NAME: ________________________________

```
class Fraction
{
public:
    Fraction(int numerator, int denominator);
    int getNumerator() const;
    int getDenominator() const;
    void setNumerator(int numerator);
    void setDenominator(int denominator);
    bool isImproper() const;
    int integerLowerBound() const;
    void simplify();

private:
    int numerator;
    int denominator;
};
```

Figure 1: Declaration of a class named Fraction

Instances of the Fraction class represent fractions. For this exam, assume that the numerator and denominator within instances of the Fraction class are positive integers.

The function *getNumerator* returns the numerator value. The function *getDenominator* returns the denominator value. The function setNumerator lets code set the numerator value. Similarly, the function setDenominator lets code set the denominator value.

The function *isImproper* returns true if the numerator is larger than the denominator, otherwise it returns false. For example, isImproper returns true for 8/6.

The function *integerLowerBound* returns the greatest integer that is less than or equal to the fraction. For example, the integer lower bound of 17/5 is 3.

The *isLucky* function returns true if the numerator is divisible by 7 and the denominator is not divisible by 13. For example, 14/80 is lucky but 14/26 is not.

The function *simplify* reduces the numerator and denominator to their smallest possible values without changing the rational number the fraction represents. For example, 8/6 simplifies to 4/3.

1) Draw the UML class diagram for the Fraction class. (20 points)

2) Provide an implementation of the Fraction constructor. (20 points)

3) Provide an implementation of the getNumerator function. (20 points)

4) Provide an implementation of the getDenominator function. (20 points)

5) Provide an implementation of the setNumerator function. (20 points)

6) Provide an implementation of the setDenominator function. (20 points)

7) Provide an implementation of the isImproper function. (20 points)

8) Provide an implementation of the integerLowerBound function. Hint: rely on integer division. (20 points)

9) Provide test code for the isImproper function. Use assert statements for this purpose. Make sure you test both the case that the function returns true and false. (20 points)

10) Provide an implementation of the simplify function. (20 points)