
Project - 1 MA517M-Basic Programming Laboratory	Last Date: 14 November 2026
Name	Roll No.: MA26M016

Rational Numbers (Fractions) using C++ Classes and Operator Overloading

Objective: To design a C++ program that implements rational numbers using classes, supports arithmetic operations, and demonstrates operator overloading.

A rational number is of the form

$$r = \frac{p}{q}, \quad p, q \in \mathbb{Z}, q \neq 0$$

Two rational numbers $\frac{p_1}{q_1}$ and $\frac{p_2}{q_2}$ are equal if $p_1q_2 = p_2q_1$. Arithmetic operations include addition, subtraction, multiplication, division, and simplification to lowest terms.

Problem Description

Design a class `Rational` that represents fractions. The class should support arithmetic operations, comparison, and display using operator overloading.

Class Specification

- **Class Name:** `Rational`
- **Private Data Members:**
 - `int numerator;`
 - `int denominator;`
- **Public Member Functions:**
 - `Rational(int p=0, int q=1);`
 - `void simplify();`
 - `void display() const;`

Operator Overloading

- `operator +()`, `-()` Fraction addition and subtraction
- `operator *()`, `/()` Fraction multiplication and division
- `operator ==(())`, `!=(())` Checks equality or inequality of two fractions
- `operator <()`, `>()`, `<=()`, `>=()` Compares fractions by converting to common denominator or using cross-multiplication
- `operator <<()` Prints fraction in standard form

Tasks

1. Create rational numbers $r_1 = \frac{3}{4}$ and $r_2 = \frac{5}{6}$
2. Compute $r_1 + r_2$, $r_1 - r_2$, $r_1 * r_2$, r_1 / r_2
3. Simplify all results to lowest terms
4. Compare r_1 and r_2 using overloaded comparison operators
5. Display results using the overloaded << operator

Expected Output Example

```
r1 = 3/4
```

```
r2 = 5/6
```

```
r1 + r2 = 19/12
```

```
r1 - r2 = -1/12
```

```
r1 * r2 = 5/8
```

```
r1 / r2 = 9/10
```

```
r1 == r2 : False
```

```
r1 < r2 : True
```

Project - 2 : Word Shuffle Game Using C++ Classes

Problem Statement

Design and implement a **Word Shuffle Game** using **C++ classes**. The program should allow the user to unscramble letters to form meaningful words. The project should utilize object-oriented programming concepts such as classes, objects, encapsulation, and methods to handle word selection, shuffling, and user interaction.

Project Requirements

1. Create a **Word** class to represent a word in the game.
 - (a) Include a method to randomly select a word from a predefined list.
 - (b) Include a method to shuffle the letters of the word.
 - (c) Include a method to display the shuffled word to the user.
2. Create a **Game** class to manage gameplay.
 - (a) Display options for the user: **Play** or **Solution**.
 - (b) If the user chooses **Play**, allow them to enter guesses.
 - (c) Validate user input and provide feedback if the guess is correct or incorrect.
 - (d) Keep track of the number of attempts.

- (e) Allow the user to quit by entering a special command (e.g., **Q/q**) and confirm before exiting.
- 3. If the user chooses **Solution**, display the original word and the correct sequence of letters.
- 4. Ensure proper encapsulation of word logic and gameplay operations within the respective classes.

Suggested Class Structure

1. **Word Class:**

- Data member: `string originalWord`, `string shuffledWord`
- Methods: `selectWord()`, `shuffleWord()`, `displayWord()`

2. **Game Class:**

- Data member: `Word` object, user choice, attempt counter
- Methods: `playGame()`, `showSolution()`, `processGuess(string guess)`, `confirmQuit()`