// Calculator.java package Assign7;

import java.util.Arrays;

public class Calculator {

// Method to add two numbers

public static double add(double n1, double n2){ return n1 + n2;

}

// Method to subtract two numbers

public static double subtract(double n1, double n2){ return n1 - n2;

}

// Method to multiply two numbers

public static double multiply(double n1, double n2){ return n1 \* n2;

}

// Method to divide two numbers

public static double divide(double n1, double n2) throws CustomException {

// Check if the divisor is zero if(n2 == 0){

throw new CustomException("Division by zero error");

} else {

return n1 / n2;

}

}

// Method to calculate the mean (average) of an array of numbers public static double mean(double[] arr){

return Arrays.stream(arr).sum() / arr.length;

}

// Method to calculate the square root of a number public static double sqrt(double n){

return Math.pow(n, 0.5);

}

// Method to calculate the standard deviation of an array of numbers public static double stddev(double[] arr){

double standardDeviation = 0.0;

// Calculate the sum of squared differences from the mean for (double num : arr) {

standardDeviation += Math.pow(num - mean(arr), 2);

}

// Calculate the square root of the average of squared differences return Math.sqrt(standardDeviation / arr.length);

}

// Method to calculate the variance of an array of numbers public static double variance(double[] arr){

// Variance is the square root of the standard deviation return sqrt(stddev(arr));

}

// Method to calculate the power of a number raised to another number public static double power(double n1, double n2){

return Math.pow(n1, n2);

}

}

// CustomException.java package Assign7;

// CustomException class extends Exception to handle custom exceptions public class CustomException extends Exception {

// Default constructor for CustomException class public CustomException(){

super("Division by 0 error"); // Set default message for division by zero error

}

// Parameterized constructor for CustomException class to handle custom error messages

public CustomException(String msg){ super(msg); // Set custom error message

}

// Nested class NonNumericInputException extends Exception to handle non- numeric input errors

public static class NonNumericInputException extends Exception {

// Default constructor for NonNumericInputException class public NonNumericInputException() {

super("Non-numeric input error"); // Set default message for non- numeric input error

}

// Parameterized constructor for NonNumericInputException class to handle custom error messages

public NonNumericInputException(String msg) {

super(msg); // Set custom error message

}

}

}

// UserInput.java package Assign7;

import java.util.Scanner; public class UserInput {

private Scanner scanner;

// Constructor initializes the Scanner object for user input public UserInput() {

scanner = new Scanner(System.in);

}

// Method to get a double input from the user with a prompt message

// Throws NonNumericInputException if input is not a valid double public double getDoubleInput(String message) throws

CustomException.NonNumericInputException { System.out.println(message); String input = scanner.next();

if (!isNumeric(input)) {

throw new CustomException.NonNumericInputException("Non-numeric input error");

}

return Double.parseDouble(input);

}

// Method to get a string input from the user with a prompt message public String getStringInput(String message) {

System.out.println(message); return scanner.next();

}

// Method to get an array input from the user with a prompt message public double[] getArrayInput() {

System.out.println("Enter the size of the array:"); int size = scanner.nextInt();

double[] array = new double[size];

System.out.println("Enter elements:"); for (int i = 0; i < size; i++) {

if (scanner.hasNextDouble()) { array[i] = scanner.nextDouble();

}

}

return array;

}

// Method to close the Scanner when it is no longer needed public void closeScanner() {

scanner.close();

}

// Method to check if a string can be parsed as a double private boolean isNumeric(String str) {

try {

Double.parseDouble(str); return true;

} catch (NumberFormatException e) { return false;

}

}

}

//Main.java package Assign7;

public class Main {

public static void main(String[] args) {

// Create an instance of UserInput to handle user input UserInput userInput = new UserInput();

try {

// Prompt the user to choose an operation

String choice = userInput.getStringInput("Choose Operation: +, -,

/, \*, ^, var, stddev, avg");

// Check if the chosen operation is a basic arithmetic operation or a statistical operation

if (choice.equals("+") || choice.equals("-") || choice.equals("\*")

|| choice.equals("/") || choice.equals("^")) {

two numbers

complete

// For basic arithmetic operations, prompt the user to enter

double n1 = userInput.getDoubleInput("Enter First Number:"); double n2 = userInput.getDoubleInput("Enter Second Number:"); userInput.closeScanner(); // Close the scanner since input is

// Perform the chosen operation and display the result if (choice.equals("+")) {

System.out.println(Calculator.add(n1, n2));

} else if (choice.equals("-")) { System.out.println(Calculator.subtract(n1, n2));

} else if (choice.equals("\*")) { System.out.println(Calculator.multiply(n1, n2));

} else if (choice.equals("/")) {

// Handle division by zero exception try {

System.out.println(Calculator.divide(n1, n2));

} catch (CustomException e) { System.out.println(e.getMessage());

}

} else if (choice.equals("^")) { System.out.println(Calculator.power(n1, n2));

} else {

System.out.println("Invalid Operation");

}

} else {

// For statistical operations, prompt the user to enter an

array of numbers

complete

double[] arr = userInput.getArrayInput(); userInput.closeScanner(); // Close the scanner since input is

result

// Perform the chosen statistical operation and display the

if (choice.equals("var")) { System.out.println(Calculator.variance(arr));

} else if (choice.equals("stddev")) { System.out.println(Calculator.stddev(arr));

} else if (choice.equals("avg")) { System.out.println(Calculator.mean(arr));

} else {

System.out.println("Invalid Operation");

}

}

} catch (CustomException.NonNumericInputException e) {

// Handle non-numeric input exception System.out.println(e.getMessage());

}

}

}

Output:

