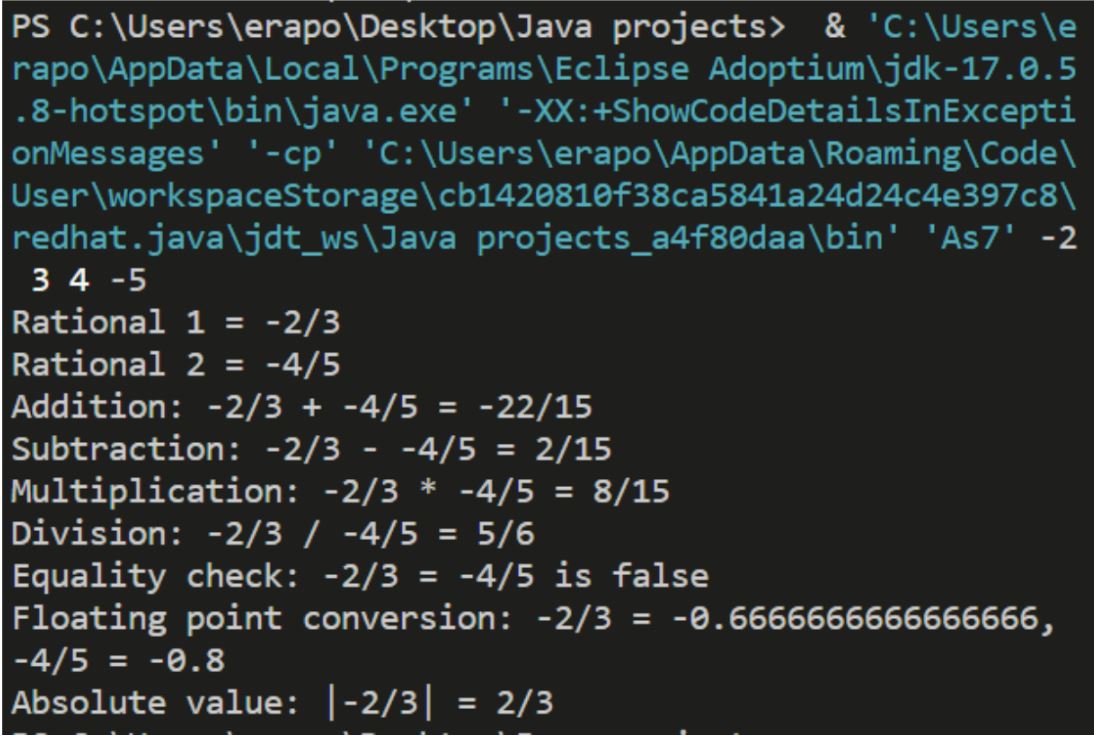
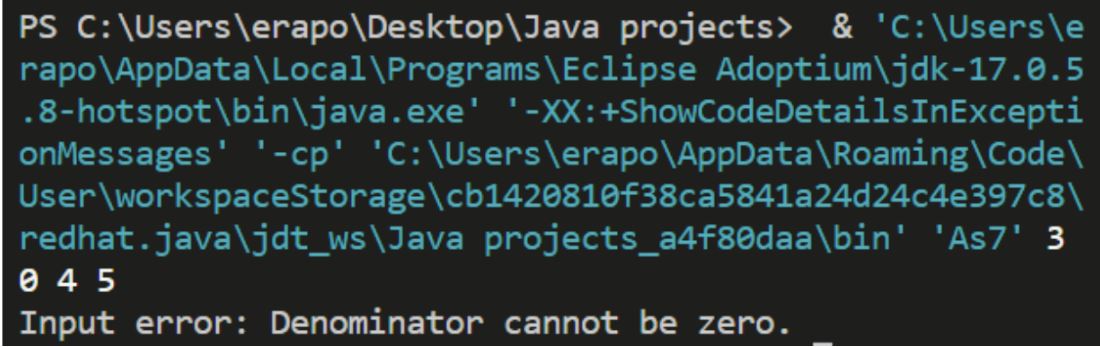
Assignment 7

import java.util.InputMismatchException;  
import java.util.Scanner;  
  
class RationalNumber {  
 private int numerator;  
 private int denominator;  
  
 public RationalNumber(int numerator, int denominator) {  
 if (denominator == 0) {  
 throw new IllegalArgumentException("Denominator cannot be zero.");  
 }  
 this.numerator = numerator;  
 this.denominator = denominator;  
 simplify();  
 }  
  
 public RationalNumber add(RationalNumber other) {  
 int resultNumerator = this.numerator \* other.denominator + other.numerator \* this.denominator;  
 int resultDenominator = this.denominator \* other.denominator;  
 return new RationalNumber(resultNumerator, resultDenominator);  
 }  
  
 public RationalNumber subtract(RationalNumber other) {  
 int resultNumerator = this.numerator \* other.denominator - other.numerator \* this.denominator;  
 int resultDenominator = this.denominator \* other.denominator;  
 return new RationalNumber(resultNumerator, resultDenominator);  
 }  
  
 public RationalNumber multiply(RationalNumber other) {  
 int resultNumerator = this.numerator \* other.numerator;  
 int resultDenominator = this.denominator \* other.denominator;  
 return new RationalNumber(resultNumerator, resultDenominator);  
 }  
  
 public RationalNumber divide(RationalNumber other) {  
 if (other.numerator == 0) {  
 throw new ArithmeticException("Cannot divide by zero.");  
 }  
 int resultNumerator = this.numerator \* other.denominator;  
 int resultDenominator = this.denominator \* other.numerator;  
 return new RationalNumber(resultNumerator, resultDenominator);  
 }  
  
 public boolean equals(RationalNumber other) {  
 return this.numerator == other.numerator && this.denominator == other.denominator;  
 }  
  
 public double toDouble() {  
 return (double) this.numerator / this.denominator;  
 }  
  
 public RationalNumber abs() {  
 int absNumerator = Math.*abs*(this.numerator);  
 int absDenominator = Math.*abs*(this.denominator);  
 return new RationalNumber(absNumerator, absDenominator);  
 }  
  
 private void simplify() {  
 int gcd = gcd(this.numerator, this.denominator);  
 this.numerator /= gcd;  
 this.denominator /= gcd;  
 if (this.denominator < 0) {  
 this.numerator = -this.numerator;  
 this.denominator = -this.denominator;  
 }  
 }  
  
 private int gcd(int a, int b) {  
 if (b == 0) {  
 return a;  
 }  
 return gcd(b, a % b);  
 }  
  
 @Override  
 public String toString() {  
 return this.numerator + "/" + this.denominator;  
 }  
}  
  
public class Main {  
 public static void main(String[] args) {  
 try {  
 int numerator1 = Integer.*parseInt*(args[0]);  
 int denominator1 = Integer.*parseInt*(args[1]);  
 RationalNumber rational1 = new RationalNumber(numerator1, denominator1);  
  
 int numerator2 = Integer.*parseInt*(args[2]);  
 int denominator2 = Integer.*parseInt*(args[3]);  
 RationalNumber rational2 = new RationalNumber(numerator2, denominator2);  
  
 System.*out*.println("Rational 1 = " + rational1);  
 System.*out*.println("Rational 2 = " + rational2);  
  
  
 // For executing a single function out of many, use the following code:  
  
 // if(args[4].equalsIgnoreCase("add")) {  
 // RationalNumber result = rational1.add(rational2);  
 // System.out.println("Addition: " + rational1 + " + " + rational2 + " = " + result);  
 // } else if(args[4].equalsIgnoreCase("subtract")){  
 // RationalNumber result = rational1.subtract(rational2);  
 // System.out.println("Subtraction: " + rational1 + " - " + rational2 + " = " + result);  
 // } else if(args[4].equalsIgnoreCase("multiply")){  
 // RationalNumber result = rational1.multiply(rational2);  
 // System.out.println("Multiplication: " + rational1 + " \* " + rational2 + " = " + result);  
 // } else if(args[4].equalsIgnoreCase("divide")) {  
 // try {  
 // RationalNumber result = rational1.divide(rational2);  
 // System.out.println("Division: " + rational1 + " / " + rational2 + " = " + result);  
 // } catch (ArithmeticException e) {  
 // System.out.println("Division error: " + e.getMessage());  
 // }  
 // } else if(args[4].equalsIgnoreCase("equals")){  
 // boolean isEqual = rational1.equals(rational2);  
 // System.out.println("Equality check: " + rational1 + " = " + rational2 + " is " + isEqual);  
 // } else if(args[4].equalsIgnoreCase("toDouble")) {  
 // double doubleValue1 = rational1.toDouble();  
 // double doubleValue2 = rational2.toDouble();  
 // System.out.println("Floating point conversion: " + rational1 + " = " + doubleValue1 + ", " + rational2 + " = " + doubleValue2);  
 // } else if(args[4].equalsIgnoreCase("abs")){  
 // RationalNumber result = rational1.abs();  
 // System.out.println("Absolute value: |" + rational1 + "| = " + result);  
 // } else {  
 // System.out.println("Invalid operation");  
 // }  
 // } catch (IllegalArgumentException e) {  
 // System.out.println("Invalid input: " + e.getMessage());  
 // }  
  
 //For executing all the functions, use the following code:  
  
 RationalNumber result = rational1.add(rational2);  
 System.*out*.println("Addition: " + rational1 + " + " + rational2 + " = " + result);  
 result = rational1.subtract(rational2);  
 System.*out*.println("Subtraction: " + rational1 + " - " + rational2 + " = " + result);  
 result = rational1.multiply(rational2);  
 System.*out*.println("Multiplication: " + rational1 + " \* " + rational2 + " = " + result);  
  
 try {  
 result = rational1.divide(rational2);  
 System.*out*.println("Division: " + rational1 + " / " + rational2 + " = " + result);  
 } catch (ArithmeticException e) {  
 System.*out*.println("Division error: " + e.getMessage());  
 }  
  
 boolean isEqual = rational1.equals(rational2);  
 System.*out*.println("Equality check: " + rational1 + " = " + rational2 + " is " + isEqual);  
  
 double doubleValue1 = rational1.toDouble();  
 double doubleValue2 = rational2.toDouble();  
 System.*out*.println("Floating point conversion: " + rational1 + " = " + doubleValue1 + ", " + rational2 + " = " + doubleValue2);  
  
 result = rational1.abs();  
 System.*out*.println("Absolute value: |" + rational1 + "| = " + result);  
 } catch (NumberFormatException e) {  
 System.*out*.println("Input error: " + e.getMessage() + ". Please enter integers as input.");  
 } catch (IllegalArgumentException e) {  
 System.*out*.println("Input error: " + e.getMessage());  
 } catch (ArrayIndexOutOfBoundsException e) {  
 System.*out*.println("Usage: java As7 <numerator1> <denominator1> <numerator2> <denominator2>");  
 }  
 }  
}

Output





Github Link-

https://github.com/NachiketRopia2003/Java-Assignment/tree/main/Assignment%201