (-6.88,16.5)          (-5.07,16)       (-8.31,16.7)|   (-6.86,15.9)
      2|        (-9.18,15.6)       (-8.42,16.6)        (-7.38,16.8)       (-7.63,15.9)|   (-8.19,16.4)
      3|        (-9.74,16.4)       (-8.69,16.8)        (-6.51,16.7)          (-7.2,17)|   (-7.91,16.1)
Sum of the matrix excluding the diagonal = (-2.1e+002,4.5e+002)


Addition of the two matrices
 Row ID|                                Content                                |        Row avg
      0|        (-15.3,30.8)       (-15.8,31.5)        (-18.5,31.7)         (-16.2,32)|   (-16.4,31.5)
      1|        (-13.3,32.1)       (-15.5,32.7)        (-12.6,31.9)       (-17.2,33.3)|   (-14.6,32.5)
      2|        (-17.5,32.1)       (-17.5,33.4)        (-13.2,32.7)       (-16.3,32.5)|   (-16.1,32.7)
      3|        (-17.4,33.2)       (-16.3,33.7)        (-11.7,33.7)       (-15.3,33.6)|   (-15.2,33.6)
Sum of the matrix excluding the diagonal = (-1.9e+002,3.9e+002)


subtraction of the two matrices
 Row ID|                                Content                                |        Row avg
      0|         (1.56,0.78)       (-2.43,0.553)       (-1.27,0.219)     (-0.893,0.953)| (-0.759,0.626)
      1|      (-0.529,0.866)      (-1.71,-0.192)      (-2.45,-0.176)     (-0.618,-0.164)| (-1.33,0.0838)
      2|       (0.844,0.994)      (-0.703,0.212)       (1.59,-0.929)       (-1.03,0.654)|  (0.175,0.233)
      3|        (2.06,0.497)       (1.1,0.00662)        (1.32,0.157)     (-0.881,-0.357)| (0.901,0.0758)
Sum of the matrix excluding the diagonal = (-4.6,4.8)
```

照老師的建議，用`std::setw()`為輸出數值預留版面，讓終端機輸出看起來更順眼整齊。

Operator Overload

二維複數陣列class名稱為`CplxMatrix`，為此class自行定義+、-與ostream operator的<<三種operator overload。

+與-定義如下，為`class CplxMatrix`的public method

```
37      CplxMatrix operator+ (const CplxMatrix& another)
38      {
39          CplxMatrix tmp(this->mNRows, this->mNCols, "Addition of the two matrices");
40          for(int i=0; i<this->mNRows; i++){
41              tmp.mData[i].reserve(this->mNCols);
42              for(int j=0; j<this->mNCols; j++){
43                  tmp.mData[i].push_back(this->mData[i][j] + another.mData[i][j]);
44              }
45          }
46          tmp.mGetNonDiagonalSum();
47          tmp.mGetRowAvg();
48          return tmp;
49      }
50
51      CplxMatrix operator- (const CplxMatrix& another)
52      {
53          CplxMatrix tmp(this->mNRows, this->mNCols, "subtraction of the two matrices");
54          for(int i=0; i<this->mNRows; i++){
55              tmp.mData[i].reserve(this->mNCols);
56              for(int j=0; j<this->mNCols; j++){
57                  tmp.mData[i].push_back(this->mData[i][j] - another.mData[i][j]);
58              }
59          }
60          tmp.mGetNonDiagonalSum();
61          tmp.mGetRowAvg();
62          return tmp;
63      }
```

ostream << 定義如下

位於`class CplxMatrix`的public區域

```
35      friend std::ostream& operator<< (std::ostream& stream, const CplxMatrix& mat);
```

位於`class CplxMatrix`之外

```
123 std::ostream& operator<< (std::ostream& stream, const CplxMatrix& mat)
124 {
125     stream << '\n';
126     stream << mat.mMatrixName << '\n';
127     stream << std::setw(7) << "Row ID" << std::setw(1) << '|' << std::setw(40) << "Content"
128         << std::setw(40) << ' ' << std::setw(1) << '|' << std::setw(15) << "Row avg" << std::endl;
129     for(int i=0; i<mat.mNRows; i++){\
130         stream << std::setw(7) << i << std::setw(1) << '|';
131         for(int j=0; j<mat.mNCols; j++){
132             if(j==mat.mNCols - 1){
133                 stream << std::setw(20) << std::setprecision(3) << mat.mData[i][j] << std::setw(1) << '|' << std::setw(15) << mat.mRowAvg[i] << '\n';
134             }
135             else{
136                 stream << std::setw(20) << std::setprecision(3) << mat.mData[i][j];
137             }
138         }
139     }
140     stream <<  "Sum of the matrix excluding the diagonal = " << std::setprecision(2) << mat.mNonDiaSum << '\n';
141     return stream;
142 }
```

這次自行定義ostream operator才知道C++還有個關鍵字叫做`friend`，修飾一個class之外的function或operator，讓非class內的函式讀寫class的private member。

心得

這份題目需整合第三方的Complex API，且練習以OOP語法定義虛數陣列。有感覺到自定義簡單算數運算子帶來的方便性，讓兩個虛數陣列相加減的程式碼看起來更直觀易懂。

同時有學到一個vector的新函式。過去我們使用`push_back()`將原始資料型別的元素加進一個vector讓它增長。但是若要把一個class instance加入一條vector，用另一個函式`emplace_back()`更為理想。

過去我們的做法是這樣：

```
#include <complex>
#include <vector>
int main(){
    std::vector<std::complex<int>> vtr; // create a vector contain float complexes
    vtr.push_back(std::complex<int>(1, 2)) // push 1+j2 into the vector
}
```

這種寫法，會在main()的stack frame裡創建一個complex instance，並複製此instance到vector裡，捨棄原本創進的complex instance。

比較好的做法應該是

```
    vtr.emplace_back(1, 2) // push 1+j2 into the vector
```

`emplace_back()`函式會自動把傳入的參數導引至complex的constructor，呼叫constructor並直接把得到的instance存入vector的記憶體中，不須複製。同時也會讓vector增長，幾乎與`push_back()`有同樣的效果，但是處理class instance時效能更好。