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:	B

Submitted by:

USN:	2SD20CS044	Name:	JASHANDEEP SINGH

1. Problem Definition:

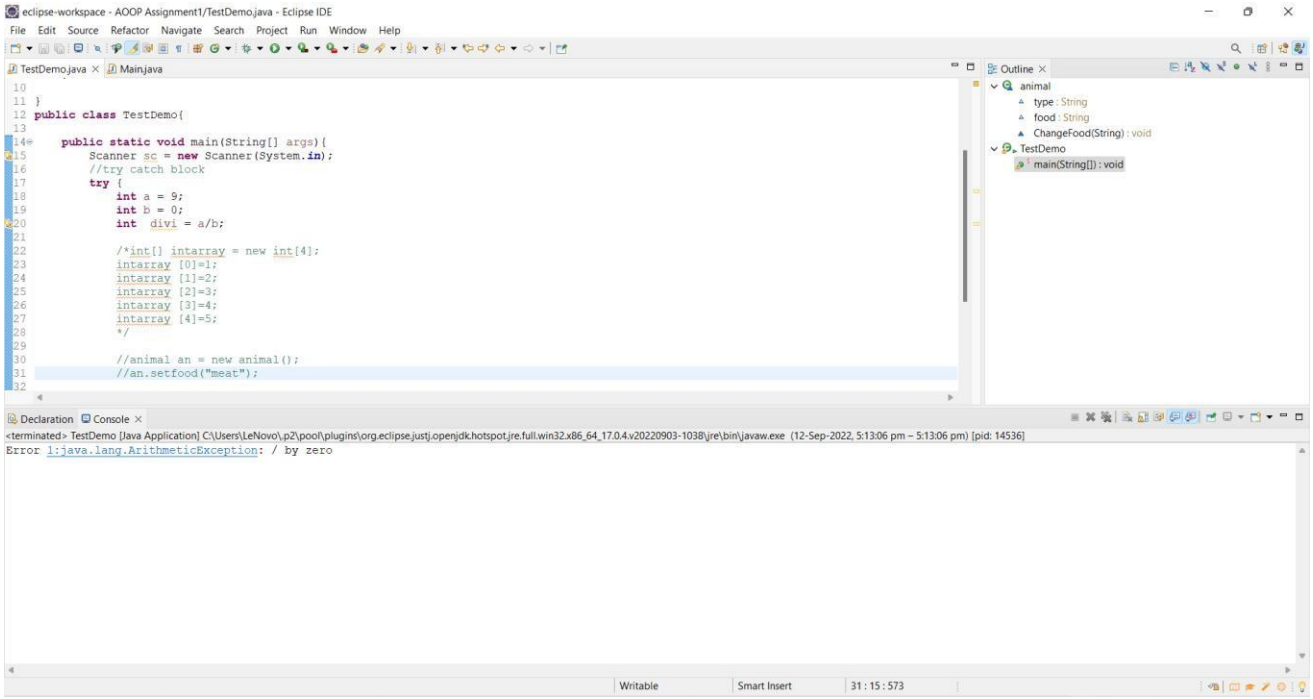
Write a Java program to generate and handle any three built-in exceptions and display appropriate error messages.

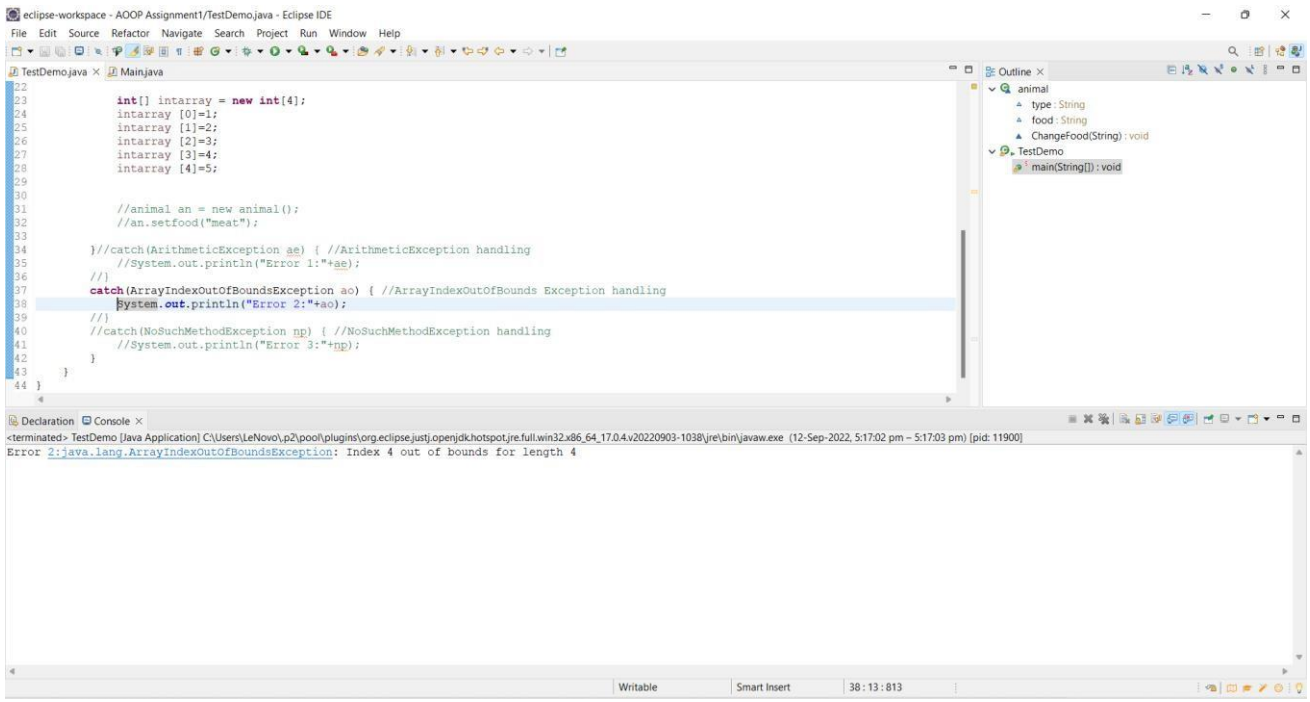
Java Program:

```
import java.util.Scanner;

//class creation

class animal{

	String type; //instance variable

	String food;


	void ChangeFood(String food) { //method declaration

		this.food=food;

	}


}

public class TestDemo {


	public static void main(String[] args){

		Scanner sc = new Scanner(System.in);

		//try catch block

		try {

			System.out.println("Enter value of a and b:");

			int a = sc.nextInt();

			int b = sc.nextInt();

			int divi = a/b;
			int[] intarray = new int[4];
			intarray [0]=1;
			intarray [1]=2;
			intarray [2]=3;
			intarray [3]=4;
			intarray [4]=5;
```

```
                animal an = new animal();
                an.setfood("meat");


            }catch(ArithmeticException ae) { //ArithmeticException handling
                System.out.println("Error 1:"+ae);
            }
            catch(ArrayIndexOutOfBounds ao) { //ArrayIndexOutOfBounds Exception handling
                System.out.println("Error 2:"+ao);
            }
            catch(NoSuchMethodException np) { //NoSuchMethodException handling
                System.out.println("Error 3:"+np);
            }
        }
    }
```

Screen Shots of Execution:





2.Problem Definition:

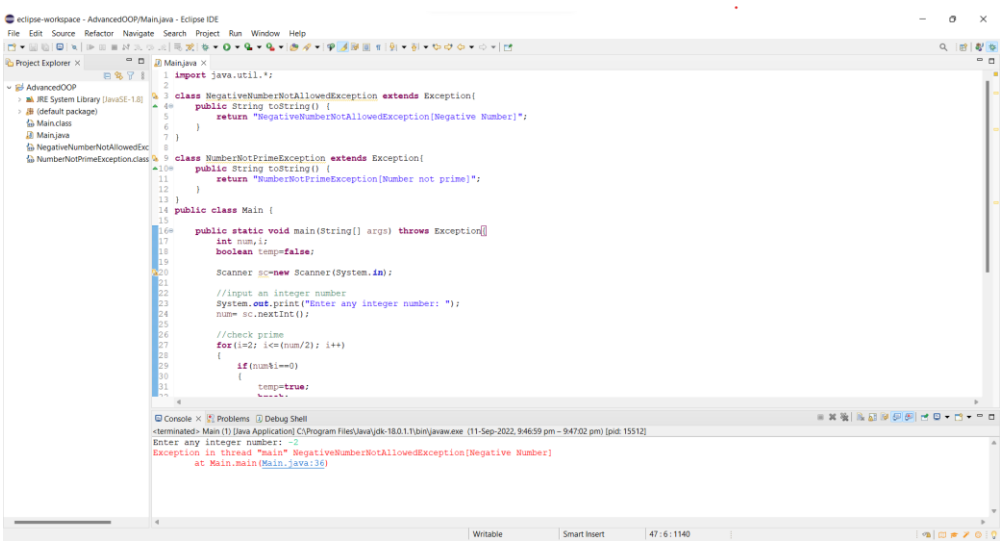
Write a Java program to read an integer and check whether the number is prime or not. If negative number is entered, throw an exception NegativeNumberNotAllowedException and if entered number is not prime, then throw NumberNotPrimeException.

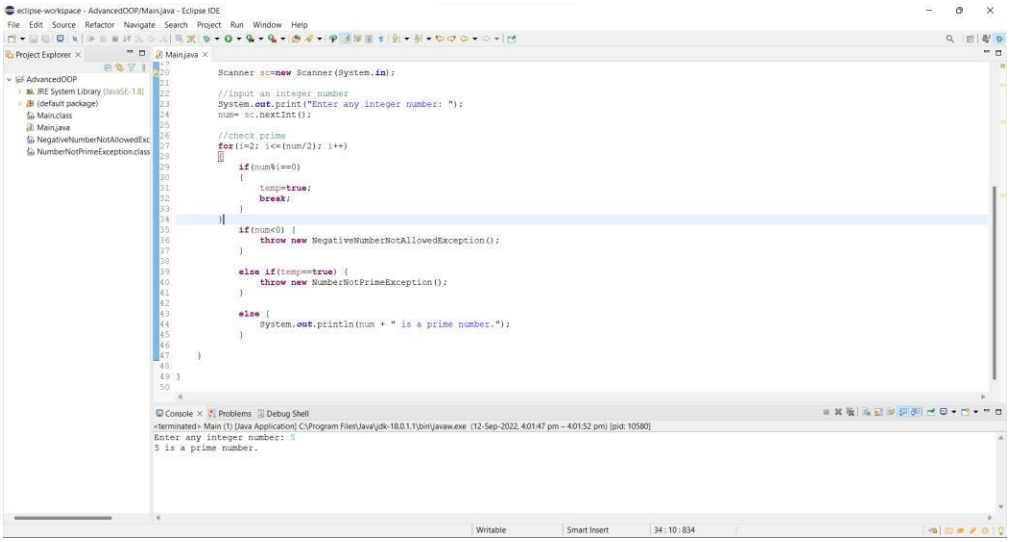
Java Program:

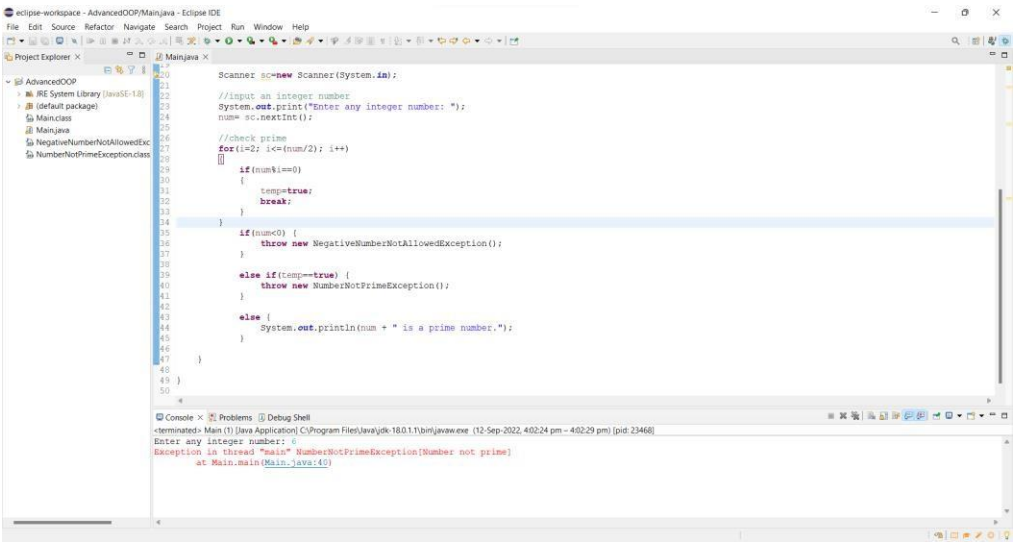
```
import java.util.*;

class NegativeNumberNotAllowedException extends Exception{
	public String toString() {
		return "NegativeNumberNotAllowedException[Negative Number]";
	}
}

class NumberNotPrimeException extends Exception{
	public String toString() {
		return "NumberNotPrimeException[Number not prime]";
	}
}
public class Main {

	public static void main(String[] args) throws Exception{
		int num,i;
	boolean temp=false;

	Scanner sc=new Scanner(System.in);

	//input an integer number
	System.out.print("Enter any integer number: ");
	num= sc.nextInt();

	//check prime
	for(i=2; i<=(num/2); i++)
	{
	   if(num%i==0)
	   {
	      temp=true;
	      break;
	   }
	}
	if(num<0) {
		throw new NegativeNumberNotAllowedException();
	}
```

```
        else if(temp==true) {
                throw new NumberNotPrimeException();
        }

        else {
                System.out.println(num + " is a prime number.");
        }


        }

}//End of Main
```

Screen Shots of Execution:







3.Problem Definition:

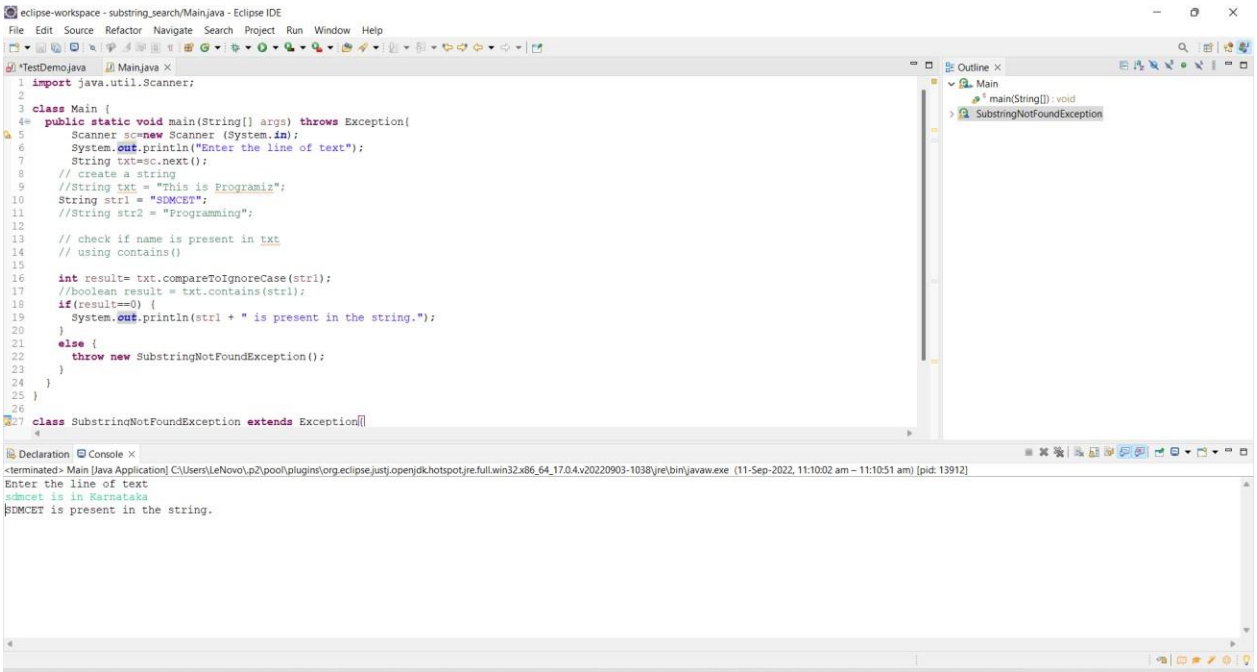
Write a Java program to perform the following operations:

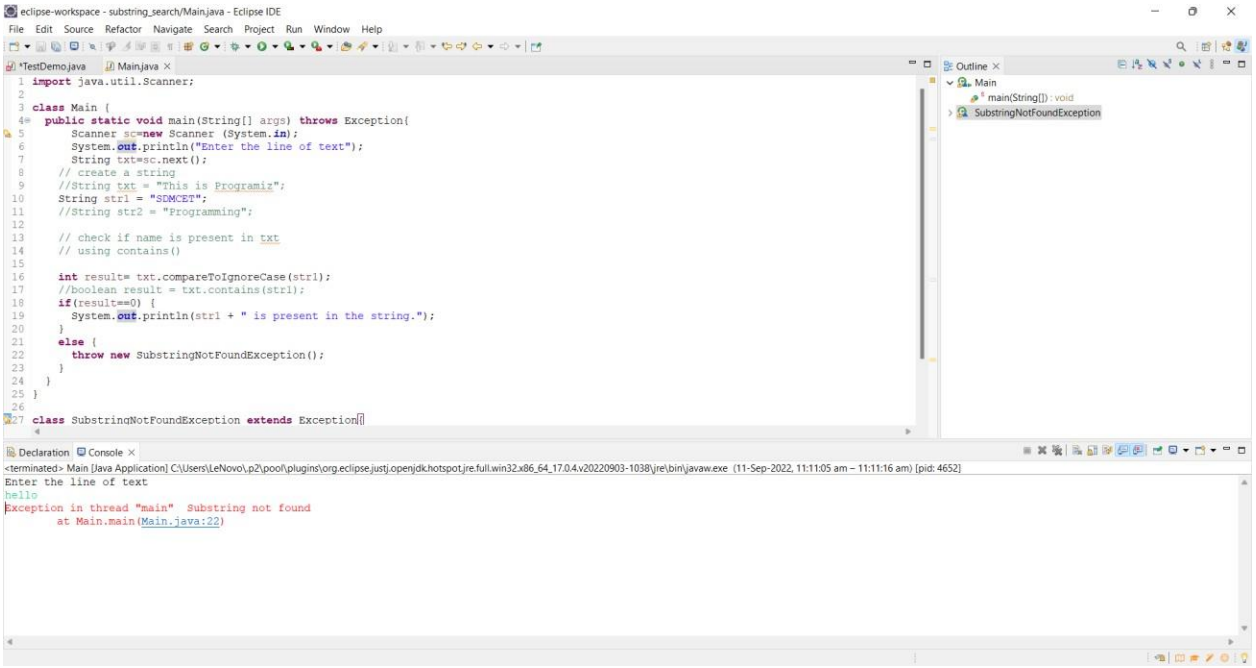
a) Read a line of text
b) Search for a sub-string SDMCET (case insensitive search)
c) If found, then print success message
d) Otherwise throw an exception SubStringNotFoundException with appropriate message

Java Program:

```
import java.util.Scanner;

class Main {
  public static void main(String[] args) throws Exception{
        Scanner sc=new Scanner (System.in);
        System.out.println("Enter the line of text");
        String txt=sc.next();
   // create a string
   //String txt = "This is Programiz";
   String str1 = "SDMCET";
   //String str2 = "Programming";

   // check if name is present in txt
   // using contains()

   int result= txt.compareToIgnoreCase(str1);
   //boolean result = txt.contains(str1);
   if(result==0) {
     System.out.println(str1 + " is present in the string.");
   }
   else {
     throw new SubstringNotFoundException();
   }
  }
}

class SubstringNotFoundException extends Exception{
      public String toString() {
                        return " Substring not found";

      }
}
```

Screen Shots of Execution:





4.Problem Definition:

Write a Java program to perform the following operations:

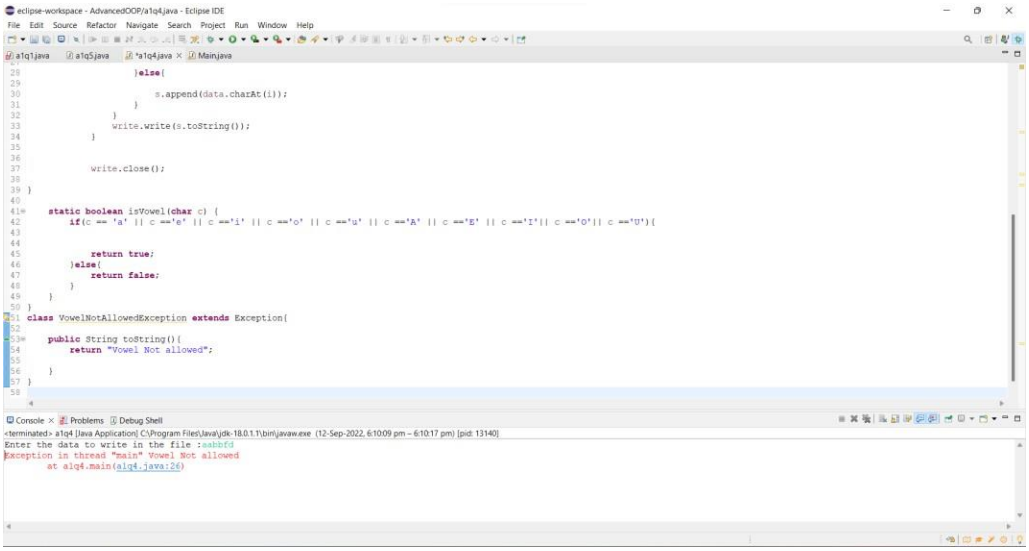
a) Create a file named Alphabets.txt and insert appropriate data into it
b) Read the file and copy all the consonants into another file named Consonants.txt
c) If vowel is encountered, throw an exception VowelNotAllowedException and continue until end of file

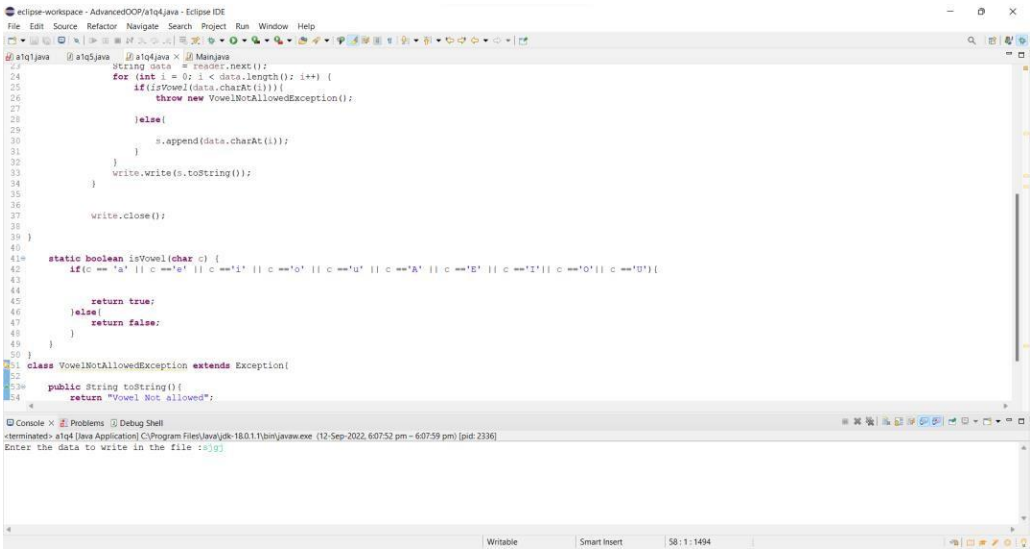
Java Program:

```
import java.util.Scanner;
import java.io.*;


class a1q4 {

    public static void main(String[] args)throws Exception {



            FileWriter w = new FileWriter("Alphabets.txt");
            Scanner sc = new Scanner(System.in);
            System.out.print("Enter the data to write in the file :");
            String str = sc.nextLine();
            w.write(str);
            w.close();
            File file = new File("Alphabets.txt");

            Scanner reader = new Scanner(file);
            StringBuilder s = new StringBuilder();
            FileWriter write = new FileWriter("Consonant.txt");
            while(reader.hasNext()){
                String data  = reader.next();
                for (int i = 0; i < data.length(); i++) {
                    if(isVowel(data.charAt(i))){
                        throw new VowelNotAllowedException();

                    }else{

                        s.append(data.charAt(i));
                    }
                }
                write.write(s.toString());
            }


            write.close();

}
```

```
    static boolean isVowel(char c) {
        if(c == 'a' || c =='e' || c =='i' || c =='o' || c =='u' || c =='A' || c =='E' || c =='I'|| c =='O'|| c =='U'){


            return true;
        }else{
            return false;
        }
    }



}
```

Screen Shots of Execution:





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 occurrence of 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:

```
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 a1q5 {
    private static int arr[];

    public static void main(String[] args) throws FileNotFoundException, InterruptedException,
IOException {
        File inputFile = new File("Integers.txt");

        File outputFile = new File("SortedIntegers.txt");

        FileWriter opWriter = new FileWriter(outputFile);

        Scanner sc = new Scanner(inputFile);

        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;
        }
    }
}
```

Screen Shots of Execution: