**QUESTION 1**

**ALGORITHM**

1. START
2. Take input in variable str
3. Reverse str
4. Replace all punctuations in str with “”
5. Loop through the characters of str and add characters to String t
6. IF t.length = 2 or t.length = 3
7. Convert t to Integer
8. Check if (char) t >= a/A and (char) t <= z/Z
9. Add (char) t to String f
10. IF (char) t = 32 add “ ” to String f
11. Print String f
12. END

**SOURCE CODE**

import java.util.\*;

class AsciiDecode

{

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

System.out.println("Enter encoded message : ");

String s=sc.nextLine();

s = s.replace(“.”, “”);

s = s.replace(“?”, “”);

s = s.replace(“!”, “”);

s=reverseString(s);

System.out.println(toAscii(s));

}

public static String reverseString(String str){

char ch[]=str.toCharArray();

String rev="";

for(int i=ch.length-1;i>=0;i--){

rev+=ch[i];

}

return rev;

}

public static String toAscii(String s)

{

String t="";

String f="";

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

{

if(s.charAt(i)!=' ')

{

t+=s.charAt(i);

if(t.length()==2||t.length()==3)

{

int t1 = Integer.parseInt(t);

if(t1>=65&&t1<=90)

{

f+=(char)t1;

t="";

}

else if(t1>=97&&t1<=122)

{

f+=(char)t1;

t="";

}

else if(t1==32)

{

f+=(char)t1;

t="";

}

else

continue;

}

}

}

return f;

}

}

Graphical user interface, text, application, email

Description automatically generated

VARIABLE DESCRIPTION

|  |  |  |
| --- | --- | --- |
| Variable | Data Type | Description |
| s | String | To input the encoded message |
| ch[] | Char array | To store the string as a temporary variable to reverse. |
| rev | String | To store the reverse and return the string |
| i | int | Loop variable |
| t | String | Temporary variable |
| f | String | To store final string to be printed |

METHOD DESCRIPTION

|  |  |  |
| --- | --- | --- |
| Method Name | Return Type | Description |
| main | void | To run the program |
| reverseString | String | To reverse the string |
| toAscii | void | To convert the ASCII to string to be printed |

**QUESTION 2**

**ALGORITHM**

1. START
2. Take input n
3. Take input str
4. Count number of sentences in str
5. If number of sentences > n, EXIT
6. Else Replace ‘!’, ‘.’ and ‘?’ with ‘’ in str
7. Make an array of words in str ar[]
8. Print the number of words in str => ar.length
9. Loop through ar and count frequency of each word using public int count\_freq(String s, String[] ar)
10. Sort the frequency array in ascending order with corresponding words
11. Print words with frequency

**SOURCE CODE**

import java.util.Scanner;

public class SentenceLimit {

public int count\_freq(String s, String[] ar) {

int c = 0;

for (String i : ar) {

if (i.equals(s)) c++;

}

return c;

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

SentenceLimit obj = new SentenceLimit();

int n = sc.nextInt();

sc.nextLine();

String str = sc.nextLine();

int c = 0;

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

char ch = str.charAt(i);

if (ch == '.' || ch == '!' || ch == '?') c++;

}

if (c > n) System.out.println("Limit exceeded!");

else {

str = str.replace("!", "");

str = str.replace(".", "");

str = str.replace("?", "");

String[] ar = str.split(" ");

System.out.printf("Total words = %d\n", ar.length);

int[] freq = new int[50];

String[] words = new String[50];

int k = 0;

for (String i : ar) {

freq[k] = obj.count\_freq(i, ar);

words[k] = i;

k++;

}

int cl = 0;

for (int i = 0; i < freq.length - 1; i++) {

if (freq[i] > freq[i + 1] && cl <= (freq.length - 3) \* freq.length) {

int temp = freq[i];

String wd = words[i];

freq[i] = freq[i + 1];

freq[i + 1] = temp;

words[i] = words[i + 1];

words[i + 1] = wd;

cl++;

i = -1;

}

}

String s = "";

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

try

{

if (!s.contains(words[i]))

{

if (words[i] != null)

System.out.printf("%s : %d\n", words[i], freq[i]);

s += words[i];

}

}

catch (Exception e)

{

continue;

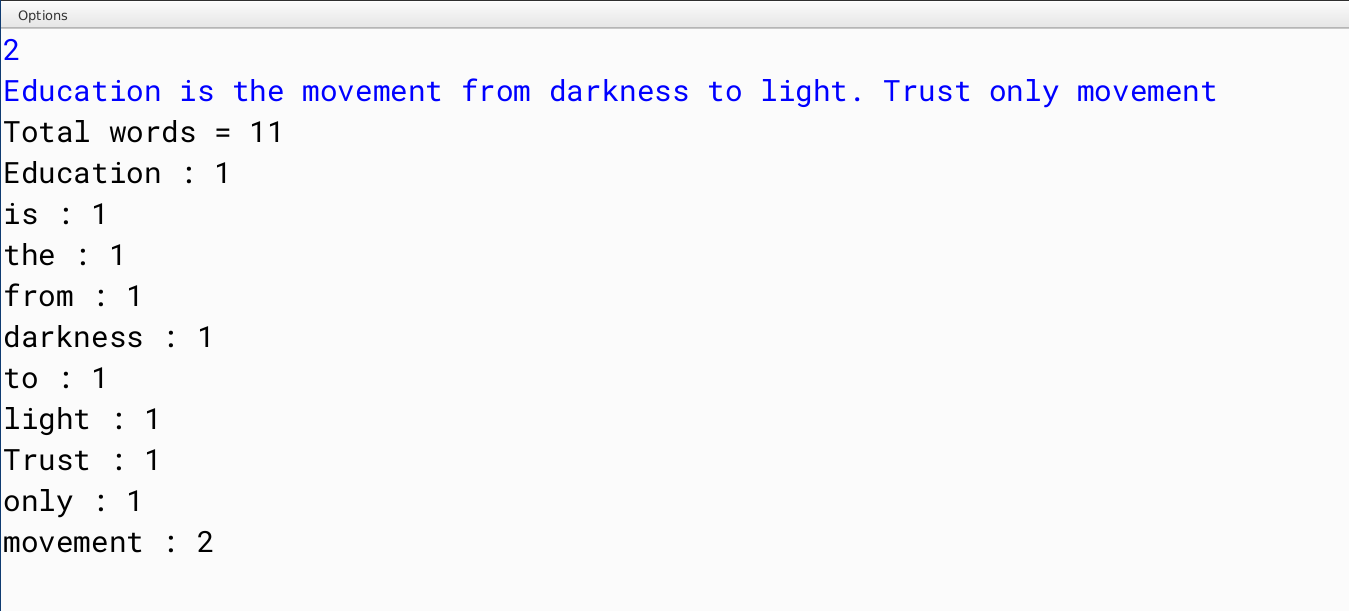
}

}

}

}

}



**VARIABLE DESCRIPTION**

|  |  |  |
| --- | --- | --- |
| Name | Data type | Description |
| n | int | number of sentences |
| c | int | counter variable |
| i | int, String | loop variable |
| k | int | counter variable |
| str | String | User input string |
| ar | String[] | Array of words in str |
| freq | String[] | Frequency of words in str |
| words | String[] | Array of unique words in str |

**Method :: public int count\_freq(String s, String[] ar)**

**DESCRIPTION**

Counts the frequency of the given word s in the array of Strings ‘ar’ containing all the words in the String str.

|  |  |  |
| --- | --- | --- |
| Parameter | Data type | Description |
| s | String | word whose frequency is to be found |
| ar | String | Array of words in str |

|  |  |  |
| --- | --- | --- |
| Name | Data type | Description |
| c | int | Counts frequency of s in ar |
| i | String | Loop variable |

**QUESTION 3**

**ALGORITHM**

1. START
2. Take input str
3. Make an array of words in str
4. Run loop through array ar
5. Call void reverse(String str) for each iteration str = ar[i]
6. END

**SOURCE CODE**

import java.util.\*;

public class RecursiveSentenceRverse

{

void reverse(String str)

{

if (str.length() <= 1)

System.out.print(str);

else

{

System.out.print(str.charAt(str.length()-1));

reverse(str.substring(0,str.length()-1));

}

}

public static void main (String[] args)

{

Scanner sc = new Scanner(System.in);

RecursiveSentenceRverse ob = new RecursiveSentenceRverse();

String s = sc.nextLine();

String[] words = s.split(" ");

for (String i : words)

{

ob.reverse(i);

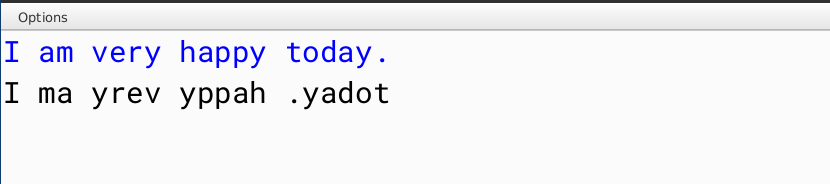
System.out.print(" ");

}

System.out.println();

}

}



|  |  |  |
| --- | --- | --- |
| Name | Data type | Description |
| s | String | User input string |
| words | String[] | Stores all the words in ‘s’ |
| ob | Object | Object of class ‘RecursiveSentenceRverse’ |
| i | String | Loop variable |

**Method :: void reverse(String str)**

**DESCRIPTION**

Reverses the given String s using recursion.

|  |  |  |
| --- | --- | --- |
| Parameter | Data type | Description |
| s | String | word whose frequency is to be found |

**QUESTION 4**

**ALGORITHM**

1. START
2. Take input n
3. Run a loop from 2 to n
4. if n % i == 0 run a while loop until n % i == 0
5. For each iteration of while loop print i and set n = n / i
6. END

**SOURCE CODE**

import java.util.\*;

public class PrimeFactorisation

{

boolean is\_prime(int n)

{

return Math.pow(2, n-1)%n == 1 || n == 2;

}

public static void main (String[] args)

{

Scanner sc = new Scanner(System.in);

PrimeFactorisation ob = new PrimeFactorisation();

System.out.print(“Enter the number :: ”);

int num = sc.nextInt();

for (int i = 2; i <= num; i++)

{

while (num % i == 0 && ob.is\_prime(i))

{

System.out.println(i);

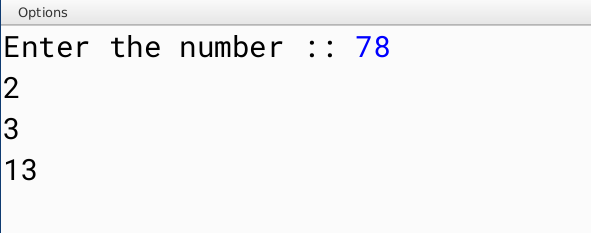
num /= i;

}

}

}

}



|  |  |  |
| --- | --- | --- |
| Name | Data type | Description |
| ob | Object | Object of class ‘PrimeFactorisation’ |
| num | int | User input for number |
| i | int | Loop variable |

**Method :: boolean is\_prime(int n)**

**DESCRIPTION**

Returns a boolean value according to prime check.

|  |  |  |
| --- | --- | --- |
| Parameter | Data type | Description |
| n | int | Number to check for prime |

**QUESTION 5**

**ALGORITHM**

1. Start
2. Take input
3. Call public void print\_fibonacci(int n)
4. For each iteration call public int get\_fibo(int n) [n = i]
5. Print fibonacci number
6. END

**SOURCE CODE**

import java.util.\*;

public class RecursiveFibonacci

{

public int get\_fibo(int n)

{

if (n <= 2)

{

return n-1;

}

else

{

return get\_fibo(n - 1) + get\_fibo(n - 2);

}

}

public void print\_fibonacci(int n)

{

for (int i = 1; i <= n; i++)

{

System.out.print(get\_fibo(i) + " ");

}

}

public static void main (String[] args)

{

Scanner sc = new Scanner(System.in);

System.out.print(“Enter number of terms :: ”);

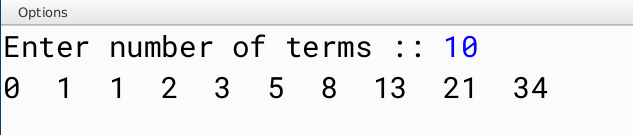
int n = sc.nextInt();

RecursiveFibonacci ob = new RecursiveFibonacci();

ob.print\_fibonacci(n);

}

}



|  |  |  |
| --- | --- | --- |
| Name | Data type | Description |
| n | int | User input for number |
| ob | Object | Object of class ‘RecursiveFibonacci’ |

**Method :: public int get\_fibo(int n)**

**DESCRIPTION**

Returns the n th fibonacci number.

|  |  |  |
| --- | --- | --- |
| Parameter | Data type | Description |
| n | int | The number whose fibonacci term is to be found |

**Method :: public void print\_fibonacci(int n)**

**DESCRIPTION**

Prints the n th fibonacci number.

|  |  |  |
| --- | --- | --- |
| Parameter | Data type | Description |
| n | int | The fibonacci term to be  printed |