**Name:Allahdad**

**Section :D**

**CMS(023-22-0056)**

**Lab 09**

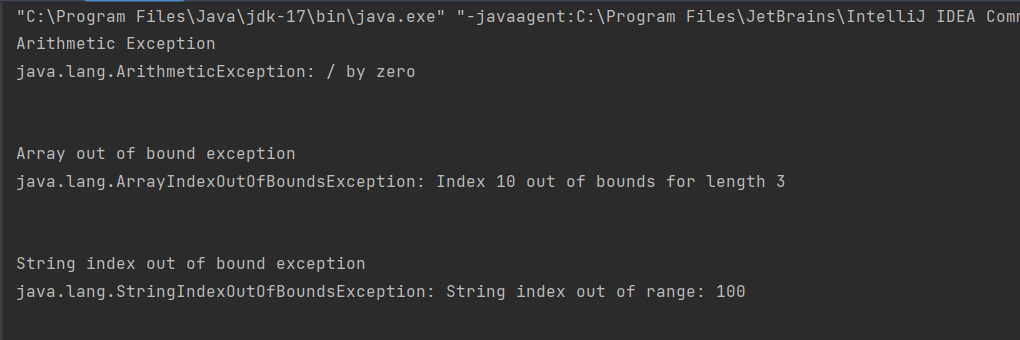
Task1: Write a program in Java that display following types of exceptions

1. ArithmeticException
2. ArryaIndexOutOfBoundException
3. StringIndexOutOfBoundsExceptions

INPUT

public class Ad1 {  
 public static void main(String args[])  
 {  
 try  
 {  
 //Code of Arithmetic Exception  
 int c=100/0;  
  
 }  
 catch (Exception e)  
 {  
 System.*out*.println("Arithmetic Exception\n"+e);  
 System.*out*.println("\n");  
 }  
  
  
 try  
 {  
 //Exception ArrayIndex out of bound  
 int []arr={1,2,3};  
 System.*out*.println(arr[10]);  
 }  
 catch (Exception e)  
 {  
 System.*out*.println("Array out of bound exception\n"+e);  
 System.*out*.println("\n");  
 }  
  
  
 try  
 {  
 //String out of bound exception  
 String name="Allahdad";  
 System.*out*.println(name.charAt(100));  
 }  
 catch (Exception e)  
 {  
 System.*out*.println("String index out of bound exception\n"+e);  
 System.*out*.println("\n");  
 }  
  
 }  
}

OUTPUT



**Task2:** Write a program that can serve as a simple calculator. This calculator keeps track of a single number (of type double ) that is called result and that starts out as 0.0 . Each cycle allows the user to repeatedly add, subtract, multiply, or divide by a second number. The result of one of these operations becomes the new value of result . The calculation ends when the user enters the letter R for “result” (either in upper- or lowercase). The user is allowed to do another calculation from the beginning as often as desired. The input format is shown in the following sample dialogue. If the user enters any operator symbol other than + , −, \* , or / , then an UnknownOperatorException is thrown and the user is asked to reenter that line of input. Defining the class UnknownOperatorException is part of this project.

**Sample output:**

Calculator is on.

result = 0.0

+5

result + 5.0 = 5.0

new result = 5.0

\* 2.2

result \* 2.2 = 11.0

updated result = 11.0

% 10

% is an unknown operation.

Reenter, your last line:

\* 0.1

result \* 0.1 = 1.1

updated result = 1.1

r

Final result = 1.1

Again? (y/n)

yes

result = 0.0

+10

result + 10.0 = 10.0

new result = 10.0

/2

result / 2.0 = 5.0

r

Final result = 5.0

Again? (y/n)

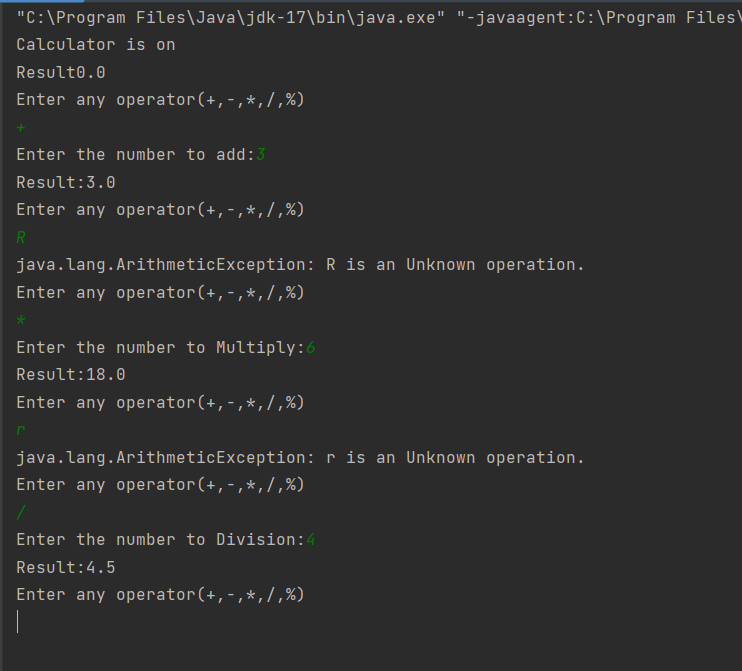
N

**End of Program**

**INPUT**

import java.util.Scanner;  
class Task2  
{  
 public static void main(String args[])  
 {  
 Scanner ip=new Scanner(System.*in*);  
 System.*out*.println("Calculator is on");  
 double result=0.0;  
 System.*out*.println("Result"+result);  
 double num;  
 char oper;  
  
 do{  
 System.*out*.println("Enter any operator(+,-,\*,/,%)");  
 oper=ip.next().charAt(0);  
  
 switch(oper)  
 {  
 case '+':  
 {  
 System.*out*.print("Enter the number to add:");  
 num=ip.nextDouble();  
 result+=num;  
 System.*out*.println("Result:"+result);  
 break;  
 }  
 case '-':  
 {  
 System.*out*.print("Enter the number to Subtract:");  
 num=ip.nextDouble();  
 result-=num;  
 System.*out*.println("Result:"+result);  
 break;  
 }  
 case '\*':  
 {  
 System.*out*.print("Enter the number to Multiply:");  
 num=ip.nextDouble();  
 result\*=num;  
 System.*out*.println("Result:"+result);  
 break;  
 }  
 case '/':  
 {  
 System.*out*.print("Enter the number to Division:");  
 num=ip.nextDouble();  
 result/=num;  
 System.*out*.println("Result:"+result);  
 break;  
 }  
  
 default:  
 {  
 try {  
 throw new ArithmeticException(oper + " is an Unknown operation.");  
 }  
 catch (Exception e)  
 {  
 System.*out*.println(e);  
 }  
  
 }  
 }  
 }  
 while(oper!='n'||oper!='N');  
 }  
}

**OUTPUT**

****

**Task3:** Here is a snippet of code that inputs two integers and divides them:

Scanner scan = new Scanner(System.in);

int n1, n2;

double r;

n1 = scan.nextInt();

n2 = scan.nextInt();

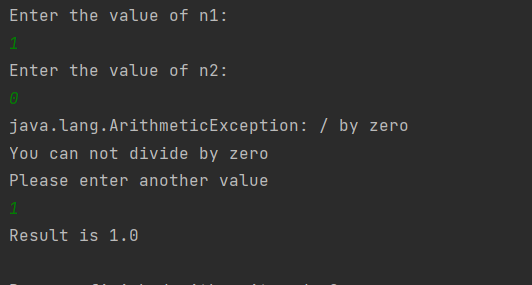
r = ( double) n1 / n2;

Place this code into a try-catch block with multiple catches so that different error messages are printed if we attempt to divide by zero or if the user enters textual data instead of integers (java.util.InputMismatchException). If either of these conditions occurs, then the program should loop back and let the user enter new data.

INPUT

import java.util.Scanner;  
public class Task3 {  
 public static void main (String args[])  
 {  
 Scanner scan=new Scanner(System.*in*);  
 int n1, n2;  
 System.*out*.println("Enter the value of n1:");  
 n1=scan.nextInt();  
 System.*out*.println("Enter the value of n2:");  
 n2=scan.nextInt();  
 double r;  
 try {  
 r = n1 / n2;  
 }  
 catch (Exception e)  
 {  
 System.*out*.println(e);  
 System.*out*.println("You can not divide by zero");  
 System.*out*.println("Please enter another value");  
 n2=scan.nextInt();  
  
 }  
 finally  
 {  
 r = n1 / n2;  
 System.*out*.println("Result is "+r);  
  
 }  
 }  
}

**output**

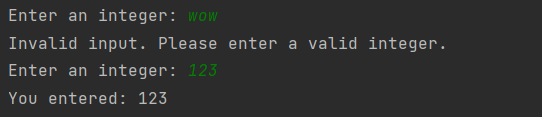
****

**Task4:** Create a program that prompts the user to enter a string until and unless user is not entering integer value. Handle the case where the input is not a valid integer catching the exception and prompting the user to enter a valid integer.

Input

import java.util.Scanner;  
  
public class Task4 {  
  
 public static void main(String[] args) {  
  
 Scanner scanner = new Scanner(System.*in*);  
 int num = 0;  
 boolean valid = false;  
  
 while (!valid) {  
 System.*out*.print("Enter an integer: ");  
 String input = scanner.nextLine();  
  
 try {  
 num = Integer.*parseInt*(input);  
 valid = true;  
 } catch (NumberFormatException e) {  
 System.*out*.println("Invalid input. Please enter a valid integer.");  
 }  
 }  
  
 System.*out*.println("You entered: " + num);  
 }  
}

output



THE END