

ENG2002 Computer Programming

# Application Development Report

## Rational Numbers

### Group 3

HUANG Fei      13102666D

LIU Yicun      14110799D

Submission Date: December 6, 2016

# Table of Contents

<b>Abstract.....</b>	<b>1</b>
<b>1. Introduction.....</b>	<b>1</b>
<b>2. Methodology .....</b>	<b>3</b>
2.1 Work Allocation .....	3
2.2 Schedule .....	3
2.3 Structure .....	4
2.3.1 Specifications of classes .....	4
2.3.2 Flow of execution .....	9
2.4 Problems and Solutions .....	13
2.4.1 Login interface .....	13
2.4.2 Reduction process .....	13
2.4.3 Display of the output.....	13
2.4.4 Separation and combination.....	14
2.5 Testing .....	14
2.5.1 Login interface .....	14
2.5.2 Division.....	17
2.5.3 Exponential .....	19
<b>3. Results .....</b>	<b>20</b>
<b>4. Conclusion .....</b>	<b>22</b>

# Application Development Report

## Abstract

The overall objective of this assignment is to realize the calculation of rational numbers in an object-oriented console application. Login process of this application is well-designed. It can identify existed users, and check the passwords; in the case of new users, the application will require user names and passwords. The application uses numbers input by users to perform the operation, and returns the result in the lowest form of fractional number. It also checks the validity of the numbers every time before division operation. Furthermore, the application can perform exponential operation for rational numbers.

## 1. Introduction

Specifically, the objectives of this application are illustrated as follows.

- Designing class *RatNum* to process rational numbers in fraction format

A class called *RatNum* is required to build up. It is required to represent a rational number, in the form of fraction number  $a/b$ , where  $a$  and  $b$  are both integers.

Firstly, the class should be able to check whether  $b$  (i.e. the denominator) is zero in the assumption that  $b$  should not be less than zero. If  $b$  equals to zero, the users will get an alert of this error, and will be asked to re-select the value of  $b$ .

Secondly, the class should have member functions to realize the division operation of two rational numbers as defined above.

Thirdly, the class should be able to handle the exponential calculation of a rational number as defined above.

- Division for rational numbers in fraction format

Within the class *RatNum*, a series of member functions should be designed to handle the division operation, and return the result which is also in rational number format. Moreover, the result should be in its lowest form and cancel out all the

common factors between the numerator and the denominator. So, member functions to perform the reduction of a fractional number are also required.

- Exponential operation for rational numbers in fraction format

The exponential operation inside class *RatNum* is required to compute a rational number in fraction format to the power of  $n$ , where  $n$  is an integer. The function should firstly check if this rational number is zero. If it is zero, the function will do nothing and ask for selection again. The result of this exponential operation should also be in the lowest form, so the reduction is also designed here.

- Setting up the login platform for identification

The login platform is for users to start this application. Users will have two choices, sign up and sign in. If the user wants to sign in, the user will be asked for a username and a password, and then the name and password are stored into the text file for storage. If the user chooses to sign in, the login platform will ask the user to provide username and password, and will check if the name exists in the storage file. The signed-up users will only be permitted to this application if the password is correct, and each time there are only three chances to input password. The application will end if the user fails to provide correct password in three times. Another class called *user* is designed to perform this function.

- Providing the text-mode user interface to perform different operations

After the successful login, the menu will be prompted out to ask for the choices of the user. The user can choose to enter the rational number, perform division calculation, perform exponential calculation, or quit in the text-mode interface. The main menu will be repeatedly promoted out unless the user chooses to quit this application.

- Building up static libraries

Separate static libraries are required to be built up and linked to the console application. All the classes and implementation of the member functions should be included in static libraries.

## 2. Methodology

### 2.1 Work Allocation

We allocated the work well in order to let both of us fully participate in this Application Development Assignment. Most importantly, we understand how to cooperate well as programmers by work division. The details of our work allocation are listed as follows.

<b>Huang Fei</b>	<ul style="list-style-type: none"> <li>• Design of the programme flow and overall structure</li> <li>• Class User implementation</li> <li>• Class RatNum implementation</li> <li>• Modifications and debugging for the whole application</li> <li>• Report writing</li> </ul>
<b>Liu Yicun</b>	<ul style="list-style-type: none"> <li>• Design of the programme flow and overall structure</li> <li>• Class User modifications</li> <li>• Class RatNum implementation</li> <li>• Modifications and debugging for the whole application</li> <li>• Report writing</li> </ul>

### 2.2 Schedule

We started to discuss and think about the project around the beginning of Week 11, and finished around the end of Week 13. The detailed project schedule is demonstrated as follows.

<b>Week 10</b>	<ul style="list-style-type: none"> <li>• Planning and discussion about the requirements</li> <li>• Overall schedule of implementation</li> </ul>
<b>Week 11</b>	<ul style="list-style-type: none"> <li>• Discussion about the general programme flow</li> <li>• Work allocation</li> </ul>

<b>Week 12</b>	<ul style="list-style-type: none"> <li>• Coding and testing for the whole programme</li> <li>• Modifications for the optimal solution</li> <li>• Report writing</li> </ul>
<b>Week 13</b>	<ul style="list-style-type: none"> <li>• Further modifications for final version</li> <li>• Report writing</li> </ul>

## 2.3 Structure

### 2.3.1 Specifications of classes

#### class RatNum

```

{
public:
    RatNum(signed*a, signed*b); //get address, pass by reference
    void division(); //ask for the input of a rational number as the divisor and do division
    void getnum(); //ask for the input of a rational number and check its validity
    void reduction(); //reduction used n1 n2 as absolute value
    void result(); //show result according to different situation
    void current(); //show current rational number
    void power(); //exponential calculation, if original rational number is 0, ask input again

private:
    signed*a1; //store the first address
    signed*b1; //store the second address
    int n1; //store absolute value
    int n2; //store absolute value
    signed a2; //the first number of divisor
    signed b2; //the second number of divisor
    signed n = 1; //the power to the rational number, initialize as 1
};

```

Member functions/ Member variables	Implementation and Explanations
<code>signed*a1;</code> <code>signed*b1;</code>	//Store the address of two integers as the rational number in main(), pass by reference
<code>int n1;</code> <code>int n2;</code>	//Store the absolute value of the two integers as the rational number
<code>signed a2;</code> <code>signed b2;</code>	// Store the two integers input by user as the divisor
<code>signed n = 1;</code>	//exponent

<pre>RatNum(signed*a, signed*b);</pre>	<pre>RatNum::RatNum(signed*a, signed*b) //constructor, get address {     a1 = a;     b1 = b; //pass by reference }</pre>
<pre>void getnum();</pre>	<pre>void RatNum::getnum() {     //get rational number     bool checkb1 = false;     cout &lt;&lt; "Please input two integers to get the rational number: ";     do //if b1=0, ask input the rational number again     {         cin &gt;&gt; *a1;         cin &gt;&gt; *b1;         cin.get();         if (*b1 == 0)             cout &lt;&lt; "The second number cannot be 0. Input two integers again." &lt;&lt; endl;         else             checkb1 = true;     } while (checkb1 == false); //if b1!=0, the do while loop ends      if (*a1 == 0) //check a1 to adjust display         cout &lt;&lt; "The rational number is 0" &lt;&lt; endl; //if a1=0, display 0     else         cout &lt;&lt; "The rational number is " &lt;&lt; *a1 &lt;&lt; "/" &lt;&lt; *b1 &lt;&lt; endl; }</pre>
<pre>void division();</pre>	<pre>void RatNum::division() {     //get divisor     bool checka2 = false;     cout &lt;&lt; "Please input two integers to get the divisor: ";     do //if a2=0 or b2=0, ask user to input divisor again     {         cin &gt;&gt; a2;         cin &gt;&gt; b2;</pre>

```

        if (b2 == 0)
        {
            cout << "The denominator of the divisor cannot be 0. Please input
again!" << endl;
        }
        else if (a2 == 0)
        {
            cout << "The divisor cannot be 0. Please input again!" << endl;
        }
        else
        {
            checka2 = true;
        }
    } while (checka2 == false); //if a2!=0 and b2!=0, do while loop ends

                                //cross-multiplication
    *a1 = (*a1)*b2;
    *b1 = (*b1)*a2; //update *a1, *b1 as the division operation
                    //reduction begin after this
}

```

```
void power();
```

```

void RatNum::power()
{
    if (*a1 == 0) //the rational number cannot be 0
    {
        do
        {
            cout << "Wrong input, the rational number should not be zero!\n";
            getnum(); //call the getnum() function to input a rational number
again
        } while (*a1 == 0); //when the number not 0 do while loop end
    }
    signed reslta = 1, resltb = 1; //store the output for exponential
function
    cout << "Please input an integer as the exponent: ";
    cin >> n; //cin the exponent
    if (n == 0)
    {
        reslta = 1;
        resltb = 1; //if n=0, result is 1
    }
    else if (n > 0)
    {
        for (int i = 0; i < n; i++)
        {

```



```

        reslta = reslta*(*a1);
        resltb = resltb*(*b1); //when n>0,do normal exponention
    }
}
else
{
    for (int i = 0; i < (0 - n); i++)
    {
        reslta = reslta*(*b1);
        resltb = resltb*(*a1); //when n<0, the result should be the
                                //reciprocal of the original number
    }
}
*a1 = reslta;
*b1 = resltb; //update the values of *a1, *b1, for futher reduction and
display results
}

```

```
void reduction();
```

```

void RatNum::reduction()
{
    int k; //to store the common factor of |*a1|, |*b1|

    //get absolute value
    if (*a1 < 0)
        n1 = (-*a1);
    else if (*a1 > 0)
        n1 = *a1; //let n1 store the absolute value of a1
    if (*b1 < 0)
        n2 = (-*b1);
    else if (*b1 > 0)
        n2 = *b1; //let n2 store the absolute value of b1
    //initialize the value of k as the the smaller one of n1,
n2
    if (n1 >= n2)
    {
        k = n2;
    }
    else
    {
        k = n1;
    }

    //reduction begins here
    while (k > 1) //the smallest value of k should be 2
    {
        if ((*a1%k == 0) && (*b1%k == 0)) //if k is common factor

```

	<pre>         {             *a1 = *a1 / k; //a1 divided by k             *b1 = *b1 / k; //b1 divided by k             break;         }         k--;     } } </pre>
<pre>void result();</pre>	<pre> void RatNum::result() {     //use this function adjust display of result     if (*a1 == 0) //0 result         cout &lt;&lt; "The result is " &lt;&lt; 0 &lt;&lt; endl;     else //result is not 0 or 1     {         if (*b1 &lt; 0) //if *b1&lt;0, change the sign of the two integers as display             cout &lt;&lt; "The result is " &lt;&lt; 0 - *a1 &lt;&lt; "/" &lt;&lt; 0 - *b1 &lt;&lt; endl;         else //else, display normal result             cout &lt;&lt; "The result is " &lt;&lt; *a1 &lt;&lt; "/" &lt;&lt; *b1 &lt;&lt; endl;     } } </pre>
<pre>void current();</pre>	<pre> void RatNum::current() {     //same with function result, display current rational number     if (*a1 == 0)         cout &lt;&lt; "Current rational number is 0" &lt;&lt; endl;     else     {         if (*b1 &lt; 0)             cout &lt;&lt; "Current rational number is " &lt;&lt; 0 - *a1 &lt;&lt; "/" &lt;&lt; 0 - *b1 &lt;&lt; endl;         else             cout &lt;&lt; "Current rational number is " &lt;&lt; *a1 &lt;&lt; "/" &lt;&lt; *b1 &lt;&lt; endl;     } } </pre>

```

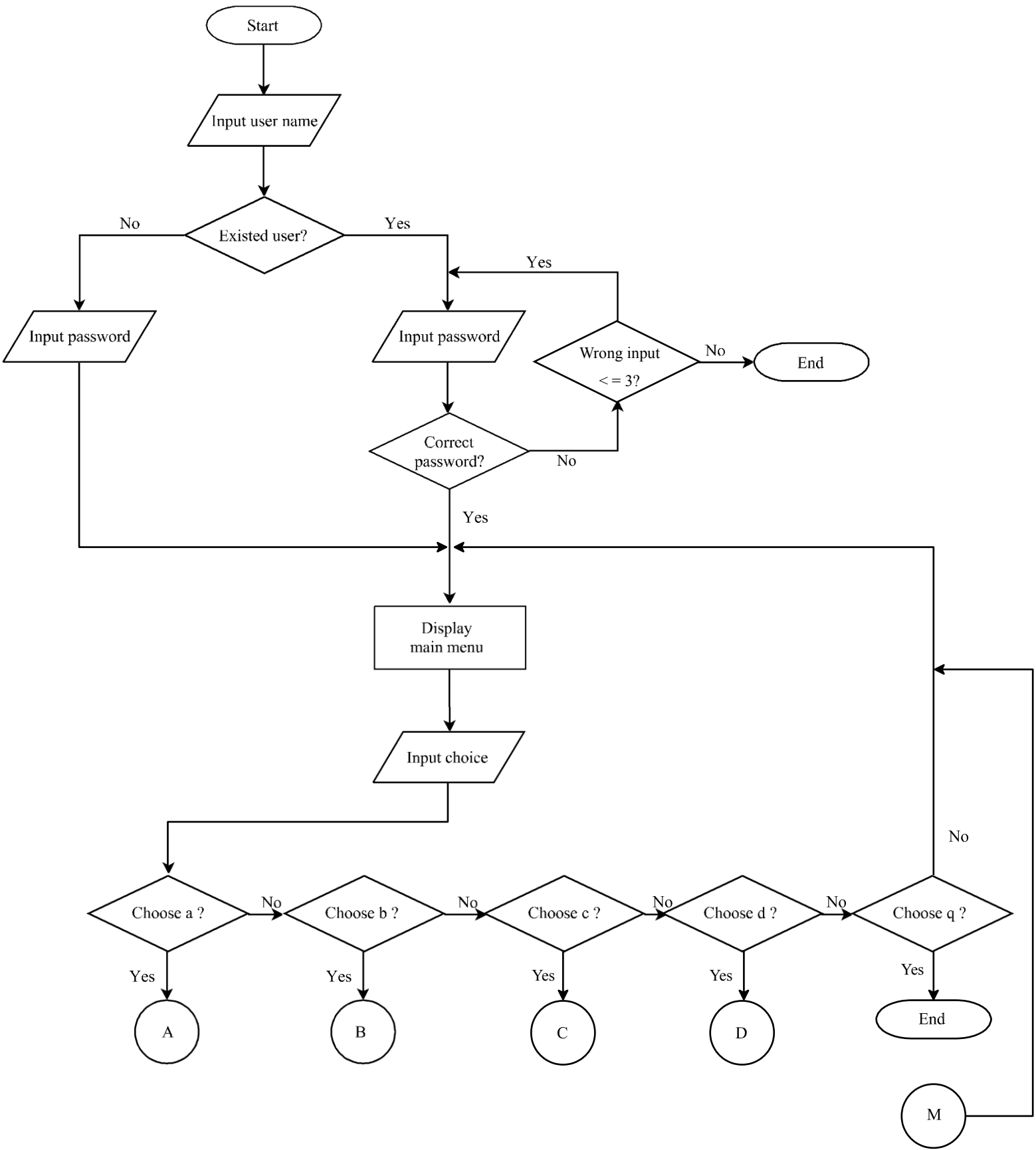
class user
{
public:
    void initial(char* a, char *b); //store information in a element
    char *nam(); //return name
    char *pas(); //return password
private:
    char name[20]; //store the name from file
    char password[20]; //store the password from file
};

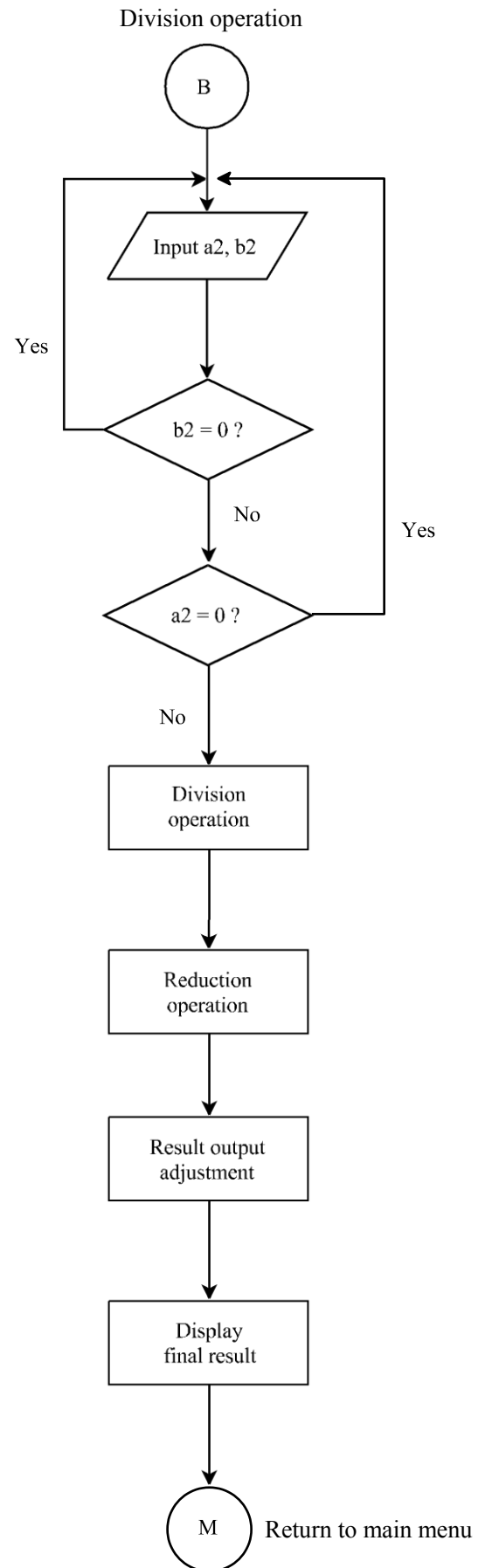
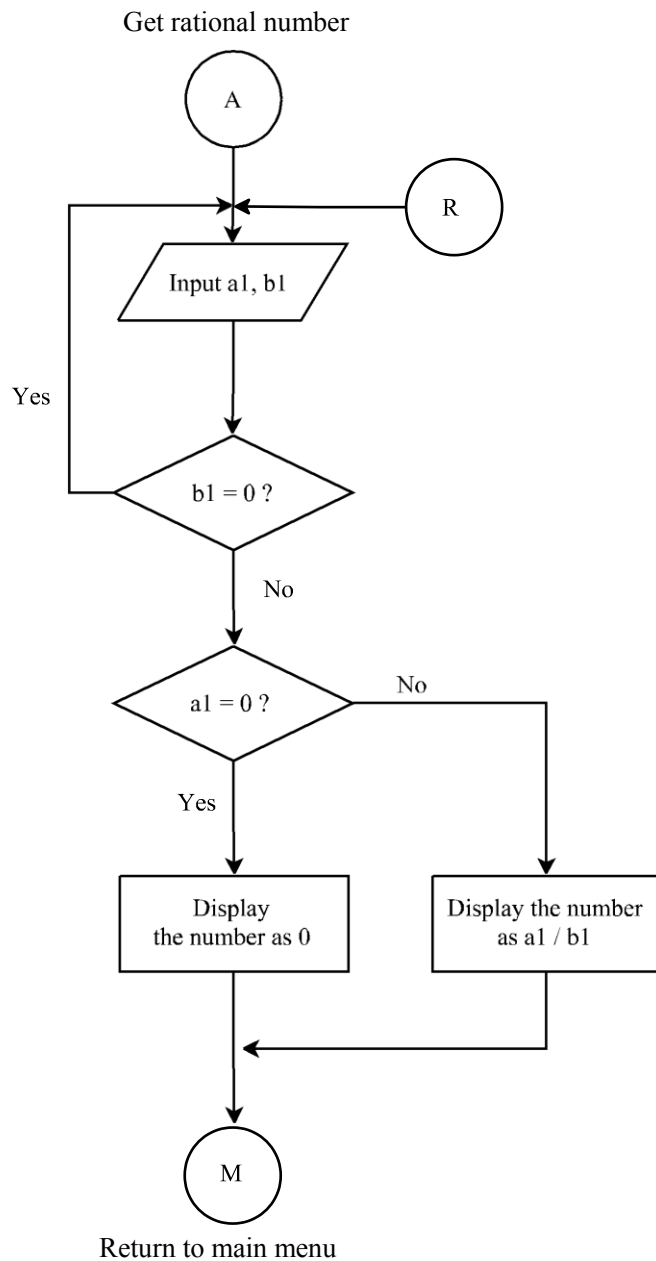
```

Member functions/ Member variables	Implementation and Explanations
<code>char name[20];</code>	<code>//To store the user's name input from file</code>
<code>char password[20];</code>	<code>//To store the user's password input from file</code>
<code>void initial(char* a, char *b);</code>	<pre> void user::initial(char*a, char *b) {     strncpy(name, a, 20); //store the name     strncpy(password, b, 20); //store the password } </pre>
<code>char *nam();</code>	<pre> char*user::nam() {     return name; //return name store in this object } </pre>
<code>char *pas();</code>	<pre> char*user::pas() {     return password; //return password store in this object } </pre>

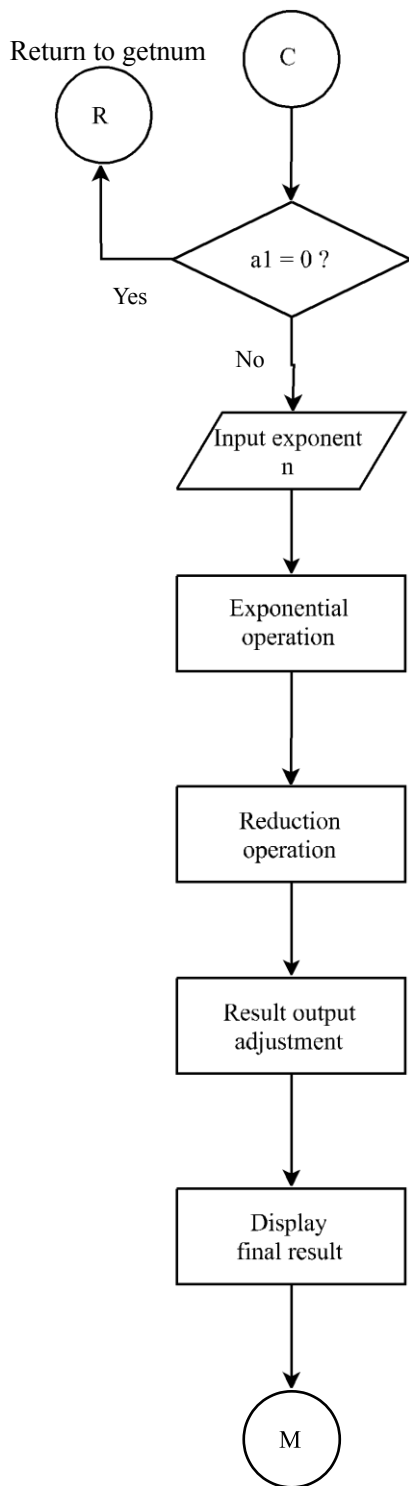
### 2.3.2 Flow of execution

The flowcharts for the design of the main, and all the cases are demonstrated as follows.



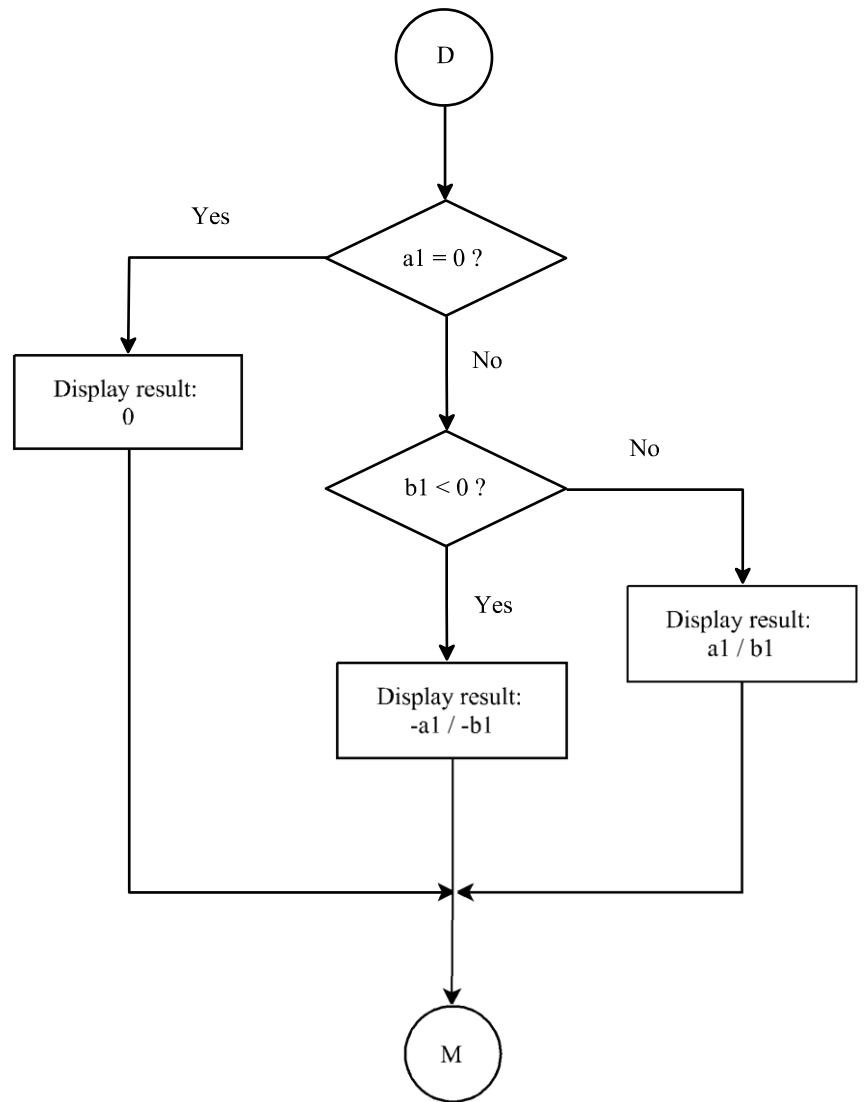


### Exponential Operation



Return to main menu

### Display the result



Return to main menu

## **2.4 Problems and Solutions**

The development of this application is successful with the excellent cooperation and knowledge of our team, but we also encountered some problems at first.

### **2.4.1 Login interface**

At first, we tried to input the information of users from the txt file directly. However, we found if we input information directly and search it, the structure will be too complex and tedious. Because of that, we planned to develop a new class, and used pointer of array to create an array with objects from this new class to store information input from the txt file. The *User* class is developed to achieve the login and sign up operation for users. With this class, the input information will be store. The searching operation would be easy to achieve with a for loop. This method can improve the structure, and achieve a better solution

### **2.4.2 Reduction process**

The reduction process is one of the most basic operation in mathematics, while it can be complicated when operated in the computer.

We firstly used for loop to find the common divisor of the two number, and categorized different situations by comparing the absolute values of the numerator and the denominator. We used % operation to find their common factors starting from 2. However, we found it quite lengthy and inflexible. In order to get an optimal solution, we changed the for loop into the while loop, and started from the smaller absolute value between denominator and numerator to find their greatest common divisor. Once it is found, the loop ends and the lowest form is also found.

### **2.4.3 Display of the output**

In order to display the results in a better way, we tried to make adjustments based on the positive or negative sign of the denominator and introduced several situations for discussion.

Later, we found that it is not the easiest way to get the perfect result. In fact, we can use the properties of signed integers in C++ language. If the denominator is negative, we just need to change the signs of the numerator and denominator at the same time, and at that time the sign of this fractional number is not changed. So, after discussing the case that the exponent is 0, and the case that dividend is 0, we can simply use “cout << “The result is ” << \*a1 << “/” << \*b1 << endl;” to display the final output on the screen.

#### 2.4.4 Separation and combination

Inside the class RatNum, we originally planned to define 4 member functions, including RatNum(signed\*a, signed\*b), void getnum(), void division(), and void power(). However, we found that the reduction process is required in both void division() and void power(), and the process of displaying the results is similar in both of them.

Thus, we decided to separate calculation, reduction and displaying results inside void division() and void power(), and to combine the similar processing for them. Finally, there are 3 more member functions inside class RatNum, which are void reduction(), void result(), and void current(). The main menu will call the calculation function, void division() or void power(), and then call void reduction() and void result(). Meanwhile, void current() is added in order for checking and convenience.

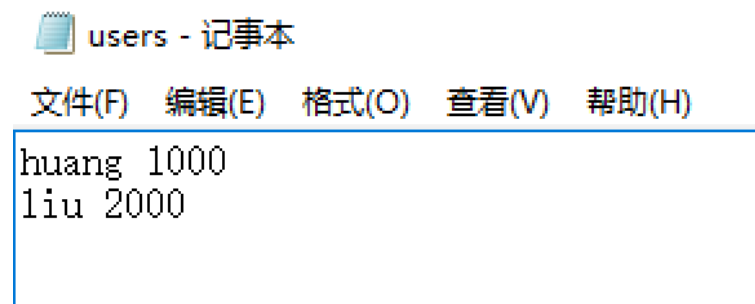
### 2.5 Testing

As we developed the application part by part by using the “divide and conquer” method, we tested our whole application part by part. The following screenshots demonstrates some special cases we used when testing in different parts.

#### 2.5.1 Login interface

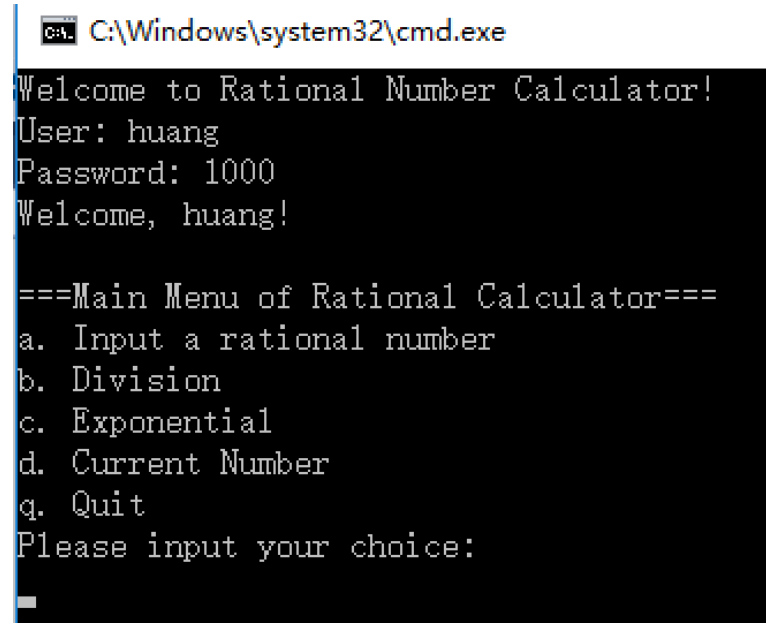


The user and password stored in txt file



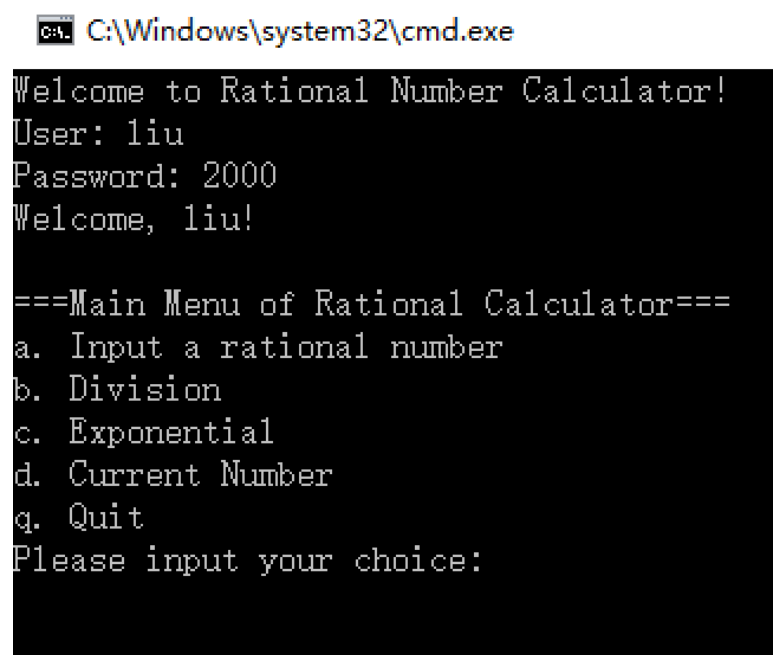
```
users - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
huang 1000
liu 2000
```

Testing login in with exist user



```
C:\Windows\system32\cmd.exe
Welcome to Rational Number Calculator!
User: huang
Password: 1000
Welcome, huang!


===Main Menu of Rational Calculator===
a. Input a rational number
b. Division
c. Exponential
d. Current Number
q. Quit
Please input your choice:
_
```



```
C:\Windows\system32\cmd.exe
Welcome to Rational Number Calculator!
User: liu
Password: 2000
Welcome, liu!


===Main Menu of Rational Calculator===
a. Input a rational number
b. Division
c. Exponential
d. Current Number
q. Quit
Please input your choice:
```

Testing login in with new user

 C:\Windows\system32\cmd.exe

```
Welcome to Rational Number Calculator!
User: new
Welcome, new user! Please enter a password: 3000

===Main Menu of Rational Calculator===
a. Input a rational number
b. Division
c. Exponential
d. Current Number
q. Quit
Please input your choice:
```


 users - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

---

huang 1000  
liu 2000  
new 3000

Testing login in with existed user but input wrong password for 3 times

 C:\Windows\system32\cmd.exe

```
Welcome to Rational Number Calculator!
User: huang
Password: 1111
Invalid password! 2 chances left.
1111
Invalid password! 1 chances left.
1111
Invalid password! 0 chances left.
Program ends!
请按任意键继续. . .
```

### 2.5.2 Division

Testing the denominator is 0

```
===Main Menu of Rational Calculator===
a. Input a rational number
b. Division
c. Exponential
d. Current Number
q. Quit
Please input your choice:
b
Please input two integers to get the rational number: 8 0
The second number cannot be 0. Input two integers again.
```

Testing the rational number is 0

```
===Main Menu of Rational Calculator===
a. Input a rational number
b. Division
c. Exponential
d. Current Number
q. Quit
Please input your choice:
a
Please input two integers to get the rational number: 0 8
The rational number is 0
```

Testing the divisor is 0

```
Please input two integers to get the rational number: 2 5
The rational number is 2/5

===Main Menu of Rational Calculator===
a. Input a rational number
b. Division
c. Exponential
d. Current Number
q. Quit
Please input your choice:
b
Please input two integers to get the divisor: 0 8
The divisor cannot be 0. Please input again!
```

Testing the denominator of divisor is 0

```
Please input two integers to get the rational number: 2 5
The rational number is 2/5

===Main Menu of Rational Calculator===
a. Input a rational number
b. Division
c. Exponential
d. Current Number
q. Quit
Please input your choice:
b
Please input two integers to get the divisor: 8 0
The denominator of the divisor cannot be 0. Please input again!
```

Testing the negative divided number

```
Please input two integers to get the rational number: -2 5
The rational number is -2/5

===Main Menu of Rational Calculator===
a. Input a rational number
b. Division
c. Exponential
d. Current Number
q. Quit
Please input your choice:
b
Please input two integers to get the divisor: 2 1
The result is -1/5
```

Testing the negative divided number and negative divisor

```
Please input two integers to get the rational number: -2 5
The rational number is -2/5

===Main Menu of Rational Calculator===
a. Input a rational number
b. Division
c. Exponential
d. Current Number
q. Quit
Please input your choice:
b
Please input two integers to get the divisor: -2 1
The result is 1/5
```

### 2.5.3 Exponential

Testing the exponent is 0

```
Please input your choice:
a
Please input two integers to get the rational number: 2 5
The rational number is 2/5

===Main Menu of Rational Calculator===
a. Input a rational number
b. Division
c. Exponential
d. Current Number
q. Quit
Please input your choice:
c
Please input an integer as the exponent: 0
The result is 1/1
```

Testing the exponent is 1

```
Please input your choice:
a
Please input two integers to get the rational number: 2 5
The rational number is 2/5

===Main Menu of Rational Calculator===
a. Input a rational number
b. Division
c. Exponential
d. Current Number
q. Quit
Please input your choice:
c
Please input an integer as the exponent: 1
The result is 2/5
```

Testing the negative exponent

```
Please input two integers to get the rational number: 2 5
The rational number is 2/5

===Main Menu of Rational Calculator===
a. Input a rational number
b. Division
c. Exponential
d. Current Number
q. Quit
Please input your choice:
c
Please input an integer as the exponent: -2
The result is 25/4
```

Testing when reduction needed

```
Please input two integers to get the rational number: 4 8
The rational number is 4/8

===Main Menu of Rational Calculator===
a. Input a rational number
b. Division
c. Exponential
d. Current Number
q. Quit
Please input your choice:
c
Please input an integer as the exponent: 2
The result is 1/4
```

### 3. Results

Following screenshots are captured when running the application using “Start without debugging” inside Microsoft Visual Studio.

ca. C:\Windows\system32\cmd.exe

Welcome to Rational Number Calculator!

User: huang

Password: 1000

Welcome, huang!

===Main Menu of Rational Calculator===

- a. Input a rational number
- b. Division
- c. Exponential
- d. Current Number
- q. Quit

Please input your choice:

a

Please input two integers to get the rational number: 6 9

The rational number is 6/9

===Main Menu of Rational Calculator===

- a. Input a rational number
- b. Division
- c. Exponential
- d. Current Number
- q. Quit

Please input your choice:

b

Please input two integers to get the divisor: 3 -2

The result is -4/9

```

===Main Menu of Rational Calculator===
a. Input a rational number
b. Division
c. Exponential
d. Current Number
q. Quit
Please input your choice:
c
Please input an integer as the exponent: -3
The result is -729/64

===Main Menu of Rational Calculator===
a. Input a rational number
b. Division
c. Exponential
d. Current Number
q. Quit
Please input your choice:
q
Goodbye!
请按任意键继续. . . █

```

## 4. Conclusion

Through this application development assignment, we gained much and got a deeper understanding of C++ Language.

- Knowledge

This project tests our knowledge learned in this semester. We got better understanding and gained more experiences towards the basic statements and operations in C++, class and objects, pointers and arrays, ifstream and ofstream, and input/output concepts.

- Programme design

We began to understand better about the programme design since we started from zero to develop this application. We understood the importance of clear logic flow and the help of flowcharts. Meanwhile, the “conquer and divide” method also plays a significant role in developing the programme, which left a deep impression



for us in the process of finishing this project.

- Cooperation

We firstly cooperate as programmers, and began to understand the way for a better cooperation as well as its requirements. We believe that adding comments, the duty of every programmer, is of great importance, which cannot be fully understood when just learning and practicing by oneself.

- Optimal solution

We encountered problems when finding the optimal solution. Sometimes the way of designing the structure is not the easiest or is not straight forward enough. If the structure is not the most concise one, there will be more problems when testing, debugging, and when the programme will also be read and used by other programmers.

Furthermore, there are also some aspects that we can further improve if there is more time, which are illustrated as follows.

- Number of Users

There is a limitation that the maximum number of user accounts can be stored in this programme is 60. If there are more than 60 users, new users' information cannot be added as existing account. We would like to develop more functions in class *User* to delete exceeding user accounts from the file, and display warning message to users when the number reaches the maximum.

- Display

There is a problem that when the programme displays an integer, it can only show the type as rational number rather than an integer, though it is requires by this assignment. For example, if the result is  $-2$ , the programme can only show  $-2/1$ . We think it would be better to adjust the display code to let the programme can display when the result is an integer.