Lab 8

Name: Yuvraj Singh
Roll No: 23072021
M.Tech CSE

Problem: Take a prime p as input and write functions to implement field operations of Z_p

Solution:

The finite field Z_p, where p is a prime number, consists of integers modulo p. It is denoted by Z_p and contains p elements, known as residues, obtained by dividing integers by p and taking the remainder.
Mathematically, the set Z_p is defined as:

$$Z_p = \{0, 1, 2, \ldots, p-1\}$$

Addition, subtraction, multiplication, and division in Z_p are defined modulo p. For any two integers a and b in Z_p, the operations are computed as follows:

- **Addition**: The sum $a + b$ is calculated modulo p as $(a + b) \mod p$.
- **Subtraction**: The difference $a - b$ is calculated modulo p as $(a - b) \mod p$.
- **Multiplication**: The product $a \times b$ is calculated modulo p as $(a \times b) \mod p$.
- **Division**: The quotient $\frac{b}{a}$ is calculated as the modular multiplicative inverse of a modulo p, denoted as a^{-1}, multiplied by b. Mathematically, $\frac{b}{a}$ is represented as $(b \times a^{-1}) \mod p$.

Example: Let's consider the finite field Z_7, where $p = 7$. Perform addition, subtraction, multiplication, and division for the following integers:

$$a = 2, \qquad b = 3$$

Addition

$$a + b = (2 + 3) \mod 7 = 5 \mod 7 = 5$$

Subtraction

$$a - b = (2 - 3) \mod 7 = -1 \mod 7 = 6$$

Multiplication

$$a \times c = (2 \times 3) \mod 7 = 8 \mod 7 = 6$$

Division To find $\frac{a}{b}$, we need to calculate the modular multiplicative inverse of b modulo 7. Since $b = 3$, we find b^{-1} such that $3 \times b^{-1} \equiv 1 \mod 7$. In this case, $b^{-1} = 5$ because $6 \times 6 \equiv 1 \mod 7$.

$$\frac{a}{b} = (2 \times 5) \mod 7 = 10 \mod 7 = 3$$

Implementation in C++

```cpp
#include <iostream>
using namespace std;

class ModularArithmetic {
private:
    int p;

    // Function to perform modulo operation
    int mod(int a, int b) {
        int result = a % b;
        if (result < 0) result += b;

        return result;
    }

public:
    ModularArithmetic(int modulus) : p(modulus) {}

    // Function to calculate modular exponentiation (a^b mod p)
    int modExp(int a, int b) {
        if (b == 0) return 1;
        long long int temp = modExp(a, b / 2);
        long long int result = (temp * temp) % p;
        if (b % 2 == 1) result = (result * a) % p;

        return static_cast<int>(result);
    }

    // Function to calculate modular inverse (a^(-1) mod p)
    int modInverse(int a) {
        return modExp(a, p - 2);
    }

    // Function to perform addition in Zp
    int add(int a, int b) {
        return mod(a + b, p);
    }

    // Function to perform subtraction in Zp
    int subtract(int a, int b) {
        return mod(a - b, p);
    }

    // Function to perform multiplication in Zp
    int multiply(int a, int b) {
        return mod(a * b, p);
    }

    // Function to perform division in Zp
```

```
    int divide(int a, int b) {
        // Division by b is equivalent to multiplication by the modular inve
        return multiply(a, modInverse(b));
    }
};

int main() {
    int p;
    cout << "Enter the prime number p: ";
    cin >> p;

    ModularArithmetic Zp(p);

    int a, b;
    cout << "Enter two integers a and b: ";
    cin >> a >> b;

    // Perform field operations
    int sum = Zp.add(a, b);
    int difference = Zp.subtract(a, b);
    int product = Zp.multiply(a, b);
    int quotient = Zp.divide(a, b);

    // Output results
    cout<<"Sum(a + b) mod "<<p<<" = "<<sum<<endl;
    cout<<"Difference(a - b) mod "<<p<<" = "<<difference<<endl;
    cout<<"Product(a * b) mod "<<p<<" = "<<product<<endl;
    cout<<"Quotient(a / b) mod "<<p<<" = "<<quotient<<endl;

    return 0;
}
```

Time Complexity: $O(\log \max(a, b))$

Output:

```
Enter the prime number p: 7
Enter two integers a and b: 2 3
Sum(a + b) mod 7 = 5
Difference(a - b) mod 7 = 6
Product(a * b) mod 7 = 6
Quotient(a / b) mod 7 = 3
```