

Project - 1 MA517M-Basic Programming Laboratory Last Date: 14 November 2026
Name _____ Roll No.: MA26M008

Prime Factorization using C++ Classes and Operator Overloading

Objective: To design a C++ program that stores an integer along with its unique prime factors using a class, and supports arithmetic operations, GCD, LCM, and comparisons using operator overloading.

Every positive integer greater than 1 can be uniquely represented as a product of prime numbers:

$$n = p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_k^{\alpha_k}$$

where p_i are prime numbers and α_i are positive integers. This is the Fundamental Theorem of Arithmetic.

Problem Description

Design a class `PrimeFactorization` to represent an integer n and its prime factors. The class should allow multiplication, division, computation of GCD and LCM with other integers, and comparison using operator overloading.

Class Specification

- **Class Name:** `PrimeFactorization`
- **Private Data Members:**
 - `int value;`
 - `vector<pair<int,int>> factors;`
 - `void factorize();`
- **Public Member Functions:**
 - `PrimeFactorization(int n);`
 - `int getValue() const;`
 - `void display() const;`
 - `vector<pair<int,int>> getFactors() const;`

Operator Overloading

- `operator *`, `/` Multiplies or divides two integers using prime factor representation
- `operator gcd()`, `lcm()` Computes GCD and LCM with another integer
- `operator ==(())`, `!=(())`, `<()`, `>()` Compares integers or their prime factorizations
- `operator <<()` Displays the integer and its prime factors in a readable form

Tasks

1. Create `PrimeFactorization` objects for $n = 60$ and $m = 48$
2. Display their prime factors
3. Compute $n \times m$ and n/m using prime factorization
4. Compute $\text{gcd}(n, m)$ and $\text{lcm}(n, m)$
5. Compare n and m using overloaded operators

Expected Output Example

```
n = 60, prime factors = 2^2 * 3^1 * 5^1
m = 48, prime factors = 2^4 * 3^1

n * m = 2880, prime factors = 2^6 * 3^2 * 5^1
n / m = 5/4 (or simplified representation)

gcd(n, m) = 12
lcm(n, m) = 240

n == m: False
n > m: 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()`