SDM College of Engineering and Technology

Dhavalagiri, Dharwad-580 002. Karnataka State. India.
Email: principal@sdmcet.ac.in, cse.sdmcet@gmail.com
Ph: 0836-2447465/ 2448327 Fax: 0836-2464638 Website: sdmcet.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

ASSIGNMENT-1

[18UCSE508- ADVANCED OBJECT ORIENTED PROGRAMMING]

Course Teacher: Prof. Indira R Umarji

2022- 2023

Submitted by
By

LAVANYA KULKARNI
2SD20CS050
5th Semester B division

1.Problem definition:

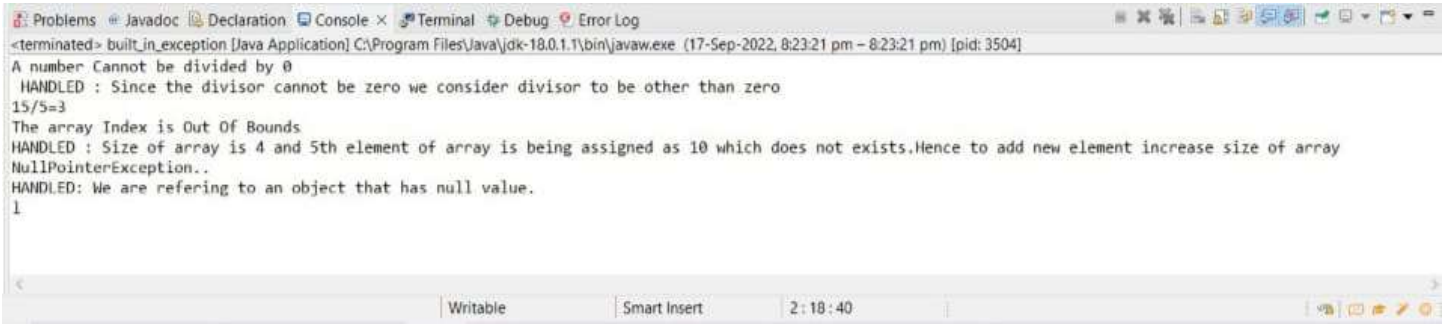
Write a Java program to generate and handle any three built-in exceptions and display appropriate error messages.

Java Program:

```
package assignment_1;
import java.io.*;
import java.util.*;

public class built_in_exception {

    public static void main(String [] args)
  {
     try {
        int a = 15,b = 0;
        int c = a / b;
        System.out.println("Result = " + c); // Divide by zero exception
     }//End of try
     catch (ArithmeticException e) {
        System.out.println("A number Cannot be divided by 0");
     }//catch
     finally {
      System.out.println(" HANDLED : Since the divisor cannot be zero we consider divisor to be other than zero");
      int a = 15,b=5;
      int c=a/b;
      System.out.println(a+ "/" +b+ "=" +c++ );
     }//finally
            try {

                  int a[] = new int[4];
              a[5] = 10;//array index out of bounds
            }
     catch (ArrayIndexOutOfBoundsException e) {
        System.out.println("The array Index is Out Of Bounds");
     }
            finally {
                  System.out.println("HANDLED : Size of array is 4 and 5th element of array is being assigned as 10 which does not exists.Hence to add new element increase size of array");
```

```
                }

                try {
            String a = null;
            System.out.println(a.charAt(0));   // Nullpointerexception
                                                                    // The
    variable has null value but we are trying to refer it
        }
        catch (NullPointerException e) {
            System.out.println("NullPointerException..");
        }
                finally {
                        System.out.println("HANDLED: We are refering to an object
    that has null value.");
                        String a="lavanya";
                        System.out.println(a.charAt(0));
                }
         }
    }
```

OUTPUT:



2.Problem definition:

Write a Java program to read an integer and check whether the number is prime or not. If negative number is entered, throw an exception NegativeNumberNotAllowedException and if entered number is not prime, then throw NumberNotPrimeException.

Java Program:

```
package neg_prime;
import java.util.*;  //import statements
import java.util.Scanner;



class NegativeNumberException extends Exception {
        int num;
                NegativeNumberException(int num){
                        this.num=num;
                        System.out.println(num+ "is a negative number");
                }

}//user-defined negative number exception


class NotPrimeException extends Exception {
        int num;
   public NotPrimeException(int num) {
        this.num=num;
        System.out.println(num+ "is not a prime number");
   }
}//user-defined not primr=e number exception

public class neg_prime {

        public static void main(String[] args) {
      Scanner sc= new Scanner(System.in);
      System.out.println("Enter a number to check if PRIME or NOT");
      int num= sc.nextInt();//input

                try{
                        if(num < 0)
                                throw new NegativeNumberException( num);
```

```
                                        // first check if entered number is negative
                                        // if the number is negative throw
    NegativeNumberNotAllowedException
                }
                catch(Exception e){
                        System.out.println(e);
                }

                int flag=0;
                for (int i = 2;i <= num / 2; i++)
                {
                        if (num%i == 0)
                        {
                                flag = 1;
                                break;
                        }
                }//to check if number is prime or not

                try{
                        if (flag==0)
                                System.out.println(num+ "is prime number");
                        else
                                throw new NotPrimeException(num);  // and if the number is
    not prime throw NumberNotPrimeException
                }
                catch (Exception e) {
             System.out.println(e);
          }
            }//end of main
}//end of neg_prime class
```

OUTPUT:


Problems Javadoc Declaration Console Terminal Debug Error Log
<terminated> neg_prime [Java Application] C:\Program Files\Java\jdk-18.0.1.1\bin\javaw.exe (17-Sep-2022, 10:47:22 pm – 10:47:30 pm) [pid: 10124]
Enter a number to check if PRIME or NOT
-9
-9is a negative number
neg_prime.NegativeNumberException
-9is prime number
Writable
Smart Insert
43 : 43 : 1091


Problems Javadoc Declaration Console Terminal Debug Error Log
<terminated> neg_prime [Java Application] C:\Program Files\Java\jdk-18.0.1.1\bin\javaw.exe (17-Sep-2022, 10:53:10 pm – 10:53:14 pm) [pid: 19652]
Enter a number to check if PRIME or NOT
178
178is not a prime number
neg_prime.NotPrimeException

3.Problem definition:

Write a Java program to perform the following operations:

a) Read a line of text

b) Search for a sub-string SDMCET (case insensitive search)

c) If found, then print success message

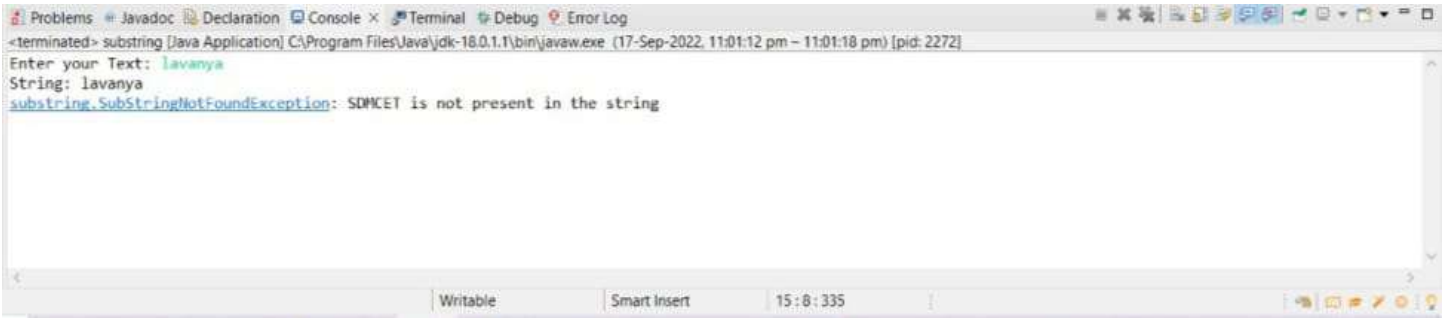
d) Otherwise throw an exception SubStringNotFoundException with appropriate message

Java Program: package neg_prime;

```
package substring;
import java.util.*;//import statements
import java.util.Scanner;

class SubStringNotFoundException extends Exception {

   public SubStringNotFoundException(String str) {
      super(str);
   }
}//user-defined exception class

public class substring
{
        public static void main(String args[])
        {
                Scanner in = new Scanner(System.in);
       System.out.print("Enter your Text: ");
                String line= in.nextLine(); //a) reading a line of text
                System.out.println("String: "+line);
                String str1 ="SDMCET";

                try{
                        line=line.toUpperCase();
                        boolean result = line.contains(str1);//b)searching for substring
SDMCET(case insensitive)
                        if(result) {
                                System.out.println(str1+ " is present in the string.");//c)success
message
                        }
```

```
                    else {
                        throw new SubStringNotFoundException("SDMCET is not
    present in the string");//d)exception
                    }
                }
                catch(Exception e){
                    System.out.println(e);
                }
            }//end of main
}//end of substring class
```

OUTPUT:



4. Problem definition:

Write a Java program to perform the following operations:

a) Create a file named Alphabets.txt and insert appropriate data into it

b) Read the file and copy all the consonants into another file named Consonants.txt.

c) If vowel is encountered, throw an exception VowelNotAllowedException and continue until end of file

Java Program:

```
import java.util.*;

import java.io.*;
```

```
class vowels_consonant{
public static void main(string[] args){
try{
FileInputStream fin=new
FileInputStream("C:\Users\lavanya\Documents\5th sem\Alphabet.txt"); //creation of file alphabets.txt and insertion of data
FileOutputStream fout=new FileOutputStream("C:\Users\lavanya
\Documents\5th sem\consonant.txt");
int ch;
while(ch=fin.read()!=-1){
if(ch=='a'||ch=='e'||ch=='i'||ch=='o'||ch=='u'){
throw new vowelNotAllowedException(); //throw exception if vowel encountered
}
else
fout.write(ch); //copy consonants into consonants.txt
}
}catch(vowelNotAllowedException e){
System.out.println(e.toString());
}
}
}
class vowelNotAllowedException extends Exception{
public String toString(){
return "vowels are not allowed";
}
}
```

OUTPUT:

5.Problem Definition:

Write a Java program to implement the following scenario:

a) Create a file named Integers.txt and insert n-random integers into it.

b) Create three threads T1, T2 and T3 that read n/3 integers in sequence of occurrenceof numbers from the file and sort the read n/3 integers.

c) Thread T4 waits for all the threads T1, T2 and T3 to complete sorting, then sorts and outputs the entire list of sorted numbers to another file named SortedIntegers.txt

Java Program:

```
mport java.io.File;
import java.io.FileNotFoundException;
import java.io.FileWriter;
import java.io.IOException;
import java.util.Arrays;
import java.util.Scanner;

class five
{
	private static int arr[];

	public static void main(String[] args) throws
FileNotFoundException,IOException,InterruptedException
	{
	    File ipFile = new File("Integers.txt");//creating file integers.txt
      File opFile = new File("SortedIntegers.txt");

      FileWriter opWriter = new FileWriter(opFile);

      Scanner sc = new Scanner(ipFile);
      int size = sc.nextInt();

	    arr = new int[size];
      int i = 0;
      while (sc.hasNext()) {
```

```
    arr[i++] = sc.nextInt();
}

Thread T1 = new Thread() { //thread 1
    public void run() {
        ThreadSorting(arr, 0, (size / 3) - 1);
    }
};

Thread T2 = new Thread() { //thread 2
    public void run() {
        ThreadSorting(arr, (size / 3), ((size / 3) * 2) - 1);
    }
};

Thread T3 = new Thread() { //thread 3
    public void run() {
        ThreadSorting(arr, ((size / 3) * 2), (size - 1));
    }
};

Thread T4 = new Thread() { //thread 4
    public void run() {
        ThreadSorting(arr, 0, size - 1);
    }
```

```
    };

    T1.start();
    T1.join();
    T2.start();
    T2.join();
    T3.start();
    T3.join();
    T4.start();
    T4.join();

    for (int num : arr) {
        opWriter.append(String.valueOf(num) + " ");
    }
    opWriter.close();

}

public static void ThreadSorting(int arr[], int start, int end) {
    int tempArr[] = new int[end - start + 1];
    int tempIndex = 0;
    for (int i = start; i <= end; i++) {
        tempArr[tempIndex++] = arr[i];
    }
    Arrays.sort(tempArr);
```

```
        int index = start;
        for (int n : tempArr) {
            arr[index++] = n;
        }


    }
}
```

OUTPUT:

Output is stored in txt file.

Integer.txt : 12 56 83 98 35

SortedInteger.txt : 12 35 56 83 98