**CPS 271: Object Oriented Programming in C++**

**Fall 2019**

**Assignment #3**

Expand on Assignment #2 using Fractions and write functions for all the math operators available and properly overloading them. Write operator overloaded functions for the following operators:

**These binary functions return Fraction objects:**

**+= -= \*= /= + - / \* =**

**These unary functions return Fraction objects:**

**- ^ ++ -- (both prefix & postfix)**

**These extraction/insertion functions for printing and inputting Fraction objects:**

**<< >>**

**Theses return boolean True / False: These casting functions:**

**< > <= >= == != ! ( double ) ( float )**

Examples: Fraction a, b, c ;

a = b + c ; a = b - c ; a += b ; a -= b ;

a = b \* c ; a = b / c ; a \*= b ; a /= b ;

if ( a == b ) if ( a != b ) int x = !a ; // 0 or 1

if ( a < b ) if ( a > b ) float f = (float) a ;

if ( a <= b ) if ( a <= b ) double d = (double) a ;

In the ++ and --, these function increment or decrement by 1/1.

So if **a** is 6/5, then at the end of each line **a** and **b** are:

a = ++b ; // a is 11/5 b is 11/5

a = b++ ; // a is 6/5 b is 11/5

a = --b ; // a is 1/5 b is 1/5

a = b-- ; // a is 7/5 b is 1/5

When a = 2/5, for - b = -a // a is 2/5 and b is –2/5

When a = 2/5, for ^ b = ^a // a is 2/5 and b is 5/2

Printing and Inputting

cout << a << endl; cin >> b;

Create a private LowTerm/Reduce() function within the class which reduces the Fraction to lowest terms. For example, Fractions like these would become these…

2/4 -> 1/2

5/8 -> 5/8

76/32 -> 19/8

17/17 -> 1/1

24/12 -> 2/1

45/65 -> 9/13

Once again, make sure to use exception handling in the event there is ever a divide by zero or any other error condition which would cause the program to terminate unexpectedly.

Finally, expand the main program, so it allows the user to call any of the operator functions you’ve created. As before, the program ends when the user enters **Q** or **q** to quit for the math operation.