CS 2A Object-Oriented Programming in C++

Assignment 9

Assignment 9.1 [75 points]

For this assignment you will be building on the fraction class you began last week. You'll be making four major changes to the class.

1. **[15 points]** Delete your set() function. Add two constructors, a default constructor that assigns the value 0 to the fraction, and a constructor that takes two parameters. The first parameter will represent the initial numerator of the fraction, and the second parameter will represent the initial denominator of the fraction.

   Since fractions cannot have denominators of 0, the default constructor should assign 0 to the numerator and 1 to the denominator.

2. **[15 points]** Add the const keyword to your class wherever appropriate. Your class may still work correctly even if you don't do this correctly, so this will require extra care!!

3. **[5 points]** Add a private "simplify()" function to your class and call it from the appropriate member

functions. (There will probably be 5 places where you need to call it.) The best way to do this is to make the function a void function with no parameters that reduces the calling object.

As you can see from the sample output given below, you are still not required to change improper fractions into mixed numbers for printing. Just print it as an improper fraction. Make sure that your class will reduce ANY fraction, not just the fractions that are tested in the provided client program. Fractions should not be simply reduced upon output, they should be stored in reduced form at all times. In other words, you should ensure that all fraction objects are reduced before the end of any member function. You are also not required to deal with negative numbers, either in the numerator or the denominator.

You must create your own algorithm for reducing fractions. Don't look up an already existing algorithm for reducing fractions or finding GCF. The point here is to have you practice solving the problem on your own. In particular, don't use Euclid's algorithm. Don't worry about being efficient. It's fine to have your function check every possible factor, even if it would be more efficient to just check prime numbers. Just create something of your own that works correctly on ANY fraction.

Note: this part of the assignment is worth 5 points. If you are having trouble keeping up with the class, I suggest you skip this part and take the 5 point deduction.

4. **[20 points]** Put the client program in a separate file from the class, and divide the class into specification file (fraction.h) and implementation file (fraction.cpp), so your code will be in 3 separate files.

5. Add documentation to your assignment. Be sure to carefully read section 1D of the Style Conventions, "Commenting in Classes".

Your class should still have exactly two data members.

I am providing a client program for you below. You should copy and paste this and use it as your client program. The output that should be produced when the provided client program is run with your class is also given below, so that you can check your results. Since you are not writing the client program, you are not required to include comments in it.

Here is the client program.

```
#include <iostream>
#include "fraction.h"
using namespace std;

int main()
{
    fraction f1(9,8);
    fraction f2(2,3);
    fraction result;

    cout << "The result starts off at ";
    result.print();
    cout << endl;

    cout << "The product of ";
    f1.print();
    cout << " and ";
    f2.print();
    cout << " is ";
    result = f1.multipliedBy(f2);
    result.print();
    cout << endl;

    cout << "The quotient of ";
    f1.print();
```

```
    cout << " and ";
    f2.print();
    cout << " is ";
    result = f1.dividedBy(f2);
    result.print();
    cout << endl;

    cout << "The sum of ";
    f1.print();
    cout << " and ";
    f2.print();
    cout << " is ";
    result = f1.addedTo(f2);
    result.print();
    cout << endl;

    cout << "The difference of ";
    f1.print();
    cout << " and ";
    f2.print();
    cout << " is ";
    result = f1.subtract(f2);
    result.print();
    cout << endl;

    if (f1.isEqualTo(f2)){
        cout << "The two fractions are equal." << endl;
    } else {
        cout << "The two fractions are not equal." << endl;
    }

    const fraction f3(12, 8);
    const fraction f4(202, 303);
    result = f3.multipliedBy(f4);
    cout << "The product of ";
    f3.print();
    cout << " and ";
    f4.print();
    cout << " is ";
    result.print();
    cout << endl;
}
```

This client should produce the output shown here:

```
The result starts off at 0/1
The product of 9/8 and 2/3 is 3/4
The quotient of 9/8 and 2/3 is 27/16
The sum of 9/8 and 2/3 is 43/24
The difference of 9/8 and 2/3 is 11/24
The two fractions are not equal.
The product of 3/2 and 2/3 is 1/1
```

You may not change the client program in any way. Changing the client program will result in a grade of 0 on the project.

Submit Your Work

Name your source code file(s) fraction.cpp and fraction.h. Execute the given client program and copy/paste the output into the bottom of the implementation file, making it into a comment. Use the Assignment Submission link to submit the source file(s). When you submit your assignment there will be a text field in which you can add a note to me (called a "comment", but don't confuse it with a C++ comment). In this "comments" section of

the submission page let me know whether the class works as required.