

Assignment 2

1301058

Zhang Junming

Exercise 1

Question

EXERCISE 1 (5 POINTS OUT OF 15)

Define a class called **Month** that is an abstract data type for a month. Your class will have one member variable of type **int** to represent a month (1 for January, 2 for February, and so forth). Include all the following methods:

1. A constructor to set the month using the first three letters in the name of the month as three arguments;
2. A constructor to set the month using an integer as an argument (1 for January, 2 for February, and so forth);
3. A default constructor;
4. An input function that reads the month as an integer;
5. An input function that reads the month as the first three letters in the name of the month;
6. An output function that outputs the month as the whole name of the month (January, February, and so forth) ;
7. and a member function that returns the next month as a value of type **Month**.

Test your class appropriately.

Model Answer

Software Development Process

1. Problem statement

Define a class which called Month to write a program, this program could construct month by their first three letters or by an integer. User could choose how to get the full name of the month, they could enter the integer or the first three letters of the month.

2. Analysis

Inputs:

- 1) An integer
- 2) First three letters of month

Outputs:

- 1) Full name of the month

Additional requirements or constraint

Just one member variable of type **int** to represent a month.

3. Design

Algorithm

1. Adding "iostream" and "string" header files.
2. Using of the std namespace.
3. Define a class which called Month

Private:

int month – represent a month

Public:

Month() - default constructor

<1> let month equal to 0(represent January).

void SetMonthOf3Letters(string l1,stringl2,stringl3) - a constructor to set month by their first three letters.

<1> Declare a string l and then store 12 first three letters of the month in it.

<2> Declare a string l4 equal to l1 plus l2 and l3 to get the first three letters of the month.

<3> Setting up a loop to confirm which months match the first three letters.

void SetMonthOfInt(int Month) - a constructor to set month by an integer.

<1> let month equal to Month minus 1 because 0 represent January.

void InputAndReadMonthAsInt() - user could input a integer and then the program will return the full name of the month with the integer.

<1> Ask user to enter an integer.

<2> declare and store the value in n.

<3> let month equal to n minus 1.

void InputAndReadMonthAs3Letters() user could input the first three letters of the month and then the program will return the full name of the month with the integer.

<1> Declare a string l and then store 12 first three letters of the month in it.

<2> Declare 4 strings to store first three letters and the letters which they connect together.

<3> Setting up a loop to ask user enter the letter, if they enter error, let user enter again.

<4> Connect the first three letters.

<5> Setting up a loop to confirm which months match the first three letters.

<6> if cannot find the related month, give the answer to user.

void OutputMonth() - program will display the full name of the month on the screen.

<1> Declare a string Fullname and then store the full name of 12 month in it.

<2> Output the full name of month based on the integer in the class member.

<3> If the value of month not in the area between 0 and 11(January to December), tell user check their input.

Month next() – program will return the value of the next month.

<1> Month t - construct a class t to store related information

<2> To judge the integer of month, if the value of the integer is 11(represents December) then return the 0(represents January) to month, else the value of integer will plus 1(represents next month).

<3> Return the value of integer to t.

4. Write the main function

<1> Display the task menu to user.

<2> Using switch and do-while function to store the different operations.

<3> test data was input in the program, therefore the default construct is January and the integer construct is October.

4. Implementation:

See the C code in file exercise1.c with comments.

```
3/*  
  
Name: Simple Program to display the full name of months  
  
File Name: EXERCISE 1.c  
  
Copyright: Free  
  
Author: Zhang Junming  
  
Description: Input the integer and first three letters of month, and the program will display the full name of the month  
  
*/  
  
#include <iostream>  
#include <string>  
  
using namespace std;  
  
class Month  
{  
    private:  
        int month; // represent the month(1 for January.....12 for December)  
    public:  
        Month();  
        void SetMonthOf3Letters( string l1 , string l2 , string l3 );// l1,l2,l3 represent the first three letters of month  
        void SetMonthOfInt( int Month );  
        void InputAndReadMonthAsInt ();  
        void InputAndReadMonthAs3Letters ();  
        void OutputMonth ();  
        Month next()  
        {  
            Month t;  
            if(month == 11) // 11 represent December  
            {  
                t.month = 0; // 0 represent January  
            }  
            else  
            {  
                t.month = month + 1; // next month  
            }  
            return t;  
        }  
};
```

```

Month :: Month()
{
    month = 0;
}

void Month :: SetMonthOf3Letters ( string l1, string l2 , string l3 )
{
    string l[12] = {"Jan", "Feb", "Mar", "Apr", "May", "Jun",
                   "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"};
    string l4 = l1 + l2 + l3; // l4 represent the connection of the first three letters
    int k = 0;
    for( ; k < 12 ; k++)
    {
        if( l4 == l[k] ) // the first three letters match the right value
        {
            month = k; // k represent the integer of the right month
        }
    }
}

void Month :: SetMonthOfInt ( int Month )
{
    month = Month - 1; // 0~11 for month but user input 1~12 represent month
}

void Month :: InputAndReadMonthAsInt ()
{
    int n;
    cout << "Please enter the number\n" << endl;
    cin >> n;

    month = n - 1;
}

void Month :: InputAndReadMonthAs3Letters ()
{
    string L[12] = {"Jan", "Feb", "Mar", "Apr", "May", "Jun",
                   "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"};
    string L1 , L2 , L3 , L4;

    do
    {
        cout << "Please enter the first letter(capital) in the name of the month\n" << endl;
        cin >> L1;
    }
}

```

```

if(L1.length() != 1)
{
    cout << "just enter one letter, first letter must be capital letter " << endl;
}
}
while(L1.length() != 1); // limit user enter one letter for once

do
{
    cout << "Please enter the first letter in the name of the month\n" << endl;
    cin >> L2;
    if(L2.length() != 1)
    {
        cout << "just enter one letter " << endl;
    }
}
while(L2.length() != 1);

do
{
    cout << "Please enter the third letter in the name of the month\n" << endl;
    cin >> L3;
    if(L3.length() != 1)
    {
        cout << "just enter one letter " << endl;
    }
}
while(L3.length() != 1);

L4 = L1 + L2 + L3; //connect the first three letters together

int i = 0;
int n = 0;
for(; i < 12 ; i++)
{
    if( L4 == L[i] )
    {
        month = i;
    }
    else
    {
        n = n + 1; // for counting
    }
}
if( n == 12 ) // when n equal to 12 represent the program match the month fail
{
    month = 13; // in output function, if month > 12, program will tell user there have problem
,

```

```

    }
}

void Month::OutputMonth()
{
    string Fullname[12] = {"January", "February", "March", "April", "May", "June",
                          "July", "August", "September", "October", "November", "December"};
    if( month >= 0 && month < 12)// it mean match the right month
    {
        cout << " \nNow, the month is: " << Fullname[month] << endl;
    }
    else// not match the right month
    {
        cout << "\nenter error please check " << endl;
    }
}

int main(void)
{
    int j = 0;
    // aims in this assignment, give convenience to test
    cout << "task 1: constructor to set the month using the first three letters in the name of the month as three arguments\n";
    cout << "task 2: constructor to set the month using an integer as an argument\n" << endl;
    cout << "task 3: A default constructor\n" << endl;
    cout << "task 4: An input function that reads the month as an integer\n" << endl;
    cout << "task 5: An input function that reads the month as the first three letters in the name of the month\n" << endl;
    cout << "task 6: An output function that outputs the month as the whole name of the month\n" << endl;
    cout << "task 7: A member function that returns the next month as a value of type Month\n" << endl;
    cout << "8-quit\n" << endl;

    do
    {
        cout << "Please choose the task number!\n" << endl;
        int chioce = 0;
        cin >> chioce;
        switch(chioce)
        {
            case 1:{ Month test1;
                    test1.SetMonthOf3Letters("J", "a", "n");// for test, here I enter Jan to test 3 letters constructor
                    test1.OutputMonth();
                    break;
                }
        }
    }
}

```



```

case 2:{ Month test2;
        test2.SetMonthOfInt(10); // for test, here I enter 10 to test integer constructor
        test2.OutputMonth();
        break;
    }

case 3:{ Month test3; // default constructor
        test3.OutputMonth();
        break;
    }

case 4:{ Month test4;
        test4.InputAndReadMonthAsInt();
        test4.OutputMonth();
        break;
    }

case 5:{ Month test5;
        test5.InputAndReadMonthAs3Letters();
        test5.OutputMonth();
        break;
    }

case 6:{ cout << "None!" << endl; break; } // output function, no test for it

case 7:{ Month test7, test8;
        test7.SetMonthOf3Letters("J", "a", "n"); // test the next month function and the I set January to the first month
        test8 = test7.next();
        test8.OutputMonth();
        break;
    }
case 8:{ j = 1;
        break; }
    }
    }
    while( j == 0 );

    return 0;
}

```

5. Testing:

```
task 1: constructor to set the month using the first three letters in the name of
the month as three arguments

task 2: constructor to set the month using an integer as an argument

task 3: A default constructor

task 4: An input function that reads the month as an integer

task 5: An input function that reads the month as the first three letters in the
name of the month

task 6: An output function that outputs the month as the whole name of the month

task 7: A member function that returns the next month as a value of type Month

8-quit

Please choose the task number!
```

Task menu

```
Month test1;
test1.SetMonthOf3Letters("J","a","n");// for test, here I enter Jan to test 3 letters constructor
test1.OutputMonth();
break;
```

```
Please choose the task number!

1

Now, the month is: January
```

Task 1

```
{ Month test2;
  test2.SetMonthOfInt(10);// for test, here I enter 10 to test integer constructor
  test2.OutputMonth();
  break;
}
```

```
Please choose the task number!

2

Now, the month is: Outtober
```

Task 2

```
3
Now, the month is: January
```

Task 3: default constructor is January

```
Please choose the task number!
4
Please enter the number
6
Now, the month is: June
```

Enter correct

```
4
Please enter the number
13
enter error please check
```

Enter wrong

Task 4

```
Please choose the task number!
5
Please enter the first letter(capital) in the name of the month
A
Please enter the first letter in the name of the month
u
Please enter the third letter in the name of the month
g
Now, the month is: August
```

Enter correct

```

Please choose the task number!
5
Please enter the first letter<capital> in the name of the month
a
Please enter the first letter in the name of the month
x
Please enter the third letter in the name of the month
g
enter error please check

```

Enter wrong

```

Please choose the task number!
5
Please enter the first letter<capital> in the name of the month
Aug
just enter one letter, first letter must be capital letter

```

Enter wrong

Task 5

```

Please choose the task number!
6
None!

```

Task 6: output function, display in other task

```

Month test7, test8;
test7.SetMonthOf3Letters("J", "a", "n"); // test the next month function and the I set January to the first month
test8 = test7.next();
test8.OutputMonth();
break;

```

```

Please choose the task number!
7
Now, the month is: February

```

Task 7: next month function and input January for the first month

Exercise 2

Question

EXERCISE 2 (5 POINTS OUT OF 15)

Design a new class to represent a fraction (a ration of two integer values).

$$\frac{15}{22} \text{ — Numerator}$$
$$\text{ — Denominator}$$

The data members of the class **Fraction** are two integers top and bottom, denoting the numerator and denominator, respectively.

```
1  class Fraction
2  {
3  private:
4      int top;           // Numerator
5      int bottom;        // Denominator
6  public:
7      . . . . .
8  };
```

Part 1: Fundamental requirements for the class Fraction.

1. Fractional numbers can be declared with both a numerator and denominator, or simple numerator:

```
Fraction a;           // represents 0/1
Fraction b(3,4);       // represents 3/4
Fraction c(5);         // represents 5/1
```

2. Fractions should be able to act just like other numbers.

Add, subtract, multiple and divide;
Compare based on values;

Input and output.

Part 2: Advanced requirements (Optional, not counted in marking)

3. Fractional number is normalized to ensure that only the numerator can be negative and the value is in least common denominator form:

2/-3 would be converted into -2/3 automatically;
15/21 would be converted into 5/7 automatically.

4. Write methods to convert between decimals and fractions.

Model Answer

Software Development Process

1. Problem statement

Write a program to construct fraction and the fraction could add, subtract, multiple and divide.

2. Analysis

Inputs:

The value of fraction (denominator and numerator).

Outputs:

The result of tow fractions' add, subtract, multiple, divide and compare.

Additional requirements or constraint

Only top and bottom in the class member.

3. Design

Algorithm

1. Adding "iostream" header files.
2. Using of the ste namespace.
3. Define a class which called Fraction

Private

int top – represent the numerator

int bottom – represent denominator

Public

Fraction() – default constructor

<1> let top equal to 0 and bottom equal to 1

Fraction(int N,intD=1) – normal constructor

<2> let top equal to N and bottom equal to D

void Output()- display the task result on screen

void output() – display the calculate result on screen

Fraction operator+(const Fraction & n) – re-define symbol+

<1> construct a new fraction t to receive data

<2> reduction of fractions to a common denominator

<3> calculate the value of top and bottom

<3> return the value of top and bottom

Fraction operator-(const Fraction & n) – re-define symbol-

The same as operator+

Fraction operator*(const Fraction & n) – re-define symbol*

multiple

<1> construct a new fraction t to receive data

<2> the result of numerator equal to two numerators' multiple

<3> the result of denominator equal to two denominators'

Fraction operator/(const Fraction & n) – re-define symbol/

The same as operator*

Fraction operator<(const Fraction & n) – re-define symbol<, could compare two fraction's value

<1> construct two new fractions to receive data

<2> reduction of fractions to a common denominator

<3> compare the value of numerator

<4> display the result

4. Write main function

<1> display the menu

<2> setting up a loop to let user choose function

<3> show tasks in one option

<4> add the calculate function to other options

4. Implementation:

See the C code in file exercise1.c with comments.

```
/*
Name: Simple Program to set fraction and calculate
File Name: EXERCISE 2.c
Copyright: Free
Author: Zhang Junming
Description: construct fraction and the fraction could add, subtract, multiple and divide
*/

#include <iostream>

using namespace std;

class Fraction
{
private:
    int top;        // Numerator
    int bottom;     // Denominator

public:
    Fraction() // default constructor
    {
        top = 0;
        bottom = 1;
    }

    Fraction(int N, int D = 1) // normal constructor
    {
        top = N;
        bottom = D;
    }

    void Output ()
    {
        cout << "represents " << top << "/" << bottom << endl;
    }
}
```

```

void output()
{
    cout << top << "/" << bottom << endl;
}

Fraction operator+(const Fraction & n) //re-define the symbol +, two fraction could use + to calculate directly
{
    Fraction t;
    t.top = ( top * n.bottom ) + ( bottom * n.top ); // the value of top after reduction of fractions to a common denominator
    t.bottom = bottom * n.bottom; // the value of bottom after reduction of fractions to a common denominator
    return t;
}

Fraction operator-(const Fraction & n) //same as last one
{
    Fraction t;
    t.top = ( top * n.bottom ) - ( bottom * n.top );
    t.bottom = bottom * n.bottom;
    return t;
}

Fraction operator*(const Fraction & n) //same as last one
{
    Fraction t;
    t.top = top * n.top;
    t.bottom = bottom * n.bottom;
    return t;
}

Fraction operator/(const Fraction & n) //same as last one
{
    Fraction t;
    t.top = top * n.bottom; // when divide one fraction, it equal to multiply by it's reciprocal
    t.bottom = bottom * n.top;
    return t;
}

Fraction operator>(const Fraction & n )
{
    Fraction t, d;
    t.top = top * n.bottom;
    t.bottom = bottom * n.bottom;
    d.top = n.top * bottom;
    d.bottom = n.bottom * bottom;
    // reduction of fractions to a common denominator

```

```

        if( t.top > d.top )
        {
            cout << top << "/" << bottom << " > " << n.top << "/" << n.bottom << endl;
            return n;
        }
        if( t.top == d.top )
        {
            cout << top << "/" << bottom << " = " << n.top << "/" << n.bottom << endl;
            return n;
        }
        if( t.top < d.top )
        {
            cout << top << "/" << bottom << " < " << n.top << "/" << n.bottom << endl;
            return n;
        }
    }

};

int main(void)
{
    int chioce;
    cout << "1-Task " << endl; // combine all task in part 1
    cout << "2-Fraction plus" << endl;
    cout << "3-Fraction minus" << endl;
    cout << "4-Fraction multiple" << endl;
    cout << "5-Fraction divide" << endl;
    cout << "6-Compare" << endl;
    cout << "7-Quit\n" << endl;
    do
    {
        cout << "\nPlease choose the function" << endl;
        cin >> chioce;
        if(chioce == 1)
        {

            cout << "Task\n" << endl;
            cout << "Fraction a; " << endl;
            Fraction a;
            a.Output();

            cout << "\nFraction b(3,4); " << endl;
            Fraction b(3,4);
            b.Output();

```

```

    cout << "\nFraction c(5); " << endl;
    Fraction c(5);
    c.Output();

    cout << "\nFraction d(2,3) test + - * / > function with Fraction b(3,4) " << endl;
    cout << "\nc + d = " << endl;
    Fraction d(2,3);
    Fraction e;
    e = b + d;
    e.output();

    cout << "\nc - d = " << endl;
    Fraction f;
    f = b - d;
    f.output();

    cout << "\nc * d = " << endl;
    Fraction g;
    g = b * d;
    g.output();

    cout << "\nc / d = " << endl;
    Fraction h;
    h = b / d;
    h.output();

    cout << "\nc compare d = " << endl;
    Fraction i;
    i = b > d;
}

if( chioce == 2 )// Function: add two fraction
{
    cout << "please enter the value of first Fraction" << endl;
    cout << "Numerator" << endl;
    int a;
    cin >> a;
    cout << "Denorminator" << endl;
    int b;
    cin >> b;

    cout << "please enter the value of second Fraction" << endl;
    cout << "Numerator" << endl;
    int c;
    cin >> c;
    cout << "Denorminator" << endl;
    int d;

```

```
        cin >> d;

        Fraction term1(a,b);
        Fraction term2(c,d);
        Fraction result;
        result = term1 + term2;// symbol + is re-define
        result.output();
    }

    if( chioce == 3 )// Function: minus two fraction
    {
        cout << "please enter the value of first Fraction" << endl;
        cout << "Numerator" << endl;
        int a;
        cin >> a;
        cout << "Denorminator" << endl;
        int b;
        cin >> b;

        cout << "please enter the value of second Fraction" << endl;
        cout << "Numerator" << endl;
        int c;
        cin >> c;
        cout << "Denorminator" << endl;
        int d;
        cin >> d;

        Fraction term1(a,b);
        Fraction term2(c,d);
        Fraction result;
        result = term1 - term2;// symbol - is re-define
        result.output();
    }

    if( chioce == 4 )// Function: multiple two fraction
    {
        cout << "please enter the value of first Fraction" << endl;
        cout << "Numerator" << endl;
        int a;
        cin >> a;
        cout << "Denorminator" << endl;
        int b;
        cin >> b;

        cout << "please enter the value of second Fraction" << endl;
        cout << "Numerator" << endl;
        int c;
```

```

        cin >> c;
        cout << "Denominator" << endl;
        int d;
        cin >> d;

        Fraction term1(a,b);
        Fraction term2(c,d);
        Fraction result;
        result = term1 / term2; // symbol / is re-define
        result.output();
    }

    if( chioce == 6 ) // compare two fraction
    {
        cout << "please enter the value of first Fraction" << endl;
        cout << "Numerator" << endl;
        int a;
        cin >> a;
        cout << "Denominator" << endl;
        int b;
        cin >> b;

        cout << "please enter the value of second Fraction" << endl;
        cout << "Numerator" << endl;
        int c;
        cin >> c;
        cout << "Denominator" << endl;
        int d;
        cin >> d;

        Fraction term1(a,b);
        Fraction term2(c,d);
        Fraction result;
        result = term1 > term2; // symbol > re-define to compare fractions
    }
    }

    while( chioce != 7 );

    return 0;
}

```

5. Testing:

```
1-Task
2-Fraction plus
3-Fraction minus
4-Fraction multiple
5-Fraction divide
6-Compare
7-Quit

Please choose the function
```

Menu

```
Please choose the function
1
Task

Fraction a;
represents 0/1

Fraction b(3,4);
represents 3/4

Fraction c(5);
represents 5/1

Fraction d(2,3) test + - * / > function with Fraction b(3,4)

c + d =
17/12

c - d =
1/12

c * d =
6/12

c / d =
9/8

c compare d =
3/4 > 2/3
```

Require of exercise 2

```
Please choose the function
2
please enter the value of first Fraction
Numerator
1
Denominator
3
please enter the value of second Fraction
Numerator
2
Denominator
5
11/15
```

Add function

```
Please choose the function
3
please enter the value of first Fraction
Numerator
1
Denominator
3
please enter the value of second Fraction
Numerator
2
Denominator
5
-1/15
```

Subtract function

```
Please choose the function
4
please enter the value of first Fraction
Numerator
1
Denominator
3
please enter the value of second Fraction
Numerator
2
Denominator
5
2/15
```

Multiple function


```
Please choose the function
5
please enter the value of first Fraction
Numerator
1
Denominator
3
please enter the value of second Fraction
Numerator
2
Denominator
5
5/6
```

Divide function

```
Please choose the function
6
please enter the value of first Fraction
Numerator
1
Denominator
3
please enter the value of second Fraction
Numerator
2
Denominator
5
1/3 < 2/5
```

Compare function

Exercise 3

Question

EXERCISE 3 (5 POINTS OUT OF 15)

Design a composite class represents complex numbers whose real and imaginary parts are Fractions.

1. Write appropriate constructors for this class;
2. Fractions should be able to add, subtract, multiple and divide.

Model Answer

Software Development Process

1. Problem statement

Write a program using composite class to calculate complex number's arithmetic.

3. Analysis

Inputs:

Two complex numbers' real part and imaginary part using fraction

Outputs:

The arithmetic result of two complex numbers

Additional requirements or constraint

Using composite class

3. Design

Algorithm

1. Adding "iostream" header files.
2. Using of the ste namespace.
3. Define a class which called Fraction

Private

int top – represent the numerator

int bottom – represent denominator

Public

Fracion(int N=0,int D=0) – normal constructor

<1> let top equal to 0 and bottom equal to 0

int Gettop() – to get the value of numerator

<1> return top

int Getbottom() – to get the value of denominator

<1> return bottom

Fraction Fadd (Fraction n) – Operator overloading Fadd

<1> construct a new Fraction t to receive data

<2> reduction of fractions to a common denominator

<3> calculate the value of top and bottom

<3> return the value of top and bottom

Fraction Fminus (Fraction n) – Operator overloading Fminus

The same as Fadd

Fraction Ftimes (Fraction n) – Operator overloading Ftimes

<1> construct a fraction t to receive data

<2> the result of numerator equal to two numerators' multiple

<3> the result of denominator equal to two denominators'

multiple

Fraction Fdivision (Fraction n) – Operator overloading Fdivision

The same as Ftimes

4. Define a class which called Fraction

Private

Fraction real – represent the fraction real part

Fraction imag – represent the fraction imaginary part

Public

complexClass() – default constructor

complexClass(Fraction r, Fraction i):real@,imag(i){}) – normal
composite constructor

complexClass Csum (complexClass n) – Operator overloading Csum

<1> construct a new complex t to receive data

<2> the sum of real part equal to the sum of two complex numbers'

fraction real part

<3> the sum of imaginary part equal to the sum of two complex

numbers' fraction imageinary part

<4> return the value of real and imag

complexClass Cminus (complexClass n) – Operator overloading Cminus

The same as Cminus

complexClass Ctimes (complexClass n) – Operator overloading Ctimes

<1> construct a new complex t to receive data

<2> multiple formula : $A*B=ac-bd+(ad+bc)i$

<3> return the value of real and imag

Fraction Fdivision (Fraction n) – Operator overloading Fdivision

<1> construct a new complex t to receive data

<2> division formula :

$$(a+bi)/(c+di)$$

$$=(a+bi)*(c-di)/(c+di)*(c-di)$$

$$=(ac+adi+bci+bd)/(c*c+d*d)$$

<3> return the value of real and imag

void output() – display the calculate result

5. Write main function

<1> display the menu

- 1 for complex number plus;
- 2 for complex number minus;
- 3 for complex number multiple;
- 4 for complex number divide
- 5 for quit

<2> setting up a loop to ask user choose function

<3> complex plus:

fraction.

1. ask user input the value of two complex number's
2. construct the related complex number
3. calculate the sum of two complex numbers
4. display the result

<4> complex minus:

fraction.

1. ask user input the value of two complex number's
2. construct the related complex number

3. calculate the sum of two complex numbers

4. display the result

<5> complex multiple:

fraction.

1. ask user input the value of two complex number's

2. construct the related complex number

3. calculate the sum of two complex numbers

4. display the result

<6> complex divide:

fraction.

1. ask user input the value of two complex number's

2. construct the related complex number

3. calculate the sum of two complex numbers

4. display the result

4. Implementation:

See the C code in file exercise1.c with comments.

```
/*  
  
Name: Simple Program to set complex number and calculate  
  
File Name: EXERCISE 3.cpp  
  
Copyright: Free  
  
Author: Zhang Junming  
  
Description: construct complex number and the complex number could add, subtract, multiple and divide  
*/  
  
#include <iostream>  
  
using namespace std;  
  
class Fraction  
{  
  
    private:  
  
        int top;        // Numerator  
        int bottom;     // Denominator  
  
    public:  
  
        Fraction(int N=0 , int D=0 )// normal constructor  
        {  
            top = N;  
            bottom = D;  
        }  
  
        int Gettop()// obtain the value of numerator  
        {  
            return top;  
        }  
  
        int Getbottom()// obtain the value of denominator  
        {  
            return bottom;  
        }  
}
```



```

Fraction Fadd(Fraction n)// Fraction addition
{
    Fraction t;
    t.top = ( top * n.bottom ) + ( bottom * n.top );// the value of top after reduction of fractions to a common denominator
    t.bottom = bottom * n.bottom;// the value of bottom after reduction of fractions to a common denominator
    return t;
}

Fraction Fminus(Fraction n)// Fraction subtraction
{
    Fraction t;
    t.top = ( top * n.bottom ) - ( bottom * n.top );
    t.bottom = bottom * n.bottom;
    return t;
}

Fraction Ftimes(Fraction n)// Fraction multiplication
{
    Fraction t;
    t.top = top * n.top;
    t.bottom = bottom * n.bottom;
    return t;
}

Fraction Fdivision(Fraction n)// Fraction division
{
    Fraction t;
    t.top = top * n.bottom;// when divide one fraction, it equal to multiply by it's reciprocal
    t.bottom = bottom * n.top;
    return t;
}

};

class complexClass
{
    private:

        Fraction real;// real part is fraction
        Fraction imag;// imaginary part is fraction

    public:
        complexClass() {}// default constructor
        complexClass(Fraction r , Fraction i):real(r),imag(i) {}// normal composite class constructor

```

```

}    complexClass Csum(complexClass n)// complex number addition
    {
        complexClass t;
        t.real = n.real.Fadd(real);// sum of real part = real part1 + real part2
        t.imag = n.imag.Fadd(imag);// sum of imaginary part = imaginary part1 + imaginary part2
        return t;
    }

}    complexClass Cminus(complexClass n)// complex subtraction
    {
        complexClass t;
        t.real = n.real.Fminus(real);// sum of real part = real part1 - real part2
        t.imag = n.imag.Fminus(imag);// sum of imaginary part = imaginary part1 - imaginary part2
        return t;
    }

}    complexClass Ctimes(complexClass n)//complex multiplication
    {
        complexClass t;
        t.real = (n.real.Ftimes(real)).Fminus(imag.Ftimes(n.imag));// A=(a+bi) B=(c+di)
        t.imag = (n.real.Ftimes(imag)).Fadd(n.imag.Ftimes(real));// A*B=ac-bd+(ad+bc)i
        return t;
    }

}    complexClass Cdivision(complexClass n)
    {
        complexClass t;
        t.real = ((n.real.Ftimes(real)).Fadd(n.imag.Ftimes(imag))).Fdivision((real.Ftimes(real)).Fadd(imag.Ftimes(imag))); // (a+bi)/(c+di)
        t.imag = ((n.imag.Ftimes(real)).Fminus(n.real.Ftimes(imag))).Fdivision((real.Ftimes(real)).Fadd(imag.Ftimes(imag)));// =(a+bi)*(c-di)/(c+di)*(c-di)
        return t; // =(ac+adi+bci+bd)/(c*c+d*d)
    }

}    void output()
    {
        cout << real.Gettop() << "/" << real.Getbottom() << " + " << imag.Gettop() << "/" << imag.Getbottom() << " i" << endl;
    }

};

int main(void)
{
    int chioce;
    cout << "1-Complex number plus " << endl;
    cout << "2-Complex number minus" << endl;
    cout << "3-Complex number multiple" << endl;
    cout << "4-Complex number divide" << endl;
}

```

```
cout << "5-Quit\n" << endl;

do
{
    cout << "\nPlease choose the function" << endl;
    cin >> chioce;

    switch(chioce)
    {
        case 1:{
            int a = 0, b = 0;
            cout << "Please enter the value of first complex number's real part's numerator" << endl;
            cin >> a;
            cout << "Please enter the value of first complex number's real part's denominator" << endl;
            cin >> b;
            Fraction term1(a,b);
            int c = 0, d = 0;
            cout << "Please enter the value of first complex number's imaginary part's numerator" << endl;
            cin >> c;
            cout << "Please enter the value of first complex number's imaginary part's denominator" << endl;
            cin >> d;
            Fraction term2(c,d);

            int e = 0, f = 0;
            cout << "Please enter the value of second complex number's real part's numerator" << endl;
            cin >> e;
            cout << "Please enter the value of second complex number's real part's denominator" << endl;
            cin >> f;
            Fraction term3(e,f);
            int g = 0, h = 0;
            cout << "Please enter the value of second complex number's imaginary part's numerator" << endl;
            cin >> g;
            cout << "Please enter the value of second complex number's imaginary part's denominator" << endl;
            cin >> h;
            Fraction term4(g,h);

            complexClass t1(term1,term2);
            complexClass t2(term3,term4);
            complexClass result;
            result = t2.Csum(t1); // result = t1 + t2
            result.output();

            break;
        }
    }
}
```

```

case 2:
{
    int a = 0, b = 0;
    cout << "Please enter the value of first complex number's real part's numerator" << endl;
    cin >> a;
    cout << "Please enter the value of first complex number's real part's denominator" << endl;
    cin >> b;
    Fraction term1(a,b);
    int c = 0, d = 0;
    cout << "Please enter the value of first complex number's imaginary part's numerator" << endl;
    cin >> c;
    cout << "Please enter the value of first complex number's imaginary part's denominator" << endl;
    cin >> d;
    Fraction term2(c,d);

    int e = 0, f = 0;
    cout << "Please enter the value of second complex number's real part's numerator" << endl;
    cin >> e;
    cout << "Please enter the value of second complex number's real part's denominator" << endl;
    cin >> f;
    Fraction term3(e,f);
    int g = 0, h = 0;
    cout << "Please enter the value of second complex number's imaginary part's numerator" << endl;
    cin >> g;
    cout << "Please enter the value of second complex number's imaginary part's denominator" << endl;
    cin >> h;
    Fraction term4(g,h);

    complexClass t1(term1,term2);
    complexClass t2(term3,term4);
    complexClass result;
    result = t2.Cminus(t1); // result = t1 - t2
    result.output();

    break;
}

case 3:{
    int a = 0, b = 0;
    cout << "Please enter the value of first complex number's real part's numerator" << endl;
    cin >> a;
    cout << "Please enter the value of first complex number's real part's denominator" << endl;
    cin >> b;
    Fraction term1(a,b);
    int c = 0, d = 0;
    cout << "Please enter the value of first complex number's imaginary part's numerator" << endl;
    cin >> c;

```

```

        cout << "Please enter the value of first complex number's imaginary part's denominator" << endl;
        cin >> d;
        Fraction term2(c,d);

        int e = 0,f = 0;
        cout << "Please enter the value of second complex number's real part's numerator" << endl;
        cin >> e;
        cout << "Please enter the value of second complex number's real part's denominator" << endl;
        cin >> f;
        Fraction term3(e,f);
        int g = 0,h = 0;
        cout << "Please enter the value of second complex number's imaginary part's numerator" << endl;
        cin >> g;
        cout << "Please enter the value of second complex number's imaginary part's denominator" << endl;
        cin >> h;
        Fraction term4(g,h);

        complexClass t1(term1,term2);
        complexClass t2(term3,term4);
        complexClass result;
        result = t2.Ctimes(t1); // result = t1 * t2
        result.output();

        break;
    }

```

case 4:

```

{
    int a = 0,b = 0;
    cout << "Please enter the value of first complex number's real part's numerator" << endl;
    cin >> a;
    cout << "Please enter the value of first complex number's real part's denominator" << endl;
    cin >> b;
    Fraction term1(a,b);
    int c = 0,d = 0;
    cout << "Please enter the value of first complex number's imaginary part's numerator" << endl;
    cin >> c;
    cout << "Please enter the value of first complex number's imaginary part's denominator" << endl;
    cin >> d;
    Fraction term2(c,d);

    int e = 0,f = 0;
    cout << "Please enter the value of second complex number's real part's numerator" << endl;
    cin >> e;

```

```

        cout << "Please enter the value of second complex number's real part's denominator" << endl;
        cin >> f;
        Fraction term3(e,f);
        int g = 0,h = 0;
        cout << "Please enter the value of second complex number's imaginary part's numerator" << endl;
        cin >> g;
        cout << "Please enter the value of second complex number's imaginary part's denominator" << endl;
        cin >> h;
        Fraction term4(g,h);

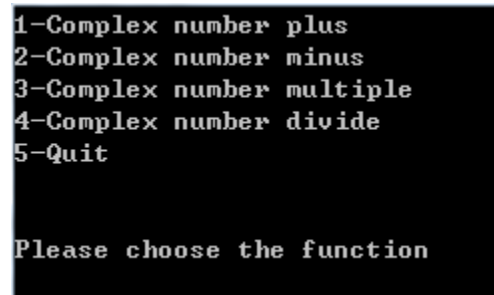
        complexClass t1(term1,term2);
        complexClass t2(term3,term4);
        complexClass result;
        result = t2.Cdivision(t1); //result = t1 / t2
        result.output();

        break;
    }
    case 5:{break;}
}
}
while(chioce != 5);

return 0;
}

```

5.Testing:



```

1-Complex number plus
2-Complex number minus
3-Complex number multiple
4-Complex number divide
5-Quit

Please choose the function

```

Menu

```

Please choose the function
1
Please enter the value of first complex number's real part's numerator
1
Please enter the value of first complex number's real part's denominator
3
Please enter the value of first complex number's imaginary part's numerator
2
Please enter the value of first complex number's imaginary part's denominator
5
Please enter the value of second complex number's real part's numerator
2
Please enter the value of second complex number's real part's denominator
5
Please enter the value of second complex number's imaginary part's numerator
1
Please enter the value of second complex number's imaginary part's denominator
3
11/15 + 11/15 i

```

Complex number addition

```

Please choose the function
2
Please enter the value of first complex number's real part's numerator
1
Please enter the value of first complex number's real part's denominator
3
Please enter the value of first complex number's imaginary part's numerator
2
Please enter the value of first complex number's imaginary part's denominator
5
Please enter the value of second complex number's real part's numerator
2
Please enter the value of second complex number's real part's denominator
5
Please enter the value of second complex number's imaginary part's numerator
1
Please enter the value of second complex number's imaginary part's denominator
3
-1/15 + 1/15 i

```

Complex number subtraction

```

Please choose the function
3
Please enter the value of first complex number's real part's numerator
1
Please enter the value of first complex number's real part's denominator
3
Please enter the value of first complex number's imaginary part's numerator
2
Please enter the value of first complex number's imaginary part's denominator
5
Please enter the value of second complex number's real part's numerator
2
Please enter the value of second complex number's real part's denominator
5
Please enter the value of second complex number's imaginary part's numerator
1
Please enter the value of second complex number's imaginary part's denominator
3
0/225 + 61/225 i

```

Complex number multiplication

```

Please choose the function
4
Please enter the value of first complex number's real part's numerator
1
Please enter the value of first complex number's real part's denominator
3
Please enter the value of first complex number's imaginary part's numerator
2
Please enter the value of first complex number's imaginary part's denominator
5
Please enter the value of second complex number's real part's numerator
2
Please enter the value of second complex number's real part's denominator
5
Please enter the value of second complex number's imaginary part's numerator
1
Please enter the value of second complex number's imaginary part's denominator
3
13500/13725 + 2475/13725 i

```

Complex number division