Searching in array is technique to find the specific defined element in an array. For this first declare the array of String then enter the size of array and then enter all the elements of that array in by user. Now enter the element you want to search for. If searching element is found return index number of element otherwise return -1.

static int binSearch(String[] arr, String s)

{

int i = 0, r = arr.length - 1;

while (i <= r) {

int m = i + (r - i) / 2;

int res = s.compareTo(arr[m]);

if (res == 0)

return m;

if (res > 0)

i = m + 1;

else

r = m - 1;

}

return -1;

}

public static void main(String []args)

{

Scanner sc = new Scanner(System.in);

int size=sc.nextInt();

String[] arr = new String[size];

//System.out.println("Enter string array");

for(int j=0;j<size;j++)

{

arr[j]=sc.next();

}

// System.out.println("Enter searching Element");

String s = sc.next();

int result = binSearch(arr, s);

if (result == -1)

System.out.println("Element is not present");

else

System.out.println("Element is found at index " + result);

}

**Q.** Mr. Chandan loves to play with strings. Today he is solving a problem in which he needs to replace a substring with a space and display the output.Help him complete the code.

static String replace\_String(String Str, String Sub)

{

Str = Str.replaceAll(Sub, " ");

Str = Str.trim();

return Str;

}

3. static char getCharFromString(String str, int index)

{

return str.charAt(index);

}

public static void main(String[] args)

{

Scanner s=new Scanner(System.in);

String str =s.nextLine();

int index= s.nextInt();

4. // Rectangle class parameterized constructor

public Rectangle(double width, double height)

{

// calling Shape class constructor

super(width, height);

}

final double getArea()

{

return this.getHeight() \* this.getWidth();

}

}

//derived class two

class Square extends Shape

{

// Rectangle class parameterized constructor

public Square(double side)

{

// calling Shape class constructor

super(side, side);

}

final double getArea()

{

return this.getHeight() \* this.getWidth();

}

}

5. import java.util.Scanner;

public class file {

static int countOfSbStrng(String st, int l, int m)

{

int count = 0;

for (int i = 0; i < l; i++)

{

int n = 0;

for (int j = i; j < l; j++)

{

n = n \* 10 + (st.charAt(j) - '0');

if (n % m == 0)

count++;

}

} return count;

}

public static void main(String []args)

{ Scanner obj = new Scanner(System.in);

// System.out.println("Enter string Interger");

String st = obj.nextLine();

int l = st.length();

int m = obj.nextInt();

System.out.println(countOfSbStrng(st, l, m));

}

}

6. //package first;

import java.util.\*;

public class file {

static int countSbStringSameEnd(String st)

{

int res = 0;

int len = st.length();

for (int i = 0; i<len; i++)

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

if (CheckSameStr(st.substring(i, i + j)))

{

res++;

System.out.println(st.substring(i, i + j));

}

return res;

}

static boolean CheckSameStr(String st)

{

return (st.charAt(0) == st.charAt(st.length() - 1));

}

public static void main(String args[])

{

Scanner sc = new Scanner(System.in);

int n=sc.nextInt();

String strng = "";

for(int i=0;i<n;i++)

{

strng = sc.next();

System.out.println(countSbStringSameEnd(strng));

}

}

}

7. static String move(String str)

{

int len = str.length();

String regx = "[a-zA-Z0-9\\s+]";

String res1 = "", res2 = "";

for (int i = 0; i < len; i++) {

char c = str.charAt(i);

if (String.valueOf(c).matches(regx))

res1 = res1 + c;

else

res2 = res2 + c;

}

return res1 + res2;

}

8. public class file {

// Function to print all substring

public static void SubString(String str, int n)

{

for (int i = 0; i < n; i++)

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

System.out.println(str.substring(i, j));

}

public static void main(String[] args)

{

Scanner in = new Scanner(System.in);

String str = in.nextLine();

SubString(str, str.length());

}

}

9.

import java.util.Scanner;

import java.util.regex.Matcher;

import java.util.regex.Pattern;

public class file {

public static void main(String[] args) {

String regex = "(\\b\\w+\\b)(\\s+\\1\\b)+";

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

Scanner in = new Scanner(System.in);

int numSentences = Integer.parseInt(in.nextLine());

while (numSentences-- > 0) {

String input = in.nextLine();

Matcher m = p.matcher(input);

// Check for subsequences of input that match the compiled pattern

while (m.find()) {

input = input.replaceAll(m.group(), m.group(1));

}

// Prints the modified sentence.

System.out.println(input);

}

in.close();

}

}

10.

import java.util.Scanner;

public class file {

public static void main(String[] args) {

/\* Read input \*/

Scanner scan = new Scanner(System.in);

String s = scan.nextLine();

scan.close();

s = s.trim(); // so that .split() works properly

/\* Check special cases \*/

if (s.length() == 0) {

System.out.println(0);

return;

}

/\* Split on all non-alphabetic characters \*/

String [] words = s.split("[^a-zA-Z]+");

/\* Print output \*/

System.out.println(words.length);

for (String word : words) {

System.out.println(word);

}

}

}

11.

import java.util.ArrayList;

import java.util.List;

import java.util.Scanner;

import java.util.Stack;

import java.util.regex.Matcher;

import java.util.regex.Pattern;

public class file {

public static void main(String[] args) {

Pattern pattern = Pattern.compile("(<|</)[^<>/]+>");

Scanner sc = new Scanner(System.in);

int N = sc.nextInt();

sc.nextLine();

for (int tc = 0; tc < N; tc++) {

String line = sc.nextLine();

Matcher matcher = pattern.matcher(line);

int index = 0;

List<String> contents = new ArrayList<String>();

Stack<String> tagStack = new Stack<String>();

String content = null;

boolean lastStartOrEnd = false;

while (matcher.find()) {

String tag = matcher.group();

boolean startOrEnd;

String tagName;

if (tag.startsWith("</")) {

startOrEnd = false;

tagName = tag.substring(2, tag.length() - 1);

} else {

startOrEnd = true;

tagName = tag.substring(1, tag.length() - 1);

}

if (startOrEnd) {

tagStack.push(tagName);

} else {

if (!tagStack.empty() && tagStack.pop().equals(tagName)) {

if (lastStartOrEnd) {

content = line.substring(index, matcher.start());

if (!content.isEmpty()) {

contents.add(content);

}

}

}

}

index = matcher.end();

lastStartOrEnd = startOrEnd;

}

System.out.println(contents.isEmpty() ? "None" : String.join("\n", contents));

}

sc.close();

}

}

12.

**import** java.util.regex.Matcher;

**import** java.util.regex.Pattern;

**import** java.util.Scanner;

**public** **class** IPAddress {

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

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

**while**(in.hasNext()) {

String IP = in.next();

System.***out***.println(IP.matches(**new** MyRegex().pattern));

}

in.close();

}

}

**class** MyRegex {

String num = "([01]?\\d{1,2}|2[0-4]\\d|25[0-5])";

String pattern = num + "." + num + "." + num + "." + num;

}

13.

int length;

int breadth;

Room(int length, int breadth)

{

this.length=length;

this.breadth=breadth;

}

int area()

{

return length\*breadth;

}

}

class Turing extends Room

{

int height;

Turing(int x, int y, int z)

{

super(x,y);

this.height=z;

}

int area()

{

return length\*breadth\*height;

}

}

class Newton extends Room

{

int height;

Newton(int x, int y, int z)

{

super(x,y);

this.height=z;

}

int area()

{

return length\*breadth\*height;

}

}

14.

static boolean is\_vow(char c)

{

return (c == 'a') || (c == 'e') ||

(c == 'i') || (c == 'o') ||

(c == 'u');

}

static void removeVowels(String str)

{

System.out.print(str.charAt(0));

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

if ((!is\_vow(str.charAt(i - 1))) || (!is\_vow(str.charAt(i))))

System.out.print(str.charAt(i));

}

15.

int ans = 0;

for (int i = 0; i < s1.length(); i++)

{

String s3 = "";

char[] s4 = s1.toCharArray();

for (int j = i; j < s1.length(); j++)

{

s3 += s4[j];

if (s2.indexOf(s3) != -1)

ans++;

}

}

return ans;

}