**ASSIGNMENT-8**

**Question-1**

**Problem Statement:** Create following methods to perform different arithmetic operations:

i) static int sum(int,int)

ii) static float sum(float,float)

iii) static double sum(double,double)

iv) static double sum(double,float)

v) static double sum(double,int)

vi) static int substract(int,int)

vii) static float substract (float,float)

viii) static double substract (double,double)

ix) static double substract (double,float)

x) static double substract (double,int)

xi) static double div(int,int)

xii) static float div (float,float)

xiii) static double div (double,double)

xiv) static double div (double,int)

xv) static int isPrime(int);

xvi) static int findFactorial(int)

xvii) static boolean isPallindrom(int)

Call above methods in main() method to display the result with appropriate input values.

**Source Code**

// Class ArithmeticOperations

public class ArithmeticOperations {

// Sum methods

public static int sum(int a, int b) {

return a + b;

}

public static float sum(float a, float b) {

return a + b;

}

public static double sum(double a, double b) {

return a + b;

}

public static double sum(double a, float b) {

return a + b;

}

public static double sum(double a, int b) {

return a + b;

}

// Subtraction methods

public static int subtract(int a, int b) {

return a - b;

}

public static float subtract(float a, float b) {

return a - b;

}

public static double subtract(double a, double b) {

return a - b;

}

public static double subtract(double a, float b) {

return a - b;

}

public static double subtract(double a, int b) {

return a - b;

}

// Division methods

public static double div(int a, int b) {

if (b == 0) {

throw new ArithmeticException("Division by zero!");

}

return (double) a / b;

}

public static float div(float a, float b) {

if (b == 0) {

throw new ArithmeticException("Division by zero!");

}

return a / b;

}

public static double div(double a, double b) {

if (b == 0) {

throw new ArithmeticException("Division by zero!");

}

return a / b;

}

public static double div(double a, int b) {

if (b == 0) {

throw new ArithmeticException("Division by zero!");

}

return a / b;

}

// Method to check if a number is prime

public static boolean isPrime(int num) {

if (num <= 1) {

return false;

}

for (int i = 2; i <= Math.sqrt(num); i++) {

if (num % i == 0) {

return false;

}

}

return true;

}

// Method to find factorial of a number

public static int findFactorial(int num) {

if (num == 0 || num == 1) {

return 1;

}

return num \* findFactorial(num - 1);

}

// Method to check if a number is palindrome

public static boolean isPalindrome(int num) {

int originalNum = num;

int reversedNum = 0;

while (num != 0) {

int digit = num % 10;

reversedNum = reversedNum \* 10 + digit;

num /= 10;

}

return originalNum == reversedNum;

}

public static void main(String[] args) {

// Testing arithmetic operations

System.out.println("Sum of 5 and 3: " + sum(5, 3));

System.out.println("Sum of 5.5 and 3.7: " + sum(5.5f, 3.7f));

System.out.println("Sum of 5.5 and 3.7: " + sum(5.5, 3.7));

System.out.println("Sum of 5.5 and 3: " + sum(5.5, 3));

System.out.println("Sum of 5.5 and 3: " + sum(5.5, 3));

System.out.println("Subtraction of 5 and 3: " + subtract(5, 3));

System.out.println("Subtraction of 5.5 and 3.7: " + subtract(5.5f, 3.7f));

System.out.println("Subtraction of 5.5 and 3.7: " + subtract(5.5, 3.7));

System.out.println("Subtraction of 5.5 and 3: " + subtract(5.5, 3));

System.out.println("Subtraction of 5.5 and 3: " + subtract(5.5, 3));

try {

System.out.println("Division of 6 and 2: " + div(6, 2));

System.out.println("Division of 6.6 and 2.2: " + div(6.6f, 2.2f));

System.out.println("Division of 6.6 and 2.2: " + div(6.6, 2.2));

System.out.println("Division of 6.6 and 2: " + div(6.6, 2));

} catch (ArithmeticException e) {

System.out.println(e.getMessage());

}

// Testing prime number check

System.out.println("Is 7 a prime number? " + isPrime(7));

System.out.println("Is 10 a prime number? " + isPrime(10));

// Testing factorial calculation

System.out.println("Factorial of 5: " + findFactorial(5));

// Testing palindrome check

System.out.println("Is 12321 a palindrome? " + isPalindrome(12321));

System.out.println("Is 12345 a palindrome? " + isPalindrome(12345));

}

}

**OUTPUT:**

Sum of 5 and 3: 8

Sum of 5.5 and 3.7: 9.2

Sum of 5.5 and 3.7: 9.2

Sum of 5.5 and 3: 8.5

Sum of 5.5 and 3: 8.5

Subtraction of 5 and 3: 2

Subtraction of 5.5 and 3.7: 1.8

Subtraction of 5.5 and 3.7: 1.7999999999999998

Subtraction of 5.5 and 3: 2.5

Subtraction of 5.5 and 3: 2.5

Division of 6 and 2: 3.0

Division of 6.6 and 2.2: 3.0

Division of 6.6 and 2.2: 2.9999999999999996

Division of 6.6 and 2: 3.3

Is 7 a prime number? true

Is 10 a prime number? false

Factorial of 5: 120

Is 12321 a palindrome? true

Is 12345 a palindrome? false

**Question-2**

**Problem Statement:** WAP in Java to design a class Fraction having data members num and denom. Define default constructor, one argument constructor, two argument constructor and a copy constructor to set the values to numerator and denominator of different objects.

Define methods such as show(), add(), subtract(), multiply(), div(), compare(), mixed() and reduce().

Create different objects using different constructors given above. Perform the above operations using appropriate methods. Show the result in reduced form of the result fraction after each operation. After reducing if the fraction is an improper fraction then represent it in the form of mixed fraction.

**Source Code**

public class Fraction {

private int num;

private int denom;

// Default constructor

public Fraction() {

num = 0;

denom = 1;

}

// One argument constructor

public Fraction(int num) {

this.num = num;

denom = 1;

}

// Two argument constructor

public Fraction(int num, int denom) {

if (denom == 0)

throw new IllegalArgumentException("Denominator cannot be zero");

this.num = num;

this.denom = denom;

reduce();

}

// Copy constructor

public Fraction(Fraction other) {

this.num = other.num;

this.denom = other.denom;

}

// Method to display fraction

public void show() {

System.out.println(num + "/" + denom);

}

// Method to add fractions

public Fraction add(Fraction other) {

int newNum = this.num \* other.denom + other.num \* this.denom;

int newDenom = this.denom \* other.denom;

Fraction result = new Fraction(newNum, newDenom);

result.reduce();

return result;

}

// Method to subtract fractions

public Fraction subtract(Fraction other) {

int newNum = this.num \* other.denom - other.num \* this.denom;

int newDenom = this.denom \* other.denom;

Fraction result = new Fraction(newNum, newDenom);

result.reduce();

return result;

}

// Method to multiply fractions

public Fraction multiply(Fraction other) {

int newNum = this.num \* other.num;

int newDenom = this.denom \* other.denom;

Fraction result = new Fraction(newNum, newDenom);

result.reduce();

return result;

}

// Method to divide fractions

public Fraction divide(Fraction other) {

if (other.num == 0)

throw new IllegalArgumentException("Cannot divide by zero");

int newNum = this.num \* other.denom;

int newDenom = this.denom \* other.num;

Fraction result = new Fraction(newNum, newDenom);

result.reduce();

return result;

}

// Method to compare fractions

public int compare(Fraction other) {

double val1 = (double) this.num / this.denom;

double val2 = (double) other.num / other.denom;

return Double.compare(val1, val2);

}

// Method to convert to mixed fraction

public void mixed() {

int whole = num / denom;

int remainder = num % denom;

System.out.println(whole + " " + remainder + "/" + denom);

}

// Method to reduce fraction to lowest terms

private void reduce() {

int gcd = gcd(Math.abs(num), denom);

num /= gcd;

denom /= gcd;

}

// Method to find greatest common divisor

private int gcd(int a, int b) {

if (b == 0)

return a;

return gcd(b, a % b);

}

public static void main(String[] args) {

Fraction f1 = new Fraction(3, 4);

Fraction f2 = new Fraction(2, 5);

// Addition

Fraction sum = f1.add(f2);

System.out.print("Sum: ");

sum.show();

// Subtraction

Fraction difference = f1.subtract(f2);

System.out.print("Difference: ");

difference.show();

// Multiplication

Fraction product = f1.multiply(f2);

System.out.print("Product: ");

product.show();

// Division

Fraction quotient = f1.divide(f2);

System.out.print("Quotient: ");

quotient.show();

// Comparison

System.out.println("Comparison result: " + f1.compare(f2));

// Mixed fraction

System.out.print("Mixed fraction of f1: ");

f1.mixed();

}

}

**OUTPUT:**

Sum: 23/20

Difference: 7/20

Product: 3/10

Quotient: 15/8

Comparison result: 1

Mixed fraction: 3/4

**Question-3**

**Problem Statement:** Create a class Complex having member variables real and img. Also create constructors and methods as follows:

a. Complex()

b. Complex(int,int)

c. Complex(Complex)

d. void showComplex()

e. Complex addComplex(Complex)

f. Complex substractComplex(Complex)

g. Complex multiplyComplex(Complex)

Write a java program to create objects of above class and perform operations as the methods specified above.

**Source Code**

public class Complex {

private int real;

private int img;

// Default constructor

public Complex() {

this.real = 0;

this.img = 0;

}

// Parameterized constructor to initialize real and img

public Complex(int real, int img) {

this.real = real;

this.img = img;

}

// Copy constructor

public Complex(Complex complex) {

this.real = complex.real;

this.img = complex.img;

}

// Method to display the complex number

public void showComplex() {

System.out.println(this.real + " + " + this.img + "i");

}

// Method to add two complex numbers

public Complex addComplex(Complex c) {

Complex result = new Complex();

result.real = this.real + c.real;

result.img = this.img + c.img;

return result;

}

// Method to subtract one complex number from another

public Complex subtractComplex(Complex c) {

Complex result = new Complex();

result.real = this.real - c.real;

result.img = this.img - c.img;

return result;

}

// Method to multiply two complex numbers

public Complex multiplyComplex(Complex c) {

Complex result = new Complex();

result.real = this.real \* c.real - this.img \* c.img;

result.img = this.real \* c.img + this.img \* c.real;

return result;

}

public static void main(String[] args) {

// Creating complex numbers

Complex c1 = new Complex(2, 3);

Complex c2 = new Complex(1, -1);

// Displaying complex numbers

System.out.println("Complex number 1:");

c1.showComplex();

System.out.println("Complex number 2:");

c2.showComplex();

// Performing addition

Complex sum = c1.addComplex(c2);

System.out.println("Sum of complex numbers:");

sum.showComplex();

// Performing subtraction

Complex difference = c1.subtractComplex(c2);

System.out.println("Difference of complex numbers:");

difference.showComplex();

// Performing multiplication

Complex product = c1.multiplyComplex(c2);

System.out.println("Product of complex numbers:");

product.showComplex();

}

}

**OUTPUT:**

Complex number 1:

2 + 3i

Complex number 2:

1 + -1i

Sum of complex numbers:

3 + 2i

Difference of complex numbers:

1 + 4i

Product of complex numbers:

5 + 1i