



Xi'an Jiaotong-Liverpool University

西交利物浦大學

Department of Electrical and Electronic Engineering

EEE102 C++ Programming and Software Engineering

Assignment4 Inheritance and Polymorphism¹
SDP Report

Name : Zheng Sun

ID Number : 1507820

I certify that I have read and understood the University's *Policy for dealing with Plagiarism, Collusion and the Fabrication of Data* With reference to this policy, I certify that:

- My work does not contain any instances of plagiarism and/or collusion.
- My work does not contain any fabricated data.

By handing in my assignment for marking, I formally declare that all the above information is true to the best of my knowledge and belief.

Signature: Zheng Sun

¹Last updated on April 18, 2017

1. Problem Statement:

This task aims to test the basic concepts of Class Inheritance and to reinforce the understanding of OOP by practice code reuse. Overall, the object of the assignment is clear and lucid.

a) Derived class **iFraction** inherited from **Fraction** should be designed data of mixed fractions consisting 2 parts – the **Fraction** base type which is reduced to proper fraction (denominator larger than the nominator) and the **int** type integer part.

b) External function **convertF()** which convert improper functions to mixed fractions, as stipulated, should be the friend function of **Fraction**. Meanwhile, since the return type of **convertF()** is **iFraction**, the principle of forward declaration should be followed.

2. Analysis:

a) Inputs:

5 integers of int type → use **intParser()** to deal with erroneous input:

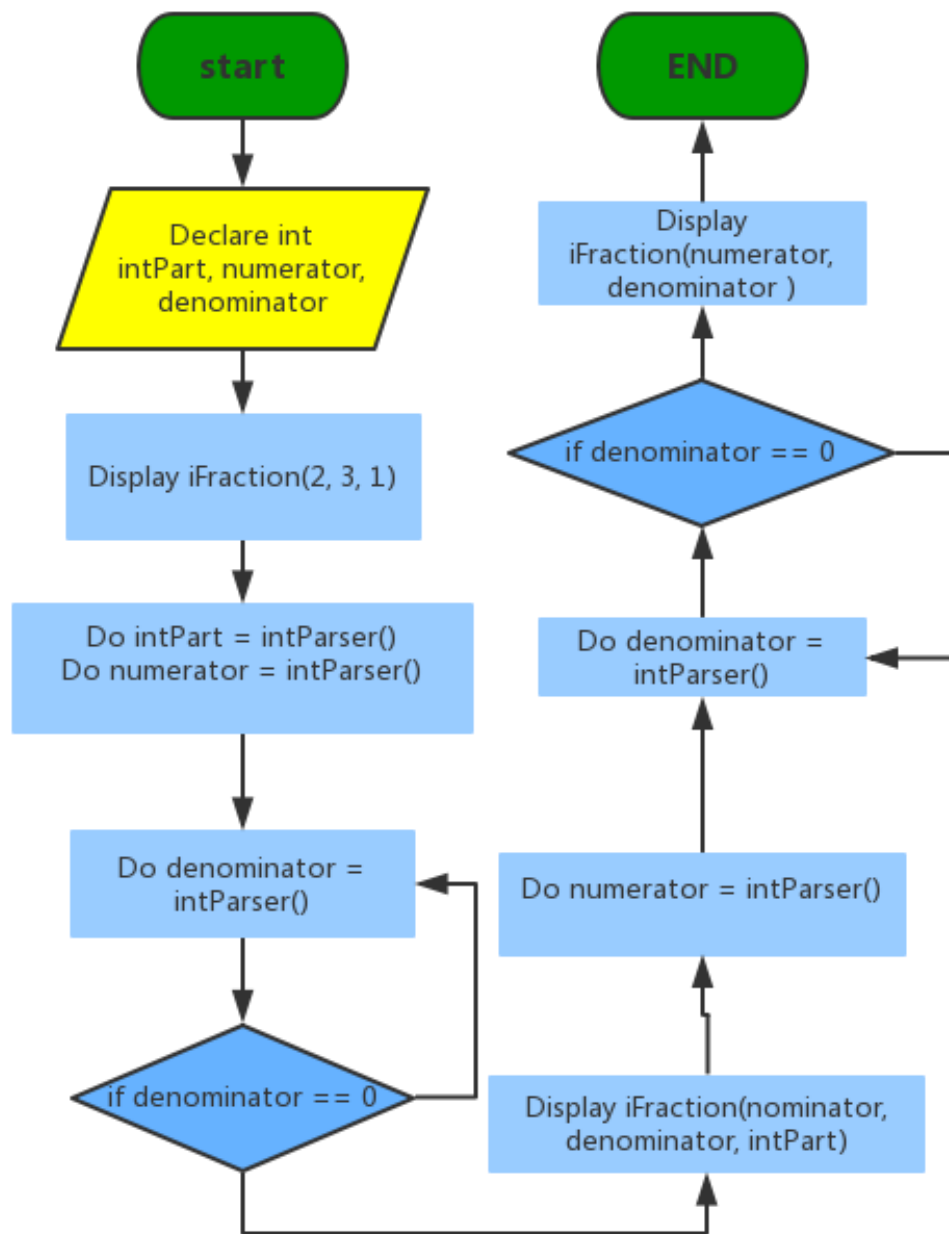
1. former 3 integers construct an **iFraction** using constructor directly;
2. latter 2 integers construct a **Fraction** and then implement **convertF()** to convert it to **iFraction**.

b) Output:

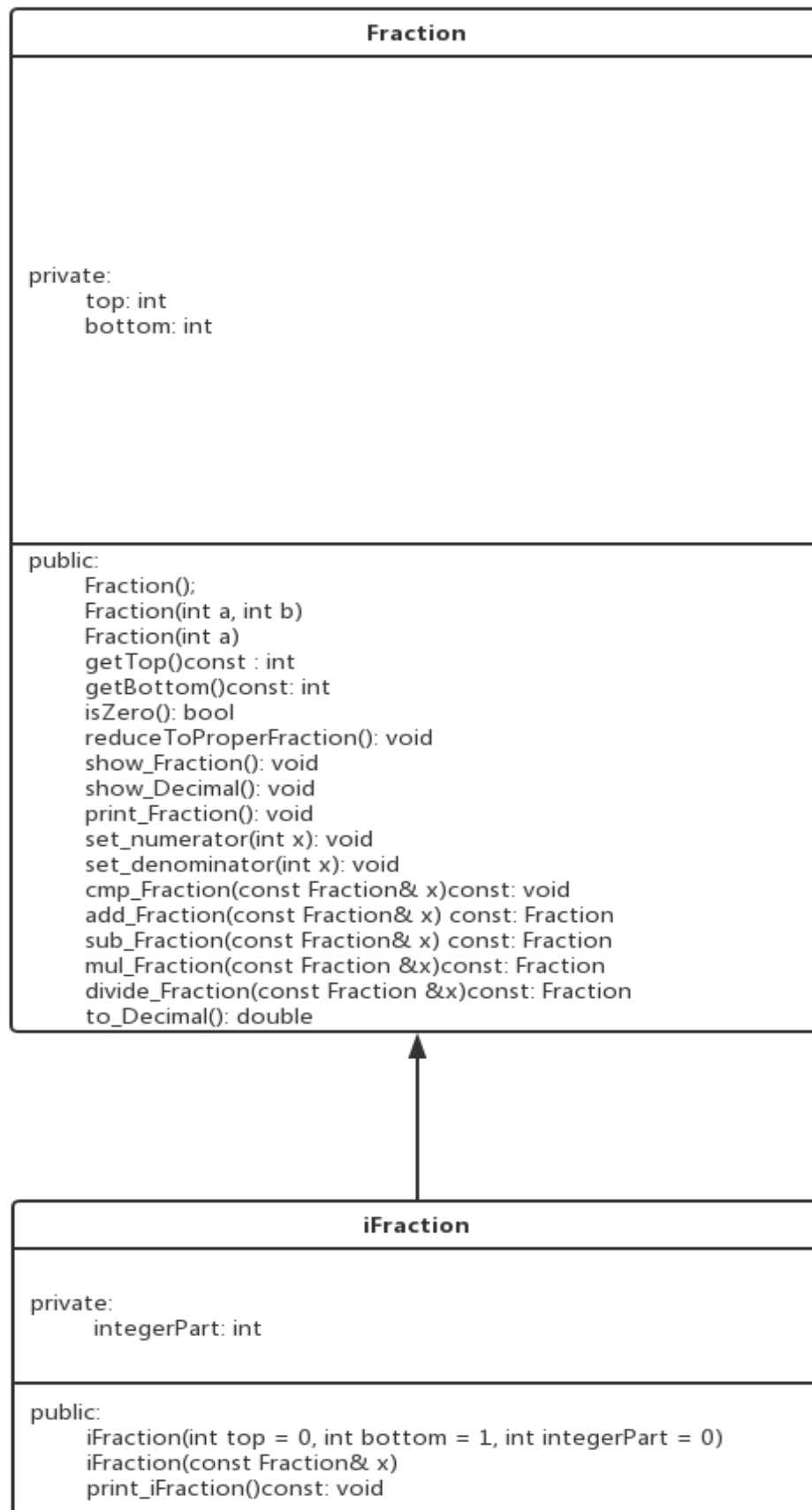
1. Interactive output will prompt the user to enter integers for demonstration.
2. Informative output given by class methods of **Fraction** and **iFraction** will show the results of the implementation.

3. Design

a) main() function design



b) Class Design: Base class: Fraction <- Subclass: iFraction



4. Implementation:

EEE102_Assessment_4\Task1\Fraction.h

EEE102_Assessment_4\Task1\Fraction.cpp

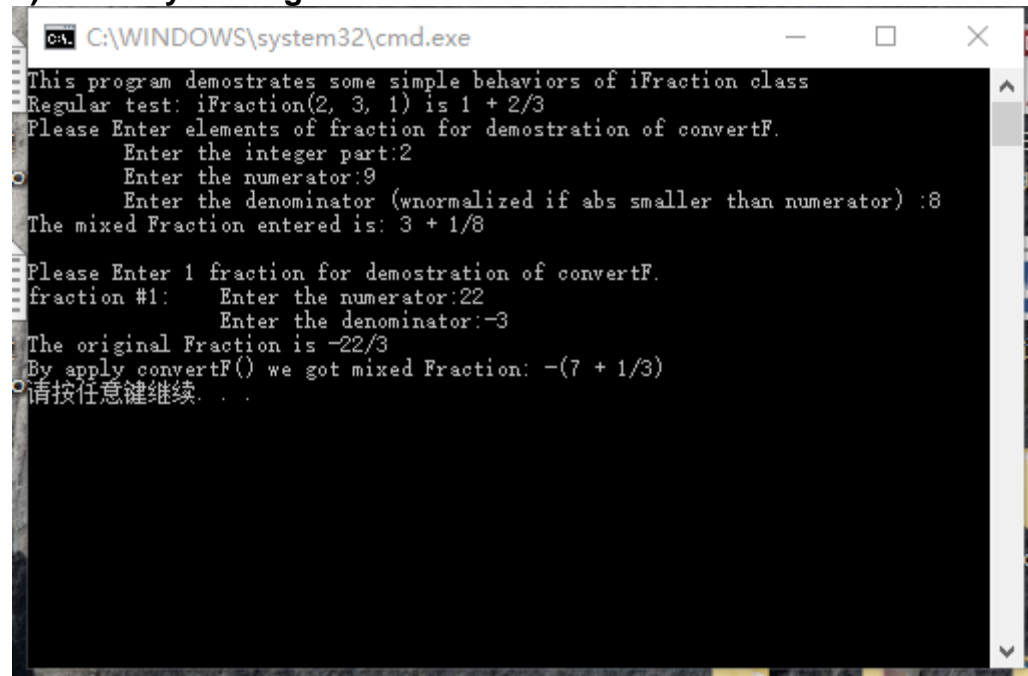
EEE102_Assessment_4\Task1\iFraction.h

EEE102_Assessment_4\Task1\iFraction.cpp

EEE102_Assessment_4\Task1\source.cpp

5. Testing

a) Ordinary Testing

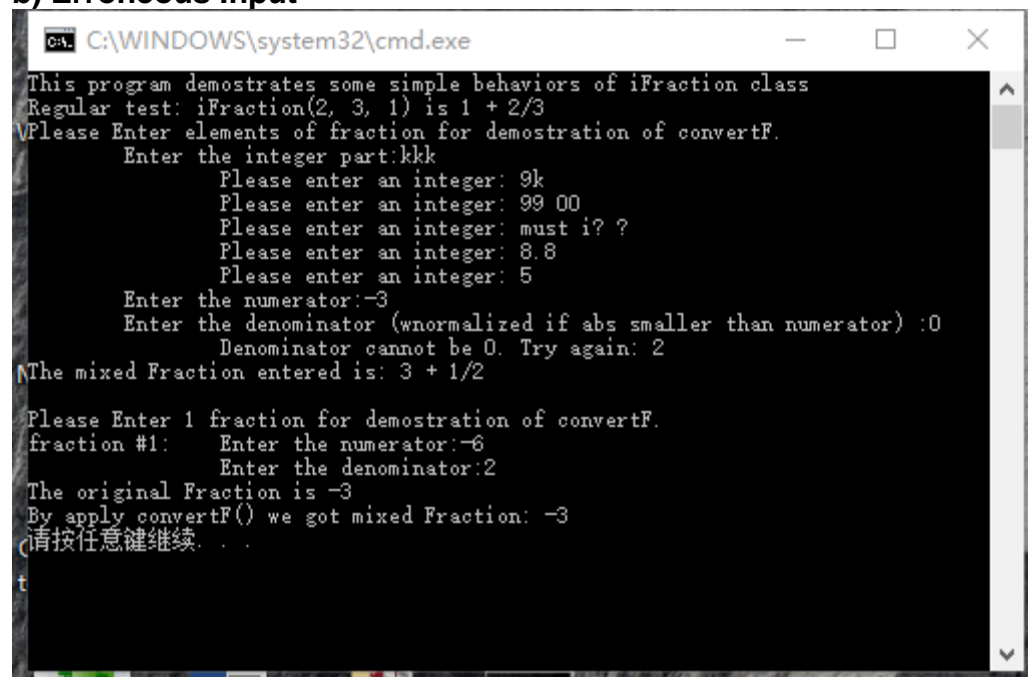


```
C:\WINDOWS\system32\cmd.exe

This program demonstrates some simple behaviors of iFraction class
Regular test: iFraction(2, 3, 1) is 1 + 2/3
Please Enter elements of fraction for demonstration of convertF.
    Enter the integer part:2
    Enter the numerator:9
    Enter the denominator (wnormalized if abs smaller than numerator) :8
The mixed Fraction entered is: 3 + 1/8

Please Enter 1 fraction for demonstration of convertF.
fraction #1:    Enter the numerator:22
               Enter the denominator:-3
The original Fraction is -22/3
By apply convertF() we got mixed Fraction: -(7 + 1/3)
请按任意键继续 . . .
```

b) Erroneous Input



```
C:\WINDOWS\system32\cmd.exe

This program demonstrates some simple behaviors of iFraction class
Regular test: iFraction(2, 3, 1) is 1 + 2/3
Please Enter elements of fraction for demonstration of convertF.
    Enter the integer part:kkk
        Please enter an integer: 9k
        Please enter an integer: 99 00
        Please enter an integer: must i? ?
        Please enter an integer: 8.8
        Please enter an integer: 5
    Enter the numerator:-3
    Enter the denominator (wnormalized if abs smaller than numerator) :0
        Denominator cannot be 0. Try again: 2
The mixed Fraction entered is: 3 + 1/2

Please Enter 1 fraction for demonstration of convertF.
fraction #1:    Enter the numerator:-6
               Enter the denominator:2
The original Fraction is -3
By apply convertF() we got mixed Fraction: -3
请按任意键继续 . . .
```