**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: | 2SD20CS084 | Name: | Rajkumar Uppar |

**1. Problem Definition:**

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

error messages.

**2. Java Program:**

**3. Screen Shots of Execution:**

Q2. 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.

**2. Java Program:**

**3. Screen Shots of Execution:**

Q3. 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

**2. Java Program:**

import java.util.\*;

import java.util.Scanner;

/\*\*

\* Assignment\_3

\*/

public class Assignment\_3 {

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

Scanner sc= new Scanner(System.in);

System.out.print("Enter the String =");

String testString = sc.nextLine();

testString = testString.toUpperCase();

String subString="SDMCET";

int i=0,j=0;

while(i<testString.length()){

if(testString.charAt(i)==subString.charAt(j) && j<subString.length()-1){

i++;

j++;

}else{

i++;

}

} //end of while

if(j == subString.length()-1){

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

}else{

throw new SubstringNotFoundException("Substring is not found !! please enter the valid input");

}

}

class SubstringNotFoundException extends Exception{

String str;

SubstringNotFoundException(String str){

this.str = str;

}

public String toString() {

return this.str;

}

}

**3. Screen Shots of Execution:**

Q4. 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.util.Scanner;

import java.io.\*;

public class Assignment\_4 {

public static void main(String[] args)

try{

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("Consonate.txt");

while(reader.hasNext()){

String data = reader.next();

for (int i = 0; i < data.length(); i++) {

if(isVowel(data.charAt(i))){

System.out.println("vowel found " + data.charAt(i));

}else{

s.append(data.charAt(i));

}

}

write.write(s.toString());

}

write.close();

}

catch(VowelNotFoundException v){

System.out.println("vowel found");

}catch(FileNotFoundException e){

System.out.println(e);

}catch(IOException ex){

System.out.println(ex);

}

}

static boolean isVowel(char c) throws VowelNotFoundException{

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;

}

}

class VowelNotFoundException extends Exception{

String str;

VowelNotFoundException(String str){

this.str = str;

}

public String toString() {

return this.str;

}

}

}

**3. Screen Shots of Execution:**

Q5. 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

**2. Java Program:**

**3. Screen Shots of Execution:**