**Name: Sandra P M**

**Roll No:34**

**Batch: MCA B**

**Date:31-05-2022**

**OBJECT ORIENTED PROGRAMMING LAB**

**Experiment No.: co4**

**Aim**

**2. Write a user defined exception class to authenticate the user name and password.**

**Procedure**

public class CustomExceptionExample {

public static class InvalidUserException extends Exception {

public InvalidUserException() {

super("Invalid username / password provided!");

}

}

public static void main(String[] args) {

String username = "san";

String password = "pass";

try {

if (username.equals("user") && password.equals("pass")) {

System.out.println("Authenticated successfully!");

} else {

throw new InvalidUserException();

}

} catch (InvalidUserException e) {

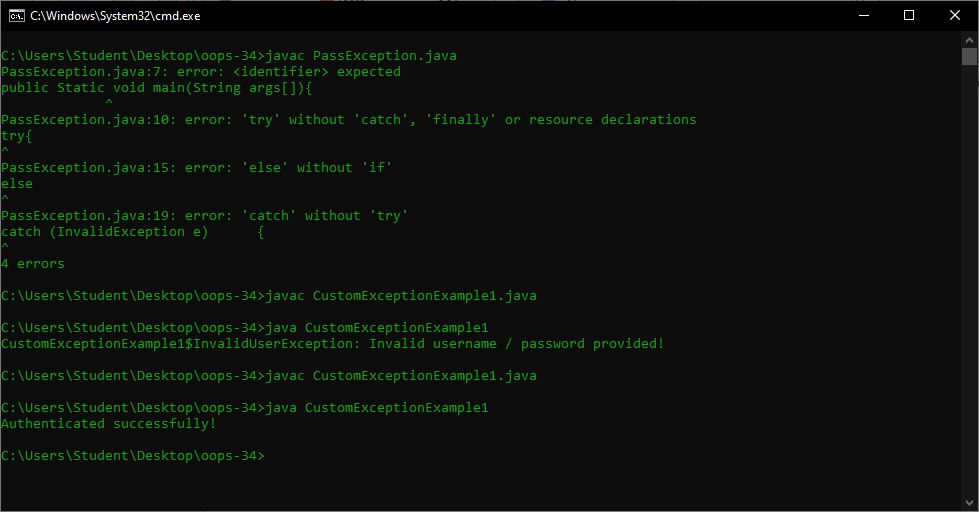
System.out.println(e);

}

}

}

**Output**



**3.Find the average of N positive integers, raising a user defined exception for each negative input.**

**Procedure**

import java.util.\*;

class MyException extends Exception {

public MyException(String value) {

super(value);

}

}

class Main {

public static void main(String args[]) {

int totalNums;

int i;

int temp, count = 0;

int sum = 0;

Scanner sc = new Scanner(System.in);

System.out.println("Total numbers");

totalNums = Integer.parseInt(sc.nextLine());

for (i = 0; i < totalNums; i++) {

try {

temp = Integer.parseInt(sc.nextLine());

if (temp > 0) {

sum += temp;

count += 1;

} else {

throw new MyException(Integer.toString(temp));

}

} catch (MyException ex) {

System.out.print(ex.getMessage());

System.out.println(" - Not a positive number");

}

}

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

System.out.println(count);

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

System.out.println(sum);

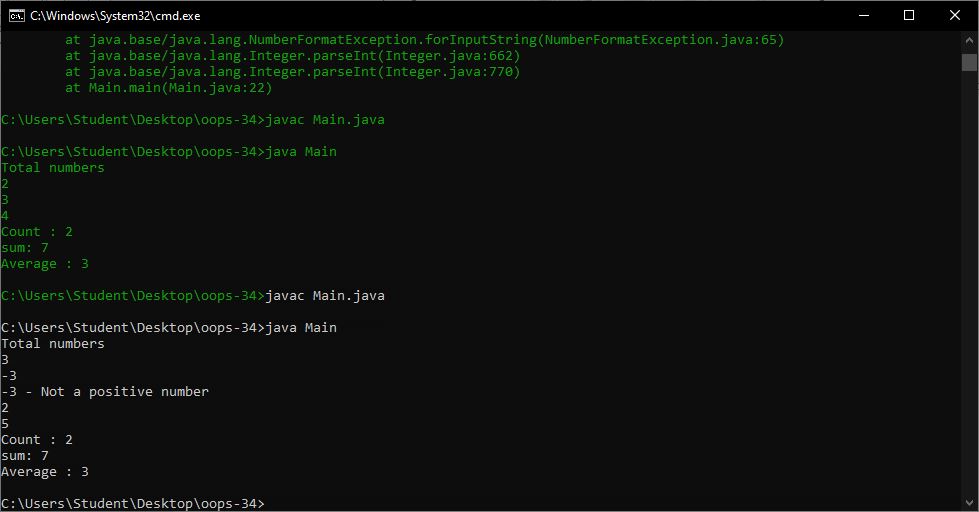
System.out.print("Average : ");

System.out.println(sum / count);

}

}

**Output**

****

**5.Program to create a generic stack and do the Push and Pop operations**.

**Procedure**

class Stack {

private int arr[];

private int top;

private int capacity;

Stack(int size) {

arr = new int[size];

capacity = size;

top = -1;

}

public void push(int x) {

if (isFull()) {

System.out.println("Stack OverFlow");

System.exit(1);

}

System.out.println("Inserting " + x);

arr[++top] = x;

}

public int pop() {

if (isEmpty()) {

System.out.println("STACK EMPTY");

System.exit(1);

}

return arr[top--];

}

public int getSize() {

return top + 1;

}

public Boolean isEmpty() {

return top == -1;

}

public Boolean isFull() {

return top == capacity - 1;

}

public void printStack() {

for (int i = 0; i <= top; i++) {

System.out.print(arr[i] + ", ");

}

}

public static void main(String[] args) {

Stack stack = new Stack(5);

stack.push(1);

stack.push(2);

stack.push(3);

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

stack.printStack();

stack.pop();

System.out.println("\nAfter popping out");

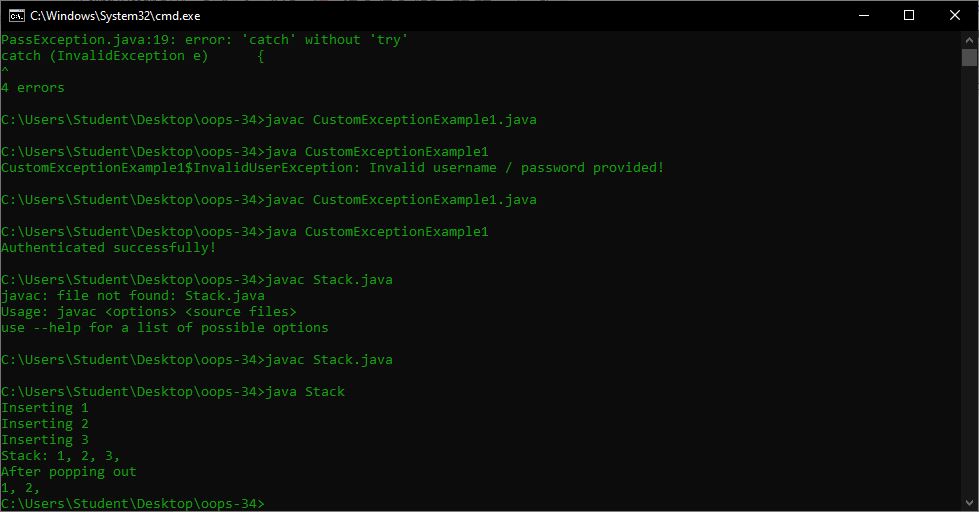
stack.printStack();

}

}

}

**Output**

****