Department of Computer Science & Engineering, SDMCET, Dharwad-2

AOOP Assignment Submission Report

[Submitted as part of CTA Assignment No-1]

Course:	Advanced Object-Oriented Programming	Course Code:	18UCSE508
Semester:	V	Division:	A

Submitted by:

USN:	2SD20CS014	Name:	AMRIT KUMAR SINGH

1. Problem Definition:

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

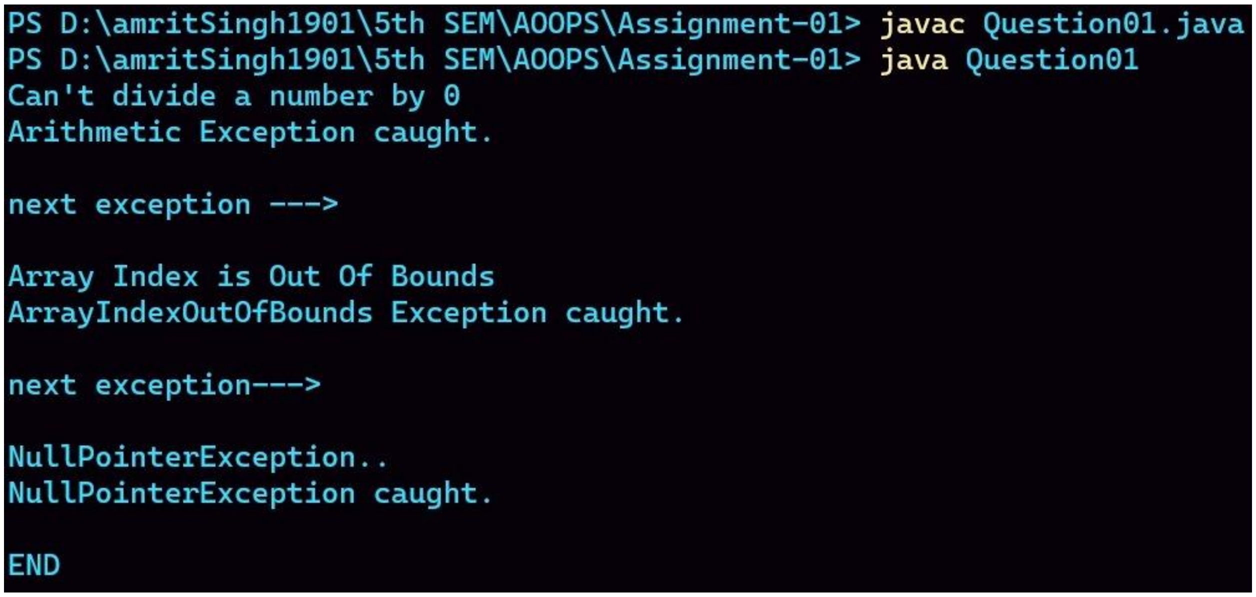
2. Java Program:

```
class Question01 {
public static void main(String args[])
   {
      //Arithmetic Exception
      try {
         int a = 19, b = 0;
         int c = a / b;
         System.out.println("Result = " + c);
      }
      catch (ArithmeticException e) {
         System.out.println("Can't divide a number by 0");
         System.out.println("Arithmetic Exception caught.");
      }finally{
         System.out.println();
         System.out.println("next exception --->");
         System.out.println();
      }

      //ArrayIndexOutOfBounds Exception

      try {
```

```
    int a[] = new int[2];
    a[3] = 19;

}
catch (ArrayIndexOutOfBoundsException e) {
    System.out.println("Array Index is Out Of Bounds");
    System.out.println("ArrayIndexOutOfBounds Exception caught.");
}finally{
    System.out.println();
    System.out.println("next exception--->");
    System.out.println();
}



//nullPointer Exception
try {
    String str= null;
    System.out.println(str.charAt(0));
}
catch (NullPointerException e) {
    System.out.println("NullPointerException..");
    System.out.println("NullPointerException caught.");
}finally{

    System.out.println();
    System.out.println("END");
    System.out.println();
}
```

```
    }
}
```

3. Screen Shots of Execution:



```
PS D:\amritSingh1901\5th SEM\AOOPS\Assignment-01> javac Question01.java
PS D:\amritSingh1901\5th SEM\AOOPS\Assignment-01> java Question01
Can't divide a number by 0
Arithmetic Exception caught.

next exception --->

Array Index is Out Of Bounds
ArrayIndexOutOfBounds Exception caught.

next exception--->

NullPointerException..
NullPointerException caught.

END
```

1. Problem Definition:

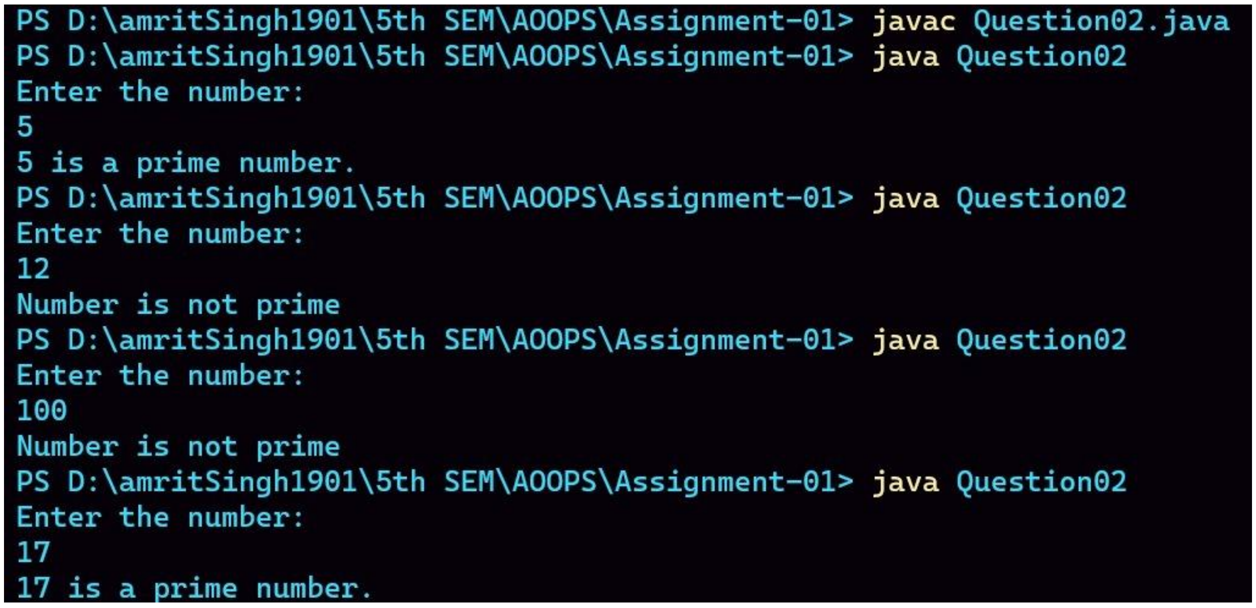
Write a Java program to read an integer and check whether the number is prime or not. If negative number is entered, throw an exception NegativeNumberNotAllowedException and if enetered number is not prime, then throw NumberNotPrimeException.

2. Java Program:

```
import java.util.Scanner;


public class Question02 {


    public static void main(String[] args) {


        try (Scanner sc = new Scanner(System.in)) {
            System.out.println("Enter the number:");
            boolean flag = false;



            try{



               int num = sc.nextInt();



                if(num < 0)
                throw new NegativeNumberNotAllowedException("Number is negative");
            for (int i = 2; i <= num / 2; ++i) {

                if (num % i == 0) {
                    flag = true;
                    break;
                }
            }

            if (!flag)
                System.out.println(num + " is a prime number.");
            else
                throw new NumberNotPrimeException("Number is not prime");

            }catch(NegativeNumberNotAllowedException | NumberNotPrimeException e){

            }
        }
  }
}
  //NegativeNumberNotAllowed  User defined Exception
class  NegativeNumberNotAllowedException extends Exception{
    public NegativeNumberNotAllowedException(String str){

        System.out.println(str);
    }
```

```
}


  //NumberNotPrime User defined Exception
class  NumberNotPrimeException  extends Exception{
    public NumberNotPrimeException(String str){
        System.out.println(str);
    }
}
```

3. Screen Shots of Execution:



```
PS D:\amritSingh1901\5th SEM\AOOPS\Assignment-01> javac Question02.java
PS D:\amritSingh1901\5th SEM\AOOPS\Assignment-01> java Question02
Enter the number:
5
5 is a prime number.
PS D:\amritSingh1901\5th SEM\AOOPS\Assignment-01> java Question02
Enter the number:
12
Number is not prime
PS D:\amritSingh1901\5th SEM\AOOPS\Assignment-01> java Question02
Enter the number:
100
Number is not prime
PS D:\amritSingh1901\5th SEM\AOOPS\Assignment-01> java Question02
Enter the number:
17
17 is a prime number.
```

1. Problem Definition:

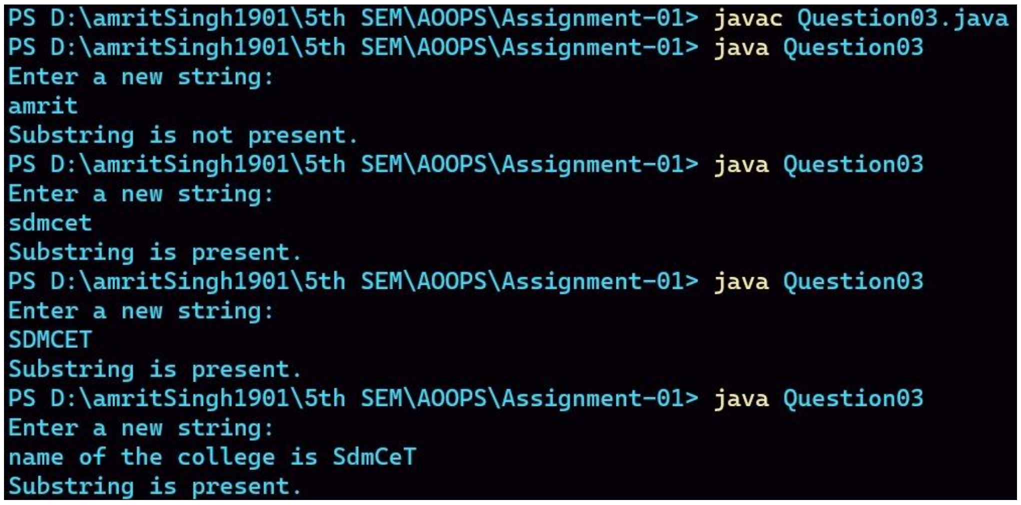
Write a Java Program to perform the following operations:

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 and continue until end of file

2. Java Program:

```
import java.util.Scanner;

public class Question03 {

    public static void main(String[] args) {

        try (Scanner sc = new Scanner(System.in)) {


            System.out.println("Enter a new string:");


            try{

            String str = sc.nextLine();


            String testString = "Sdmcet";


            boolean check = str.toLowerCase().contains(testString.toLowerCase());


            if(check){


                System.out.println("Substring is present.");

            }else{

                throw new SubStringNotFoundException("Substring is not present.");
            }



    }catch(SubStringNotFoundException e){}

        }

    }

}


    //subStringNotFound user defined exception

class  SubStringNotFoundException  extends Exception{


    public SubStringNotFoundException(String str){


        System.out.println(str);

    }

}
```

3. Screen Shots of Execution:



```
PS D:\amritSingh1901\5th SEM\AOOPS\Assignment-01> javac Question03.java
PS D:\amritSingh1901\5th SEM\AOOPS\Assignment-01> java Question03
Enter a new string:
amrit
Substring is not present.
PS D:\amritSingh1901\5th SEM\AOOPS\Assignment-01> java Question03
Enter a new string:
sdmcet
Substring is present.
PS D:\amritSingh1901\5th SEM\AOOPS\Assignment-01> java Question03
Enter a new string:
SDMCET
Substring is present.
PS D:\amritSingh1901\5th SEM\AOOPS\Assignment-01> java Question03
Enter a new string:
name of the college is SdmCeT
Substring is present.
```

1. Problem Definition:

Write a Java program to perform the following operations:

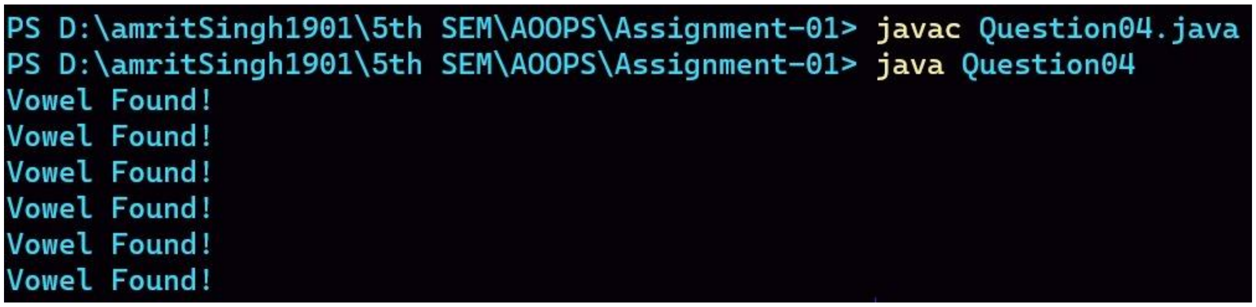
a) Create a file named Alphabets.txt and insert appropriate data into it
b) Read the file and copy all the consonants into another file named Consonants.txt
c) If vowel is encountered, throw an exception VowelNotAllowedException and continue until end of file

2. Java Program:

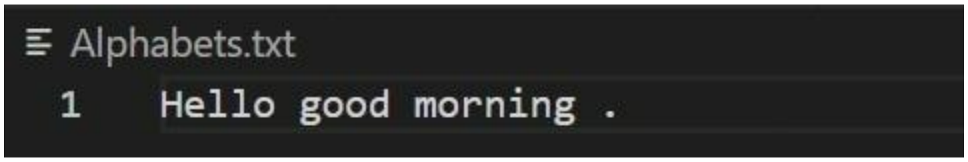
```
import java.io.BufferedReader;

import java.io.File;

import java.io.FileNotFoundException;

import java.io.FileReader;

import java.io.FileWriter;

import java.io.IOException;


class Question04 {
    public static void main(String[] args) throws FileNotFoundException, IOException {


        //input file
        File inputFile = new File("Alphabets.txt");
        FileReader inputReader = new FileReader(inputFile);



        //output file
        File outputFile = new File("Consonants.txt");
        FileWriter outputWriter = new FileWriter(outputFile);


        BufferedReader Reader = new BufferedReader(inputReader);
        int temp;

        while ((temp = Reader.read()) != -1) {
            char c = (char) temp;

            if (Question04.isVowel(c) == false) {
                outputWriter.append(c);
            } else {
                try {
                    throw new VowelNotAllowedException("Vowel Found!");
                } catch (VowelNotAllowedException e) {}
            }
        }
        outputWriter.close();
        Reader.close();

    }
    //checking for vowels
    public static boolean isVowel(char c) {
        if (c == 'a' || c == 'A')
            return true;
        else if (c == 'e' || c == 'E')
            return true;
        else if (c == 'i' || c == 'I')
            return true;
        else if (c == 'o' || c == 'O')
            return true;
        else if (c == 'u' || c == 'U')
```

```
            return true;
        else
            return false;
    }
}

//vowelNotAllowed user defined exception
class VowelNotAllowedException extends Exception {
    VowelNotAllowedException(String str) {
        System.out.println(str);
    }
}
```

3. Screen Shots of Execution:

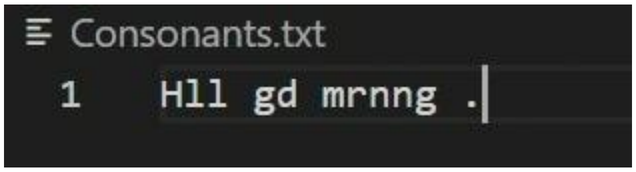


```
PS D:\amritSingh1901\5th SEM\AOOPS\Assignment-01> javac Question04.java
PS D:\amritSingh1901\5th SEM\AOOPS\Assignment-01> java Question04
Vowel Found!
Vowel Found!
Vowel Found!
Vowel Found!
Vowel Found!
Vowel Found!
```



Alphabets.txt

```
Hello good morning .
```



Consonants.txt

```
Hll gd mrnng .
```

1. 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 occurrence of numbers form the file and sort the read n/3 integers
c) Thread T4 waits for all the thread T1, T2 and T3 to complete sorting, then sorts and outputs the entire list of sorted numbers to another file named SortedIntegers.txt

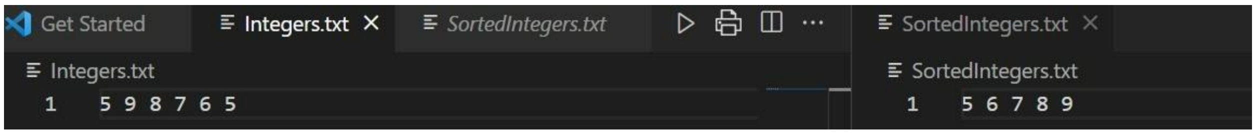
2. Java Program:

```
import java.io.File;

import java.io.FileNotFoundException;

import java.io.FileWriter;

import java.io.IOException;

import java.util.Arrays;

import java.util.Scanner;


class Question05 {

    private static int arr[];


    public static void main(String[] args) throws FileNotFoundException,
InterruptedException, IOException {

        File ipFile = new 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();
}
sc.close();

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

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

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

Thread T4 = new Thread() {
    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;
        }
    }
}
```

3. Screen Shots of Execution:



Integers.txt

```
5 9 8 7 6 5
```

SortedIntegers.txt

```
5 6 7 8 9
```