CISS245: Advanced Programming
Assignment 2

Name: _____

Objectives

1. Review of CISS240.

For this assignment, you will implement a library for modeling fractions. Each fraction is modeled (of course) with two integer variables (to model the numerator and denominator). The ideas present in this assignment will be used again in a later assignment when you "combine" two integer variable into a single variable.

The questions build on top of each other until the last one. Therefore you should start with Q1, when you are done with Q1, copy the code to Q2, etc. The first 8 questions are very simple. In fact the first few questions already appear in the review set. You will then use the library to factorize polynomials of degree 3 (the last two questions). The only one that really requires more work is Q10.

Q1. The follow skeleton code is given. There are three files. Note that the files are (of course) incomplete not just because the functions are not implemented. The preprocessor directives (i.e., `ifndef`, ...) are not included: you must complete them too.

a02q01/skel/main.cpp

```
// File  : main.cpp
// Author: smaug

#include <iostream>
#include "Fraction.h"

void test_Fraction_print()
{
    int xn = 0, xd = 0; // numerator and denominator of a fraction
    std::cin >> n >> d;
    Fraction_print(xn, xd);
    std::cout << std::endl;
}


int main()
{
    int option;
    std::cin >> option:
    switch (option)
    {
        case 1:
            test_Fraction_print();
            break;
    }

    return 0;
}
```

a02q01/skel/Fraction.h

```
// File: Fraction.h
// Author: smaug

//--------------------------------------------------------------------------
// Print fraction modeled by numerator n and denominator d.
//--------------------------------------------------------------------------
void Fraction_print(int, int);
```

a02q01/skel/Fraction.cpp

```
// File: Fraction.cpp
// Author: smaug

#include <iostream>


void Fraction_print(int n, int d)
{
}
```

Q2. This is a continuation of Q1. (In other words, after you finished Q1, copy the files from Q1 to the directory/folder for Q2.)

Now include a function to add fractions in your library. Test it throughly. I will not supply test cases since you should know how to add fractions!

See the skeleton code below for the prototype.

Note that, as stated in the comments for the function to add fractions, the add function does *not* simplify the fraction, i.e., it does *not* perform reduction of the fraction by divide the numerator and denominator by the greatest common divisor.

a02q02/skel/main.cpp

```
// File  : main.cpp
// Author: smaug

#include <iostream>
#include "Fraction.h"

void test_Fraction_print()
{
    int xn = 0, xd = 0; // numerator and denominator of a fraction

    std::cin >> xn >> xd;
    Fraction_print(xn, xd);
    std::cout << std::endl;
}


void test_Fraction_add()
{
    int xn = 0, xd = 0; // Fraction xn/xd
    int yn = 0, yd = 0; // Fraction yn/yd
    int zn = 0, zd = 0; // Fraction zn/zd

    std::cin >> yn >> yd >> zn >> zd;
    Fraction_add(xn, xd, yn, yd, zn, zd);
    Fraction_print(xn, xd);
    std::cout << std::endl;
}


int main()
{
    int option;
    std::cin >> option:
    switch (option)
    {
```

```
        case 1:
            test_Fraction_print();
            break;
        case 2:
            test_Fraction_add();
            break;
    }

    return 0;
}
```

a02q02/skel/Fraction.h

```
// File: Fraction.h

//-----------------------------------------------------------------------------
// Print fraction modeled by numerator n and denominator d.
//-----------------------------------------------------------------------------
void Fraction_print(int n, int d);

//-----------------------------------------------------------------------------
// The fraction modeled by xn and xd as numerator and denominator, i.e., xn/xd,
// is set to the sum of the fractions modeled by yn/yd and zn/zd, i.e.,
// xn/xd = yn/yd + zn/zd
//-----------------------------------------------------------------------------
void Fraction_add(int & xn, int & xd,
                  int yn, int yd,
                  int zn, int zd);
```

a02q02/skel/Fraction.cpp

```
// File: Fraction.cpp

void Fraction_print(int n, int d)
{
}


void Fraction_add(int & xn, int & xd,
                  int yn, int yd,
                  int zn, int zd)
{
}
```

Q3. This is a continuation of Q2. (In other words, after you finished Q2, copy the files from Q2 to the directory/folder for Q3.)

Now include a function to subtract fractions in your library. Test it throughly.

You must add a test function to the main C++ file and allow user to test your function when the user enters option 3. (Refer to Q2.) Previous code for testing with option 1 and 2 must be kept.

a02q03/skel/Fraction.h

```
// File: Fraction.h

//--------------------------------------------------------------------------
// Print fraction modeled by numerator n and denominator d.
//--------------------------------------------------------------------------
void Fraction_print(int n, int d);

//--------------------------------------------------------------------------
// The fraction modeled by xn and xd as numerator and denominator, i.e., xn/xd,
// is set to the sum of the fractions modeled by yn/yd and zn/zd, i.e.,
// xn/xd = yn/yd + zn/zd
//--------------------------------------------------------------------------
void Fraction_add(int & xn, int & xd,
                  int yn, int yd,
                  int zn, int zd);

//--------------------------------------------------------------------------
// The fraction modeled by xn and xd as numerator and denominator, i.e., xn/xd,
// is set to the difference of the fractions modeled by yn/yd and zn/zd, i.e.,
// xn/xd = yn/yd - zn/zd
//--------------------------------------------------------------------------
void Fraction_sub(int & xn, int & xd,
                  int yn, int yd,
                  int zn, int zd);
```

a02q03/skel/Fraction.cpp

```
// File: Fraction.cpp

void Fraction_print(int n, int d)
{
}


void Fraction_add(int & xn, int & xd,
                  int yn, int yd,
                  int zn, int zd)
```

```
{
}


void Fraction_sub(int & xn, int & xd,
                  int yn, int yd,
                  int zn, int zd)
{
}
```