

**LAB-11**

By-Arun Lal

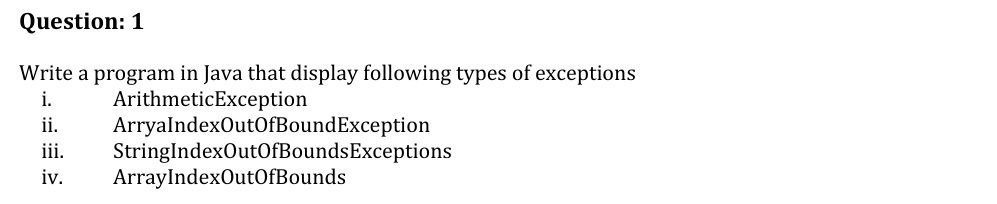


Sec-E

BSCS(II)

CMS-ID=023-24-0120

Exercise of Lab: 10



Code

public class Task1{

    public static void main(String[] *args*) {

    int a = 10, b = 0;

    String str = "Arun";

    int[] arr = new int[2];

    try { int c = a / b; } catch (ArithmeticException *e*) {

System.out.println(e); }

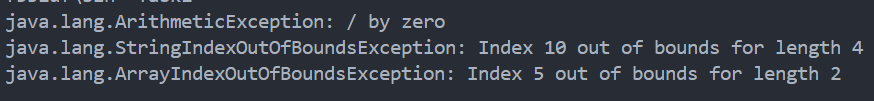
    try { System.out.println(str.charAt(10)); } catch (StringIndexOutOfBoundsException *e*) { System.out.println(e); }

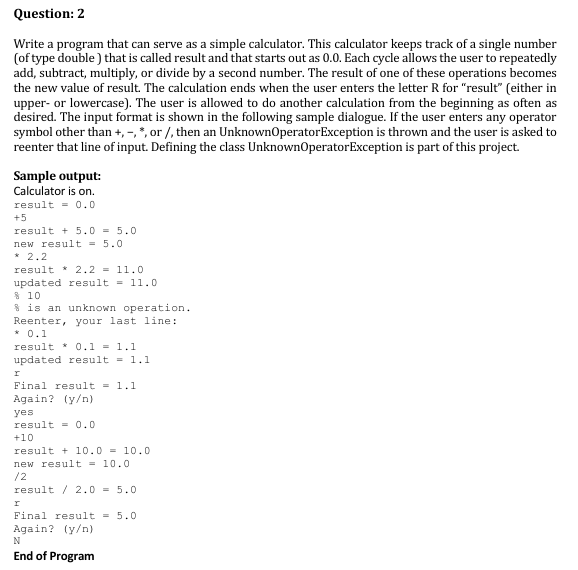
    try { arr[5] = 10; } catch (ArrayIndexOutOfBoundsException *e*) { System.out.println(e); }

    }

    }

OutPut





Code

import java.util.Scanner;

class UnknownOperatorException extends Exception {

public UnknownOperatorException(char *op*) {

super(op + " is an unknown operation.");

}

}

public class Task2 {

public static void main(String[] *args*) {

Scanner scan = new Scanner(System.in);

System.out.println("Calculator is on.");

boolean again = true;

while (again) {

double result = 0.0;

System.out.println("result = " + result);

boolean done = false;

while (!done) {

String input = scan.nextLine();

char op = input.charAt(0);

if (op == 'r' || op == 'R') {

System.out.println("Final result = " + result);

done = true;

continue;

}

double value = Double.parseDouble(input.substring(1));

try {

switch (op) {

case '+': System.out.println("result + " + value + " = " + (result + value)); result += value; break;

case '-': System.out.println("result - " + value + " = " + (result - value)); result -= value; break;

case '\*': System.out.println("result \* " + value + " = " + (result \* value)); result \*= value; break;

case '/': System.out.println("result / " + value + " = " + (result / value)); result /= value; break;

default: throw new UnknownOperatorException(op);

}

System.out.println("updated result = " + result);

} catch (UnknownOperatorException *e*) {

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

System.out.println("Reenter, your last line:");

}

}

System.out.println("Again? (y/n)");

String ans = scan.nextLine();

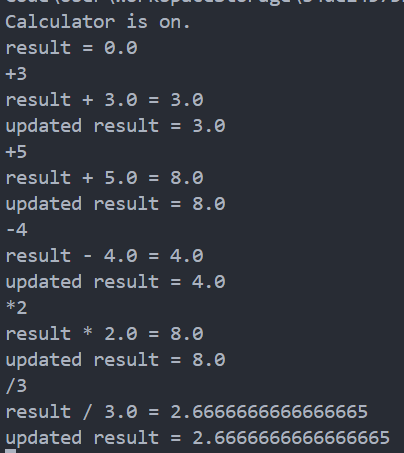
if (!ans.equalsIgnoreCase("y")) again = false;

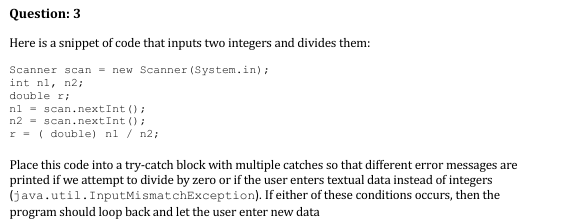
System.out.println("result = 0.0");

}

System.out.println("End of Program");

OutPut





Code

import java.util.\*;

public class Task3 {

public static void main(String[] *args*) {

Scanner scan = new Scanner(System.in);

int n1 = 0, n2 = 0;

double r = 0;

boolean valid = false;

while (!valid) {

try {

n1 = scan.nextInt();

n2 = scan.nextInt();

r = (double) n1 / n2;

System.out.println("Result = " + r);

valid = true;

} catch (ArithmeticException *e*) {

System.out.println("Cannot divide by zero.");

scan.nextLine();

} catch (InputMismatchException *e*) {

System.out.println("Invalid input. Please enter integers.");

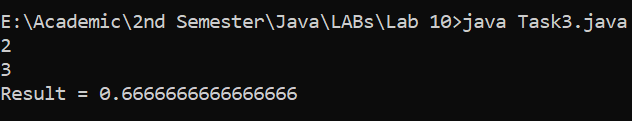
scan.nextLine();

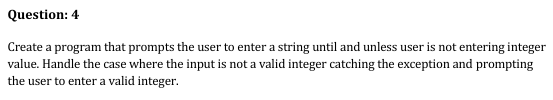
}

}

}

OutPut





Code

import java.util.Scanner;

public class Task4 {

public static void main(String[] *args*) {

Scanner scan = new Scanner(System.in);

boolean valid = false;

while (!valid) {

System.out.println("Enter an integer:");

try {

int number = Integer.parseInt(scan.nextLine());

System.out.println("Valid integer entered: " + number);

valid = true;

} catch (NumberFormatException *e*) {

System.out.println("Invalid input. Please enter a valid integer.");

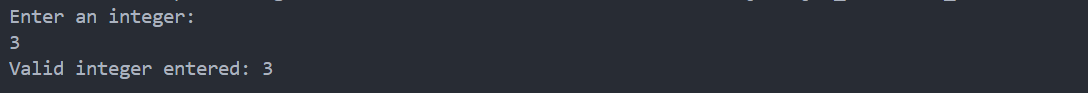
}

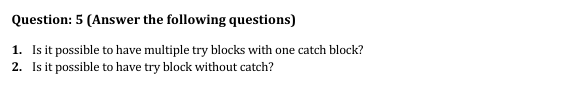
}

}

}

OutPut





1. **Yes**, it is possible to have multiple try blocks with **one catch block** by using a **multi-catch** block or surrounding multiple try blocks inside one catch scope using appropriate nesting or sequential execution.
2. **Yes**, a try block can be used **without a catch** if it is followed by a finally block, e.g.,

java

CopyEdit

try {

//

} finally {

// }