**String Assignments – 3**

**Q1.** A Simple Text Formatter as in text book.

Solution:-

**import** java.io.\*;

**import** java.util.\*;

**public** **class** TextFormatter\_Q1 {

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

**int** linewidth = 72;

FileReader fr = **new** FileReader("input.txt");

BufferedReader br = **new** BufferedReader(fr);

FileWriter fw = **new** FileWriter("output.txt");

String line;

StringBuilder res = **new** StringBuilder();

res.setLength(0);

**while** ((line = br.readLine()) != **null**)

{

StringTokenizer st = **new** StringTokenizer(line);

**while** (st.hasMoreTokens())

{

String word = st.nextToken();

**if** (res.length() + word.length() > linewidth)

{

fw.write(res.toString());

res.setLength(0);

}

res.append(word).append(' ');

}

}

**if** (res.length() > 0)

{

fw.write(res.toString());

}

**else**

{

System.***out***.println();

}

fw.close();

}

}

/\* input.txt

It's a nice

day, isn't it, Mr. Mxyzzptllxy?

I think we should

go for a walk.\*/

/\* output.txt

It's a nice day, isn't it, Mr. Mxyzzptllxy? I think we should go for a walk.

\*/

**Q2.** Soundex Name Comparisons.

Solution:-

**class** SoundexName\_Q2 {

**public** **static** **final** **char**[] ***MAP*** = {

'0','1','2','3','0','1','2','0','0','2','2','4','5',

'5','0','1','2','6','2','3','0','1','0','2','0','2'

};

**public** **static** String soundex(String s)

{

String t = s.toUpperCase();

StringBuffer res = **new** StringBuffer();

**char** c, prev = '?', prevOutput = '?';

**for** (**int** i = 0; i < t.length() && res.length() < 4 && (c = t.charAt(i)) != ','; i++)

{

**if** (c >= 'A' && c <='Z' && c != prev)

{

prev = c;

**if** (i == 0)

{

res.append(c);

}

**else**

{

**char** m = ***MAP***[c - 'A'];

**if** (m != '0' && m != prevOutput)

{

res.append(m);

prevOutput = m;

}

}

}

}

**if** (res.length() == 0)

{

**return** **null**;

}

**for** (**int** i = res.length(); i < 4; i++)

{

res.append('0');

}

**return** res.toString();

}

**public** **static** **void** main(String[] args) {

String[] names = { "Darwin, Ian", "Davidson, Greg", "Darwent, William", "Derwin, Daemon"};

**for** (String name : names) {

System.***out***.println(*soundex*(name) + ' ' + name);

}

}

}

Output:-

D650 Darwin, Ian

D132 Davidson, Greg

D653 Darwent, William

D650 Derwin, Daemon

**Q3.** Write a program to enter a string and count how many vowels present in it.

Solution:-

**import** java.util.\*;

**public** **class** CountVowels\_Q3 {

**public** **int** countVowel(String str)

{

**int** count = 0;

**for** (**char** ch:str.toCharArray())

{

**if** (ch == 'A' || ch == 'a' || ch == 'E' || ch == 'e' || ch == 'I' || ch == 'i' || ch == 'O' || ch == 'o' || ch == 'U' || ch == 'u')

{

count ++;

}

}

**return** count;

}

**public** **static** **void** main(String[] args) {

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter a string");

String str = sc.nextLine();

CountVowels\_Q3 obj = **new** CountVowels\_Q3();

System.***out***.println("No of vowels = "+ obj.countVowel(str));

}

}

Output:-

Enter a string

O Jio Dhan Dhana Dhan

No of vowels = 7

**Q4.** Write a program to enter a string and count the frequency each character present in it.

Solution:-

**import** java.util.Scanner;

**public** **class** FrequencyOfEachChar\_Q4

{

**public** **static** **void** countFrequency(String st)

{

**char** ch1, ch2;

**int** count;

st = st.toLowerCase();

**for** (**int** i = 97; i <= 122; i++)

{

count = 0;

ch1 = (**char**)i;

**for** (**int** j = 0; j < st.length(); j++)

{

ch2 = st.charAt(j);

**if** (ch1 == ch2)

count = count + 1;

}

**if** (count != 0)

System.***out***.println("Frequency of "+ ch1 + " = "+ count);

}

}

**public** **static** **void** main(String[] args) {

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter a string");

String st = sc.nextLine();

*countFrequency*(st);

}

}

Output:-

Enter a string

Bandar kya jaane Adrak ka Swaad

Frequency of a = 10

Frequency of b = 1

Frequency of d = 3

Frequency of e = 1

Frequency of j = 1

Frequency of k = 3

Frequency of n = 2

Frequency of r = 2

Frequency of s = 1

Frequency of w = 1

Frequency of y = 1

**Q5.** Write a program to enter a string and print the string in a format which says that 1st letter of each word present in the string must be printed in capital letters.

Solution:-

**import** java.util.\*;

**class** FirstLetterOfEachWordCapital\_Q5 {

**public** **static** String FirstLettersCapital(String str)

{

StringTokenizer s1 = **new** StringTokenizer(str);

StringBuilder s2 = **new** StringBuilder();

**while** (s1.hasMoreTokens())

{

String word = s1.nextToken();

String fl = word.substring(0,1);

fl = fl.toUpperCase();

String rl = word.substring(1,word.length());

word = fl + rl;

s2.append(word).append(" ");

}

**return** s2.toString();

}

**public** **static** **void** main(String[] args) {

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter a string");

String str = sc.nextLine();

String new\_str = *FirstLettersCapital*(str);

System.***out***.println("Original String: "+ str);

System.***out***.println("New String: "+ new\_str);

}

}

Output:-

Enter a string

swag se karenge sabka swagat

Original String: swag se karenge sabka swagat

New String: Swag Se Karenge Sabka Swagat

**Q6.** Write a program to enter N number of strings arrange them in ascending order.

Solution:-

**import** java.util.Scanner;

**public** **class** SortStringsAscending\_Q6 {

**public** String[] input(String arr[], **int** N)

{

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter "+ N +" strings");

**for**(**int** i = 0; i < N; i++)

{

arr[i] = sc.next();

}

**return** arr;

}

**public** String[] Sort(String a[])

{

**int** len = a.length;

**for** (**int** i = 0; i < len - 1; i++)

{

**for** (**int** j = i + 1; j < len; j++)

{

**if** (a[i].compareTo(a[j]) > 0)

{

String temp = a[i];

a[i] = a[j];

a[j] = temp;

}

}

}

**return** a;

}

**public** **void** print(String s\_arr[])

{

System.***out***.println("Sorted strings:- ");

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

System.***out***.println(s\_arr[i]);

}

**public** **static** **void** main(String[] args) {

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter size of array");

**int** N = sc.nextInt();

String arr[] = **new** String[N];

String s\_arr[] = **new** String[N];

SortStringsAscending\_Q6 obj = **new** SortStringsAscending\_Q6();

arr = obj.input(arr, N);

s\_arr = obj.Sort(arr);

obj.print(s\_arr);

}

}

Output:-

Enter size of array

5

Enter 5 strings

Yudhistir

Duryodhan

Abhimanyu

Karan

Arjun

Sorted strings:-

Abhimanyu

Arjun

Duryodhan

Karan

Yudhistir

**Q7.** Write a program to enter a name as string and print the name in shorter manner.

Ex: Ram Chandra Dash will become R.C.Dash

Solution:-

**import** java.util.\*;

**public** **class** ShorterName\_Q7 {

**public** String process(String name)

{

StringTokenizer s1 = **new** StringTokenizer(name, " ");

StringBuilder s2 = **new** StringBuilder();

String word = s1.nextToken();

**while**(s1.hasMoreTokens())

{

s2.append(word.charAt(0)).append(".");

word = s1.nextToken();

}

s2.append(word);

**return** s2.toString();

}

**public** **static** **void** main(String[] args) {

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter a name");

String name = sc.nextLine();

ShorterName\_Q7 obj = **new** ShorterName\_Q7();

String s\_name = obj.process(name);

System.***out***.println("Name: "+ name);

System.***out***.println("Shorter Name: "+ s\_name);

}

}

Output:-

Enter a name

Chinnaswamy Muthuswamy Venugopal Iyer

Name: Chinnaswamy Muthuswamy Venugopal Iyer

Shorter Name: C.M.V.Iyer

**Q8.** Write a program to check whether the entered string is a palindrome or not.

Solution:-

**import** java.util.\*;

**public** **class** StringPalindrome\_Q8 {

**public** **boolean** checkPalindrome(String str)

{

**char** arr[] = **new** **char**[str.length()];

str = str.toLowerCase();

arr = str.toCharArray();

**boolean** flag = **true**;

**for**(**int** i = 0, j = str.length() - 1; i < str.length() && j >= 0; i++, j--)

{

**if**(arr[i] != arr[j])

{

flag = **false**;

**break**;

}

}

**return** flag;

}

**public** **static** **void** main(String[] args) {

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter a string");

String str = sc.nextLine();

StringPalindrome\_Q8 obj = **new** StringPalindrome\_Q8();

**boolean** k = obj.checkPalindrome(str);

**if** (k)

System.***out***.println("Palindrome");

**else**

System.***out***.println("Not Palindrome");

}

}

Output:-

Enter a string

Malayalam

Palindrome

**Q9.** Write a method that takes a delimited string to store a name and address, from which you can print a mailing label. For example, if the string contains "Sam Penn: 14 Bridge St.:Hoboken, NJ 01881," the method should print the label shown in the below:

Sam Penn

14 Bridge St.

Hoboken, NJ 01881

Solution:-

**import** java.util.\*;

**public** **class** MailingLabel\_Q9 {

**public** **void** PrintMailingLabel(String str)

{

StringTokenizer s1 = **new** StringTokenizer(str, ":");

**while**(s1.hasMoreTokens())

{

System.***out***.println(s1.nextToken());

}

}

**public** **static** **void** main(String[] args) {

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter a delimitted string containing name and address");

String str = sc.nextLine();

MailingLabel\_Q9 obj = **new** MailingLabel\_Q9();

obj.PrintMailingLabel(str);

}

}

Output:-

Enter a delimitted string containing name and address

Sam Penn:14 Bridge St.:Hoboken, NJ 01881

Sam Penn

14 Bridge St.

Hoboken, NJ 01881

**Q10.** The global replace function is a string-processing algorithm found in every word processor. Write a method that takes three String arguments: a document, a target string, and a replacement string. The method should replace every occurrence of the target string in the document with the re- placement string. For example, if the document is " To be or not to be, that is the question," and the target string is "be," , and the replacement string is "see,' the result should be, "To see or not to see, that is the question.'

Solution:-

**import** java.util.\*;

**public** **class** ReplaceTargetStrings\_Q10 {

**public** String replaceAll(String D, String T, String R)

{

StringTokenizer s1 = **new** StringTokenizer(D, ", ", **true**);

StringBuilder s2 = **new** StringBuilder();

String word;

**while**(s1.hasMoreTokens())

{

word = s1.nextToken();

**if** (word.equalsIgnoreCase(T))

s2.append(R);

**else**

s2.append(word);

}

**return** s2.toString();

}

**public** **static** **void** main(String[] args) {

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter Document");

String Document = sc.nextLine();

System.***out***.println("Enter Target String");

String Target\_String = sc.next();

System.***out***.println("Enter Replacement String");

String Replacement\_String = sc.next();

ReplaceTargetStrings\_Q10 obj = **new** ReplaceTargetStrings\_Q10();

String new\_Doc = obj.replaceAll(Document, Target\_String, Replacement\_String);

System.***out***.println("Original Document: "+ Document);

System.***out***.println("New Document: "+ new\_Doc);

}

}

Output:-

Enter Document

To be or not to be, that is the question

Enter Target String

be

Enter Replacement String

see

Original Document: To be or not to be, that is the question

New Document: To see or not to see, that is the question

**Q11.** Design and write a Java program that searches for single- digit numbers in a text and changes them to their corresponding words. For ex- ample, the string "4 score and 7 years ago" would be converted into 'four score and seven years ago

Solution:-

**import** java.util.\*;

**public** **class** DigitsToWords\_Q11 {

**public** String process(String s)

{

StringTokenizer s1 = **new** StringTokenizer(s, " ,", **true**);

StringBuilder s2 = **new** StringBuilder();

**while**(s1.hasMoreTokens())

{

String word = s1.nextToken();

**if** (word.length() == 1 && word.charAt(0) >= 48 && word.charAt(0) <= 57)

{

**char** ch = word.charAt(0);

**switch**(ch)

{

**case** '0':

s2.append("zero");

**break**;

**case** '1':

s2.append("one");

**break**;

**case** '2':

s2.append("two");

**break**;

**case** '3':

s2.append("three");

**break**;

**case** '4':

s2.append("four");

**break**;

**case** '5':

s2.append("five");

**break**;

**case** '6':

s2.append("six");

**break**;

**case** '7':

s2.append("seven");

**break**;

**case** '8':

s2.append("eight");

**break**;

**case** '9':

s2.append("nine");

**break**;

**default**:

s2.append("");

}

}

**else**

{

s2.append(word);

}

}

**return** s2.toString();

}

**public** **static** **void** main(String[] args) {

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter a string");

String str = sc.nextLine();

DigitsToWords\_Q11 obj = **new** DigitsToWords\_Q11();

String new\_str = obj.process(str);

System.***out***.println("Original String: "+ str);

System.***out***.println("New String: "+ new\_str);

}

}

Output:-

Enter a string

4 score and 7 years ago

Original String: 4 score and 7 years ago

New String: four score and seven years ago

**Q12.** Write a method that converts its String parameter so that letters are written in blocks five characters long. For example, consider the following two versions of the same sentence:

Plain : This i s how we would o r d i n a r i I y w r i t ea s e n t e n c e .

Blocked : T h i s ishowwewouldordi n a r i I ywriteasen t e n c e .

Solution:-

**import** java.util.\*;

**public** **class** StringsToBlocks5Char\_Q12 {

**public** String convertString(String str)

{

StringBuilder s1 = **new** StringBuilder();

**int** i = 0;

**char** arr[] = **new** **char**[str.length()];

arr = str.toCharArray();

**for**(**int** k = 0; k < arr.length; k++)

{

**if**(i != 5)

{

**if**(arr[k] != ' ')

{

s1.append(arr[k]);

i++;

}

}

**else**

{

s1.append(" ");

i = 0;

k--;

}

}

**return** s1.toString();

}

**public** **static** **void** main(String[] args) {

Scanner sc = **new** Scanner(System.***in***);

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

String str = sc.nextLine();

StringsToBlocks5Char\_Q12 obj = **new** StringsToBlocks5Char\_Q12();

String new\_str = obj.convertString(str);

System.***out***.println("Original String: "+ str);

System.***out***.println("Converted String: "+ new\_str);

}

}

Output:-

Enter a string:

I'm in love with the shape of you, push and pull like a magnet do

Original String: I'm in love with the shape of you, push and pull like a magnet do

Converted String: I'min lovew ithth eshap eofyo u,pus handp ullli keama gnetd o

**Q13.** Write a Java program to find longest Palindromic Substring within a string Example: The given string is: thequickbrownfoxxofnworbquickthe

The longest palindrome substring in the given string is; brownfoxxofnworb The length of the palindromic substring is: 16

Solution:-

**import** java.util.\*;

**public** **class** LongestPalindromicString\_Q13 {

**public** **boolean** checkPalindrome(String str)

{

str = str.toLowerCase();

**for**(**int** i = 0, j = str.length() - 1; i <= j; i++, j--)

{

**if**(str.charAt(i) != str.charAt(j))

{

**return** **false**;

}

}

**return** **true**;

}

**public** **int** LongestPalindrome(String str)

{

**if** (checkPalindrome(str))

{

**return** str.length();

}

**else**

{

**int** countMax = 0;

**int** Length;

String Sub\_Str = "";

**for**(**int** i = 0; i <= str.length() - 1; i++)

{

**for**(**int** j = i + 1; j <= str.length(); j++)

{

Sub\_Str = str.substring(i, j);

**if** (checkPalindrome(Sub\_Str))

{

Length = Sub\_Str.length();

countMax = (Length > countMax)?Length:countMax;

}

}

}

**return** countMax;

}

}

**public** **static** **void** main(String[] args) {

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter a string");

String S = sc.nextLine();

LongestPalindromicString\_Q13 obj = **new** LongestPalindromicString\_Q13();

**int** count = obj.LongestPalindrome(S);

System.***out***.println("Length of longest palindrome: "+ count);

}

}

Output:-

Enter a string

thequickbrownfoxxofnworbquickthe

Length of longest palindrome: 16

**Q14.** Write a Java program to find first non repeating character in a string . Example: The given string is: gibblegabbler

The first non repeated character in String is: i

Solution:-

**import** java.util.\*;

**public** **class** FirstNonRepeatChar\_Q14 {

**public** **char** checkRepetition(String str)

{

str = str.trim();

str = str.toLowerCase();

**char** arr[] = **new** **char**[26];

**for**(**char** ch: str.toCharArray())

{

**int** count = 0;

**if** (!(arr[ch - 97] > 1))

{

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

{

**if**(ch == str.charAt(i))

count++;

}

**if**(count == 1)

**return** ch;

}

}

**return** '\0';

}

**public** **static** **void** main(String[] args) {

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter a string");

String str = sc.nextLine();

FirstNonRepeatChar\_Q14 obj = **new** FirstNonRepeatChar\_Q14();

**char** c = obj.checkRepetition(str);

**if**(c != '\0')

System.***out***.println("First non repeated character is "+ c);

**else**

System.***out***.println("No non repeated character found");

}

}

Output:-

Enter a string

gibblegabbler

First non repeated character is i

**Q15.** Write a java program to remove the title from a name.

Ex. Input: Mr. Sachin Tendulkar

Ms. SainaNehwal

Output: Sachin Tendulkar

SainaNehwal

Solution:-

**import** java.util.\*;

**public** **class** RemoveTitleFromName\_Q15 {

**public** **static** String Remove(String str)

{

StringTokenizer s1 = **new** StringTokenizer(str, " ");

StringBuilder s2 = **new** StringBuilder();

s1.nextToken();

**while**(s1.hasMoreTokens())

{

s2.append(s1.nextToken()).append(" ");

}

**return** s2.toString();

}

**public** **static** **void** main(String[] args) {

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter name ");

String name = sc.nextLine();

System.***out***.println("Name with title: "+ name);

name = *Remove*(name);

System.***out***.println("Name without title: "+ name);

}

}

Output:-

Enter name

Mr. Amitabh Bachchan

Name with title: Mr. Amitabh Bachchan

Name without title: Amitabh Bachchan

**Q16.** Write a java program to read two string as user input and check if first contains second?

Solution:-

**import** java.util.\*;

**public** **class** FirstContainsSecond\_Q16 {

**public** **boolean** ContainsString(String a, String b)

{

**if**(b.length() > a.length())

**return** **false**;

**else**

{

**int** len = b.length();

**for**(**int** i = 0; i <= a.length() - b.length(); i++)

{

**if** (a.substring(i, i + b.length()).equalsIgnoreCase(b))

**return** **true**;

}

}

**return** **false**;

}

**public** **static** **void** main(String[] args) {

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter two strings");

String a = sc.nextLine();

String b = sc.nextLine();

FirstContainsSecond\_Q16 obj = **new** FirstContainsSecond\_Q16();

**boolean** k = obj.ContainsString(a, b);

**if** (k)

System.***out***.println(a +" contains "+ b);

**else**

System.***out***.println(a +" does not contain "+ b);

}

}

Output:-

Enter two strings

London Bridge is falling down

bridge is fall

Yes London Bridge is falling down contains bridge is fall

**Q17.** Write a java code to remove all occurrences of a given character from an input String

Solution:-

**import** java.util.\*;

**public** **class** RemoveCharacters\_Q17 {

**public** **static** String RemoveChar(String str, **char** ch)

{

StringBuilder s1 = **new** StringBuilder();

**for** (**char** c: str.toCharArray())

{

**if**(c != ch)

s1.append(c);

}

**return** s1.toString();

}

**public** **static** **void** main(String[] args) {

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter any String:- ");

String str = sc.nextLine();

System.***out***.println("Enter character to be removed:- ");

**char** c = sc.next().charAt(0);

System.***out***.println("Original String: "+ str);

str = *RemoveChar*(str, c);

System.***out***.println("New String: "+ str);

}

}

Output:-

Enter any String:-

Good Morning

Enter character to be removed:-

o

Original String: Good Morning

New String: Gd Mrning

**Q18.** Write a java code to find the longest substring without repeating characters in the given string.

Solution:-

**import** java.util.\*;

**public** **class** LongestSubStrNoRepetition\_Q18 {

**public** **boolean** checkRepetition(String S)

{

S = S.toLowerCase();

**int** arr[] = **new** **int**[26];

**char** ch;

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

{

ch = S.charAt(i);

arr[ch - 97]++;

}

**for** (**int** i = 0; i < 26; i++)

{

**if** (arr[i] > 1)

**return** **false**;

}

**return** **true**;

}

**public** String LongestSubstring(String S)

{

**int** countMax = 0;

String S\_Max = "";

**int** Length;

String Sub\_Str = "";

**for**(**int** i = 0; i <= S.length() - 1; i++)

{

**for**(**int** j = i + 1; j <= S.length(); j++)

{

Sub\_Str = S.substring(i, j);

**if** (checkRepetition(Sub\_Str))

{

Length = Sub\_Str.length();

**if** (Length > countMax)

{

S\_Max = Sub\_Str;

countMax = Length;

}

}

}

}

**return** S\_Max;

}

**public** **static** **void** main(String[] args) {

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter a string");

String S = sc.next();

LongestSubStrNoRepetition\_Q18 obj = **new** LongestSubStrNoRepetition\_Q18();

String str = obj.LongestSubstring(S);

System.***out***.println("Longest substring without repeating characters: "+ str);

}

}

Output:-

Enter a string

AbCabbacdB

Longest substring without repeating characters: bacd

**Q19.** Write a java program to check if a String contains only digits.

Solution:-

**import** java.util.Scanner;

**public** **class** checkStringOnlyDigits\_Q19 {

**public** **static** **boolean** check(String str)

{

**boolean** flag = **true**;

**for**(**char** ch: str.toCharArray())

{

**if**(!((ch >= 48) && (ch <= 57)))

{

flag = **false**;

**break**;

}

}

**return** flag;

}

**public** **static** **void** main(String[] args) {

Scanner sc = **new** Scanner(System.***in***);

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

String str = sc.nextLine();

**boolean** k = *check*(str);

**if** (k)

System.***out***.println("Contains only digits");

**else**

System.***out***.println("Does not contain only digits");

}

}

Output:-

Enter a string:

1234

Contains only digits

**Q20.** Write a java program to input a string from user and reverse each word of the string

Solution:-

**import** java.util.\*;

**public** **class** ReverseEachWord\_Q20 {

**public** String Reverse(String str)

{

StringTokenizer s1 = **new** StringTokenizer(str, " ");

StringBuilder s2 = **new** StringBuilder();

StringBuilder word = **new** StringBuilder();

**while**(s1.hasMoreTokens())

{

word.append(s1.nextToken());

s2.append(word.reverse()).append(" ");

word.delete(0, word.length());

}

**return** s2.toString();

}

**public** **static** **void** main(String[] args) {

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter a string");

String str = sc.nextLine();

ReverseEachWord\_Q20 obj = **new** ReverseEachWord\_Q20();

String new\_str = obj.Reverse(str);

System.***out***.println("Original String: "+ str);

System.***out***.println("New String: "+ new\_str);

}

}

Output:-

Enter a string

Pirates of the Caribbean

Original String: Pirates of the Caribbean

New String: setariP fo eht naebbiraC

**Q21.** Write a java code to capitalize first alphabet of each word in a string.

Solution:-

**import** java.util.\*;

**class** FirstLetterOfEachWordCapital\_Q5 {

**public** **static** String FirstLettersCapital(String str)

{

StringTokenizer s1 = **new** StringTokenizer(str);

StringBuilder s2 = **new** StringBuilder();

**while** (s1.hasMoreTokens())

{

String word = s1.nextToken();

String fl = word.substring(0,1);

fl = fl.toUpperCase();

String rl = word.substring(1,word.length());

word = fl + rl;

s2.append(word).append(" ");

}

**return** s2.toString();

}

**public** **static** **void** main(String[] args) {

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter a string");

String str = sc.nextLine();

String new\_str = *FirstLettersCapital*(str);

System.***out***.println("Original String: "+ str);

System.***out***.println("New String: "+ new\_str);

}

}

Output:-

Enter a string

swag se karenge sabka swagat

Original String: swag se karenge sabka swagat

New String: Swag Se Karenge Sabka Swagat

**Q22.** Write a java code to swap two string variables without using third or temp variable.

Solution:-

**import** java.util.Scanner;

**public** **class** Swap2Strings\_Q22 {

**public** **void** Swap(String a, String b)

{

System.***out***.println("Before Swap:-");

System.***out***.println("a = "+ a);

System.***out***.println("b = "+ b);

//swap operation

a = a + b;

b = a.substring(0, a.length() - b.length());

a = a.substring(b.length());

//done

System.***out***.println("After Swap:-");

System.***out***.println("a = "+ a);

System.***out***.println("b = "+ b);

}

**public** **static** **void** main(String[] args) {

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter two strings");

String a = sc.next();

String b = sc.next();

Swap2Strings\_Q22 obj = **new** Swap2Strings\_Q22();

obj.Swap(a, b);

}

}

Output:-

Enter two strings

Hattori

Ninja

Before Swap:-

a = Hattori

b = Ninja

After Swap:-

a = Ninja

b = Hattori

**Q23.** Write a java code to move all uppercase characters to the end of string.

Solution:-

**import** java.util.\*;

**public** **class** MoveUpperCaseToEnd\_Q23 {

**public** String move\_UC\_toEnd(String str)

{

StringBuilder s\_upper = **new** StringBuilder();

StringBuilder s\_other = **new** StringBuilder();

**for**(**char** ch: str.toCharArray())

{

**if** (ch >= 65 && ch <= 90)

s\_upper.append(ch);

**else**

s\_other.append(ch);

}

**return** s\_other.toString() + s\_upper.toString();

}

**public** **static** **void** main(String[] args) {

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter a string");

String str = sc.nextLine();

System.***out***.println("Orginal string: "+ str);

MoveUpperCaseToEnd\_Q23 obj = **new** MoveUpperCaseToEnd\_Q23();

str = obj.move\_UC\_toEnd(str);

System.***out***.println("New String: "+ str);

}

}

Output:-

Enter a string

Risk Hai Toh Ishq Hai

Orginal string: Risk Hai Toh Ishq Hai

New String: isk ai oh shq aiRHTIH

**Q24.** Write a java code to generate random string of a given size.

Solution:-

**import** java.util.\*;

**public** **class** GenerateRandomString\_Q24 {

**public** String RandomStr(**int** N)

{

StringBuilder s1 = **new** StringBuilder();

**for**(**int** i = 1; i <= N; i++)

{

**int** c = 1 + (**int**)(Math.*random*() \* (3 + 1 - 1));

**switch**(c)

{

**case** 1:

**int** U = 65 + (**int**)(Math.*random*() \* (90 + 1 - 65));

s1.append((**char**)U);

**break**;

**case** 2:

**int** L = 97 + (**int**)(Math.*random*() \* (122 + 1 - 97));

s1.append((**char**)L);

**break**;

**case** 3:

**int** D = 48 + (**int**)(Math.*random*() \* (57 + 1 - 48));

s1.append((**char**)D);

**break**;

**default**:

s1.append("");

}

}

**return** s1.toString();

}

**public** **static** **void** main(String[] args) {

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter size of random string");

**int** N = sc.nextInt();

GenerateRandomString\_Q24 obj = **new** GenerateRandomString\_Q24();

String R\_Str = obj.RandomStr(N);

System.***out***.println("Random String: "+ R\_Str);

}

}

Output:-

Enter size of random string

10

Random String: y1FGNVe0fP

**Q25.** Write an program to print all permutations of a given String in Java. For example, if given input is "123" then your program should print all 6 permutations e.g.

123 132" "213" "231 " 312" and •'321 '

Solution:-

**import** java.util.\*;

**public** **class** AllPermutationsOfString\_Q25 {

**public** **void** printPermutations(String str, String p)

{

**if** (str.length() == 0)

{

System.***out***.println(p);

**return**;

}

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

{

**char** ch = str.charAt(i);

String r = str.substring(0, i) + str.substring(i + 1, str.length());

printPermutations(r, p + ch);

}

}

**public** **static** **void** main(String[] args) {

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter a string");

String S = sc.next();

AllPermutationsOfString\_Q25 obj = **new** AllPermutationsOfString\_Q25();

obj.printPermutations(S, "");

}

}

Output:-

Enter a string

abc

abc acb bac bca cab cba

**Q26.** Write a Java program which will take a String input and print out a number of vowels and consonants on that String. For example, if the input is "Java" then your program should print "2 vowels and 2 consonants .

Solution:-

**import** java.util.\*;

**public** **class** CountVowelsConsonants\_Q26 {

**public** **void** count\_V\_C(String str)

{

str = str.trim();

str = str.toLowerCase();

**int** cv = 0, cc = 0;

**for**(**char** ch:str.toCharArray())

{

**if** (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u')

{

cv++;

}

**else** **if** (ch >= 97 && ch <= 122)

{

cc++;

}

}

System.***out***.println(cv +" Vowels and "+ cc +" Consonants");

}

**public** **static** **void** main(String[] args) {

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter a string");

String str = sc.nextLine();

CountVowelsConsonants\_Q26 obj = **new** CountVowelsConsonants\_Q26();

obj.count\_V\_C(str);

}

}

Output:-

Enter a string

the quick brown fox jumps over the lazy dog

11 Vowels and 24 Consonants

**Q27.** You are given 3 strings:first, second, and third. Third String is said to be a shuffle of first and second if it can be formed by interleaving the characters of first and second String in a way that maintains the left to right ordering of the characters from each string.

For example, given first “abc" and second “def", third “dabecf" is a valid shuffe since it preserves the character ordering of the two strings. So, given these 3 strings write a function that detects whether the third String is a valid shuffe of first and second String.

Solution:-

**import** java.util.\*;

**public** **class** ValidShuffle\_Q27 {

**public** **boolean** checkValid\_Shuffle(String a, String b, String c)

{

**if** (c.length() != (a.length() + b.length()))

{

**return** **false**;

}

**else**

{

**int** i = 0, j = 0;

**char** ch;

**for** (**int** k = 0; k < c.length(); k++)

{

ch = c.charAt(k);

**if** ((i < a.length()) && (ch == a.charAt(i)))

i++;

**else** **if** ((j < b.length()) && (ch == b.charAt(j)))

j++;

}

**if** ((i == a.length()) && (j == b.length()))

**return** **true**;

**else**

**return** **false**;

}

}

**public** **static** **void** main(String[] args) {

String a = "abc";

String b = "def";

String c = "dabecf";

System.***out***.println("First string: "+a+"\nSecond String: "+b+"\nThird String: "+c);

ValidShuffle\_Q27 obj = **new** ValidShuffle\_Q27();

**boolean** res = obj.checkValid\_Shuffle(a, b, c);

**if** (res)

System.***out***.println("Third String is a valid shuffle of first and second string");

**else**

System.***out***.println("Third String is NOT a valid shuffle of first and second string");

}

}

Output:-

First string: abc

Second String: def

Third String: dabecf

Third String is a valid shuffle of first and second string

**Q28.** Write a function to search for the existence of a string (target) in another string source . The function takes two strings as the input and returns the index where the second string is found. If the target string cannot be found, then return -1. If you are a Java developer, then you can relate its behavior to indexOf() method from java.lang.String class. This question is also asked as Code and algorithm to check if a given short string is a substring of the main string. Can you get a linear solution (O(n)) if possible?

Solution:-

**import** java.util.\*;

**public** **class** ExistenceOfString\_Q28 {

**public** **int** checkExistence(String Target, String Source)

{

String str = "";

**if** (Target.length() > Source.length())

{

System.***out***.println("Target is larger than Source");

**return** -1;

}

**else**

{

**int** diff = Source.length() - Target.length();

**for** (**int** i = 0; i <= diff; i++)

{

str = Source.substring(i, i + Target.length());

**if** (str.equalsIgnoreCase(Target))

**return** i;

}

}

**return** -1;

}

**public** **static** **void** main(String[] args) {

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter source string");

String S = sc.nextLine();

System.***out***.println("Enter target string");

String T = sc.nextLine();

ExistenceOfString\_Q28 obj = **new** ExistenceOfString\_Q28();

**int** Index = obj.checkExistence(T, S);

**if** (Index == -1)

System.***out***.println("Not found");

**else**

System.***out***.println("Found at "+Index);

}

}

Output:-

Enter source string

knowledge

Enter target string

led

Found at 4

**Q29.** Write a function to implement an algorithm which will accept a string of characters and should find the highest occurrence of the character and display it. For example if input is "aaaaaaaaaaaaaaaaabbbbcddddeeeeee" it should return "a .

Solution:-

**import** java.util.\*;

**public** **class** HighestOccurenceOfCharacter\_Q29 {

**public** **char** countOccurence(String str)

{

**int** arr[] = **new** **int**[26];

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

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

str = str.toLowerCase();

**for**(**char** ch:str.toCharArray())

{

arr[ch - 97]++;

}

**int** max = -1;

**int** maxIndex = -1;

**for** (**int** i = 0; i < 26; i++)

{

**if** (arr[i]!=0)

{

**if** (arr[i] > max)

{

max = arr[i];

maxIndex = i;

}

}

}

**return** (**char**)(97 + maxIndex);

}

**public** **static** **void** main(String[] args) {

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter a string");

String str = sc.nextLine();

HighestOccurenceOfCharacter\_Q29 obj = **new** HighestOccurenceOfCharacter\_Q29();

**char** maxChar = obj.countOccurence(str);

System.***out***.println("Entered string: "+ str);

System.***out***.println("Highest Occurring Character is "+ maxChar);

}

}

Output:-

Enter a string

zindagi na milegi dobaara

Entered string: zindagi na milegi dobaara

Highest Occurring Character is a

**Q30.** Write a Program to sort String on their length in Java? Your method should accept an array of String and return a sorted array based upon the length of String.

Solution:-

**import** java.util.\*;

**public** **class** SortArrayOfStrings\_Q30 {

**public** String[] Input(**int** N)

{

Scanner sc = **new** Scanner(System.***in***);

String arr[] = **new** String[N];

System.***out***.println("Enter "+ N +" strings:");

**for**(**int** i = 0; i < N; i++)

{

arr[i] = sc.nextLine();

}

**return** arr;

}

**public** String[] SortStrings(String arr[])

{

String temp = "";

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

{

**for**(**int** j = 0; j < arr.length - 1; j++)

{

**if** (arr[j].length() > arr[j + 1].length())

{

temp = arr[j];

arr[j] = arr[j + 1];

arr[j + 1] = temp;

}

}

}

**return** arr;

}

**public** **void** Print(String a[])

{

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

{

System.***out***.println(a[i]);

}

}

**public** **static** **void** main(String[] args) {

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter size of array:");

**int** N = sc.nextInt();

SortArrayOfStrings\_Q30 obj = **new** SortArrayOfStrings\_Q30();

String arr1[] = obj.Input(N);

System.***out***.println("Original Array of strings:-");

obj.Print(arr1);

String arr2[] = obj.SortStrings(arr1);

System.***out***.println();

System.***out***.println("Sorted Array of strings:-");

obj.Print(arr2);

}

}

Output:-

Enter size of array:

6

Enter 6 strings:

Joey Tribbiani

Monica Geller

Chandler Bing

Rachel Green

Ross Geller

Phoebe Buffay

Original Array of strings:-

Joey Tribbiani

Monica Geller

Chandler Bing

Rachel Green

Ross Geller

Phoebe Buffay

Sorted Array of strings:-

Ross Geller

Rachel Green

Monica Geller

Chandler Bing

Phoebe Buffay

Joey Tribbiani

**Ch-4 Pattern Matching with Regular Expression**

**Q1.** Write a program in Java to remove whitespaces from a string.

Hint: Use replaceAll() method of Matcher class.

Solution:-

**import** java.util.regex.\*;

**import** java.util.Scanner;

**public** **class** RemoveWhiteSpaces\_Q1 {

**public** **static** **void** main(String[] args) {

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter a string");

String str = sc.nextLine();

String patt = " ";

Pattern p = Pattern.*compile*(patt);

Matcher m = p.matcher(str);

String new\_str = m.replaceAll("");

System.***out***.println("Original string: "+ str);

System.***out***.println("New String: "+ new\_str);

m.reset();

}

}

Output:-

Enter a string

The quick brown fox jumps over the lazy dog

Original string: The quick brown fox jumps over the lazy dog

New String: Thequickbrownfoxjumpsoverthelazydog

**Q2.** Write a Java program to read all mobile numbers present in given file and validate it on below criteria:

-The first digit should contain number between 7 to 9.

-The rest 9 digit can contain any number between 0 to 9.

-The mobile number can have 11 digits also by including 0 at the beginning.

-The mobile number can be of 12 digits by including 91 at the beginning.

The number which satisfies the above criteria is a valid mobile Number.

Solution:-

**import** java.util.regex.\*;

**import** java.io.\*;

**public** **class** MatchMobileNumbers\_Q2 {

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

FileReader fr = **new** FileReader("Mobile\_Numbers\_Q2.txt");

BufferedReader br = **new** BufferedReader(fr);

String patt = "(0|(91))?[7-9][0-9]{9}";

Matcher m = Pattern.*compile*(patt).matcher("");

String line = "";

System.***out***.println("Valid Mobile Numbers:-");

**while** ((line = br.readLine()) != **null**)

{

m.reset(line);

**if**(m.matches())

{

System.***out***.println(line);

}

}

}

}

/\*

Mobile\_Numbers\_Q2.txt

1234567890

7623456832

0912235679010

07693421871

913452678321

073476543889

917087436521

\*/

Output:-

Valid Mobile Numbers:-

7623456832

07693421871

917087436521

**Q3.** Write a program to Check if given email or URL (both) addresses are valid or not.

mail validation

--------------------Example: rama@gmail.com

1. name: [a-zA-z\_-]+
2. @ : @
3. subdomain: [a-zA-Z]{2,256} (Ex: gmail, yahoo, etc) 4) dot (.): \.
4. domain: [a-zA-Z]{2,5} (Ex: in, com etc)

url validation --------------

Example: https://www.gmail.com

1. URL must start with either http or https : https?
2. followed by :// : ://
3. then it must contain www. : w{3}\.
4. subdomain: [a-zA-Z]{2,256} (Ex: gmail, yahoo, etc) 4) dot (.): \.
5. domain: [a-zA-Z]{2,5} (Ex: in, com etc)

Hint: You are free to add some additional restrictions to the email and URL.

The pattern must satisfy both email and URL.

Solution:-

**import** java.util.regex.\*;

**import** java.util.Scanner;

**public** **class** ValidEmailOrURL\_Q3 {

**public** **static** **void** main(String[] args) {

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter an email or url");

String str = sc.nextLine();

String patt = "(^((http)[s]?://w{3}\\.)|([a-zA-Z0-9\_-]{2,256}@))[a-zA-Z]{2,256}\\.[a-zA-Z]{2,5}";

Matcher m = Pattern.*compile*(patt).matcher(str);

**if** (m.matches())

System.***out***.println("Valid");

**else**

System.***out***.println("Invalid");

}

}

Output:-

Enter an email or url

http://www.google.com

Valid

**Q4.** Write a program to extract maximum numeric value from a given string.

**Example**

Input: There is 60 students in cse-d section, 40 in cse-b, and 55 in cse-a Output:Max: 60.

Solution:-

**import** java.util.regex.\*;

**import** java.util.Scanner;

**public** **class** MaxNumValueFromString\_Q4 {

**public** **int** findMaxNumValue(String S)

{

**int** N;

**int** Max = Integer.***MIN\_VALUE***;

String str;

Matcher m = Pattern.*compile*("\\d+").matcher(S);

**while** (m.find())

{

str = S.substring(m.start(), m.end());

N = Integer.*parseInt*(str);

Max = (N > Max)?N:Max;

}

**return** Max;

}

**public** **static** **void** main(String[] args) {

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter a string");

String S = sc.nextLine();

MaxNumValueFromString\_Q4 obj = **new** MaxNumValueFromString\_Q4();

**int** MaxValue = obj.findMaxNumValue(S);

System.***out***.println("String: "+ S);

System.***out***.println("Max: "+ MaxValue);

}

}

Output:-

Enter a string

Dhoom 2, 3 Idiots, Scam 1992, Around the World in 80 days

String: Dhoom 2, 3 Idiots, Scam 1992, Around the World in 80 days

Max: 1992

**Q5.** Write a program to demonstrate the working of splitting a text by a given pattern. The given input is “CSE1ECE2EEE3CIVIL”.The output of the program is look like below:

CSE

ECE

EEE

CIVIL

Use the split () and case controlling flags to solve this.

Solution:-

**import** java.util.regex.\*;

**import** java.util.Scanner;

**public** **class** SplitTextUsingPattern\_Q5 {

**public** **static** **void** main(String[] args) {

Scanner sc = **new** Scanner(System.***in***);

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

String str = sc.nextLine();

String patt = "\\d";

System.***out***.println("String: "+ str);

System.***out***.println("Pattern: "+ patt);

Pattern p = Pattern.*compile*(patt, Pattern.***CASE\_INSENSITIVE***);

String S[] = p.split(str);

**for**(String s1: S)

{

System.***out***.println(s1);

}

}

}

Output:-

Enter a string:

CSE1ECE2EEE3CIVIL

String: CSE1ECE2EEE3CIVIL

Pattern: \d

CSE

ECE

EEE

CIVIL

**Q6.** Write a program to get the first letter of each word in a string using regex in Java. For example: the input string is “This is CSE Students” and output of the program is: TiCS.

Solution:-

**import** java.util.regex.\*;

**import** java.util.Scanner;

**public** **class** PrintFirstCharUsingRegex\_Q6 {

**public** String FirstCharsOfString(String S)

{

String str = "";

String arr[] = Pattern.*compile*(" ").split(S);

**for**(String s1: arr)

{

Matcher m = Pattern.*compile*("\\w").matcher(s1);

**if** (m.find())

str = str + s1.substring(m.start(), m.end());

}

**return** str;

}

**public** **static** **void** main(String[] args) {

Scanner sc = **new** Scanner(System.***in***);

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

String str = sc.nextLine();

PrintFirstCharUsingRegex\_Q6 obj = **new** PrintFirstCharUsingRegex\_Q6();

String new\_str = obj.FirstCharsOfString(str);

System.***out***.println("Original String: "+ str);

System.***out***.println("New String: "+ new\_str);

}

}

Output:-

Enter a string:

Laugh out Loud

Original String: Laugh out Loud

New String: LoL

**Q7.** Write a program to demonstrate the differences of various quantifiers e.g. Greedy Quantifiers VS Reluctant Quantifiers VS Possessive Quantifiers.

Solution:-

**import** java.util.regex.\*;

**public** **class** QuantifierTypes\_Q7 {

**public** **static** **void** main(String[] args) {

//Greedy Quantifier

String P1 = "g+";

String S1 = "gggsss";

Matcher m1 = Pattern.*compile*(P1).matcher(S1);

**while**(m1.find())

System.***out***.println("Pattern found from "+ m1.start() +" to "+ (m1.end() - 1));

//Reluctant Quantifier

String P2 = "g+?";

String S2 = "gggsss";

Matcher m2 = Pattern.*compile*(P2).matcher(S2);

**while**(m2.find())

{

System.***out***.println("Pattern found from "+ m2.start() +" to "+ (m2.end() - 1));

}

//Possessive Quantifier

String P = ".\*+foo";

String S = "afooaaaafoo";

Matcher m = Pattern.*compile*(P).matcher(S);

**if** (m.find())

System.***out***.println("Pattern found from "+ m.start() +" to "+ (m.end() - 1));

**else**

System.***out***.println("Pattern not found");

}

}

Output:-

Pattern found from 0 to 2

Pattern found from 0 to 0

Pattern found from 1 to 1

Pattern found from 2 to 2

Pattern not found