SET-1

2.Find the day(Friday) of a date input given as MM-dd-yyyy (format)

input:12-27-2012

output:Thursday

package Set1;

import java.util.\*;

import java.text.\*;

public class ClassSet2 {

public static String getDay(Date d1){

String s1;

SimpleDateFormat sdf=new SimpleDateFormat("EEEEE");

s1=sdf.format(d1);

return s1;

}

public static void main(String[] args) {

Date d1=new Date(2012/12/27);

System.out.println("day is:"+getDay(d1));

}

}

---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

8. Given date in dd-MM-yyyy format.return the month in full name format(January)

input:"23-01-2012"

output:January

package Set1;

import java.text.SimpleDateFormat;

import java.util.\*;

public class ClassSet8 {

public static String monthDiff(Date d1){

SimpleDateFormat sdf=new SimpleDateFormat("MMMM");

String s=(sdf.format(d1));

return s;

}

public static void main(String[] args) {

Date d1=new Date(23/01/2012);

System.out.println(monthDiff(d1));

}

}

----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

9. Two dates are given as input in "yyyy-MM-dd" format. Find the number of months between the two dates

input1:"2012-12-01"

input2:"2012-01-03"

output:11

package Set1;

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Calendar;

import java.util.Date;

public class ClassSet9 {

public static int monthDiff(String s1,String s2) throws ParseException{

SimpleDateFormat sdf=new SimpleDateFormat("yyyy-MM-dd");

Calendar c=Calendar.getInstance();

Date d1=sdf.parse(s1);

Date d2=sdf.parse(s2);

c.setTime(d1);

int m1=c.get(Calendar.MONTH);

int y1=c.get(Calendar.YEAR);

c.setTime(d2);

int m2=c.get(Calendar.MONTH);

int y2=c.get(Calendar.YEAR);

int n=(y1-y2)\*12+(m1-m2);

return n;

}

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

String s1=new String("2013-12-01");

String s2=new String("2012-01-03");

System.out.println(monthDiff(s1,s2));

}

}

---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

19.accept a string and find if it is of date format "dd/mm/yyyy".

input:01/13/2012

output:false

package Set1;

import java.util.StringTokenizer;

public class ClassSet19 {

public static boolean formattingDate(String s){

boolean b1=false;

StringTokenizer t=new StringTokenizer(s,"/");

int n1=Integer.parseInt(t.nextToken());

int n2=Integer.parseInt(t.nextToken());

String s2=t.nextToken();

int n3=Integer.parseInt(s2);

int n4=s2.length();

if(n4==4)

{

if(n3%4==0)

{

if((n2==2)&&(n1<=29))

b1=true;

else if((n2==4)||(n2==6)||(n2==9)||(n2==11)&&(n1<=30))

b1=true;

else if((n2==1)||(n2==3)||(n2==5)||(n2==7)||(n2==8)||(n2==10)||(n2==12)&&(n1<=31))

b1=true;

}

else

{

if((n2==2)&&(n1<=28))

b1=true;

else if((n2==4)||(n2==6)||(n2==9)||(n2==11)&&(n1<=30))

b1=true;

else if((n2==1)||(n2==3)||(n2==5)||(n2==7)||(n2==8)||(n2==10)||(n2==12)&&(n1<=31))

b1=true;

}

}

return b1;

}

public static void main(String[] args) {

String s="31/5/2012";

boolean b=formattingDate(s);

if(b==true)

System.out.println("date is in dd/MM/yyyy format");

else

System.out.println("date is not in dd/MM/yyyy format");

}

}

--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

30. Find the number of days between two input dates.

package Set1;

import java.text.\*;

import java.util.\*;

public class ClassSet30 {

public static int dateDifference(String s1,String s2) throws ParseException{

SimpleDateFormat sd=new SimpleDateFormat("dd/MM/yyyy");

Date d=sd.parse(s1);

Calendar c=Calendar.getInstance();

c.setTime(d);

long d1=c.getTimeInMillis();

d=sd.parse(s2);

c.setTime(d);

long d2=c.getTimeInMillis();

int n=Math.abs((int) ((d1-d2)/(1000\*3600\*24)));

return n;

}

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

String s1="27/12/2009";

String s2="15/09/2012";

System.out.println(dateDifference(s1,s2));

}

}

--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

SET-2

---------------------------------------------------------------------------------------------------------

15)What will be the DAY of current date in next year

package Set2;

import java.text.SimpleDateFormat;

import java.util.\*;

public class ClassSet15 {

public static void main(String[] args) {

Date d=new Date();

System.out.println(sameDayOnUpcomingYear(d));

}

public static String sameDayOnUpcomingYear(Date d) {

Date d1=new Date();

d1.setYear(d.getYear()+1);

SimpleDateFormat sdf=new SimpleDateFormat("EEEE");

String s=sdf.format(d1);

return s;

}

}

----------------------------------------------------------------------------------------------------------||

22)no of days in a month in specific year

package Set2;

import java.util.\*;

public class ClassSET22 {

public static void main(String[] args){

Calendar ca=new GregorianCalendar(2013,Calendar.FEBRUARY,03);

System.out.println(noOfDaysInaMonth(ca));

}

public static int noOfDaysInaMonth(Calendar ca){

int n=ca.getActualMaximum(Calendar.DAY\_OF\_MONTH);

return n;

}

}

----------------------------------------------------------------------------------------------------------||

SET-3

48) leap year or not using API?

package Set3;

import java.util.\*;

public class ClassSeT48 {

public static void main(String[] args) {

String s="2013";

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

}

public static boolean leapYear(String s) {

int n=Integer.parseInt(s);

GregorianCalendar c=new GregorianCalendar();

boolean b=c.isLeapYear(n);

return b;

}

}

----------------------------------------------------------------------------------------------------||

SILK

1) Given a method with a date string in format dd/mm/yyyy.

Write code to validate the given date against the given formate.

if it is a valid date return 1 else return -1.

Example1:

Input= 12/06/1987

output=1

Example1:

Input= 03/1/1987

output=-1

package com.Packas;

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Date;

public class Dd {

public static void main(String[] args) {

String s1="29/02/2006";

getvalues(s1);

}

public static void getvalues(String s1) {

if(s1.matches("[0-9]{2}[/]{1}[0-9]{2}[/]{1}[0-9]{4}"))

{

SimpleDateFormat sdf=new SimpleDateFormat("dd/MM/yyyy");

sdf.setLenient(false);

try {

Date d1=sdf.parse(s1);

System.out.println(1);

} catch (ParseException e) {

System.out.println(-1);

}

}

else

System.out.println(-1);

}

.............................................................................

14) Given a method with a date string as input.write code to validate if the given date is in any of the following formats:

dd.mm.yyyy

dd/mm/yy

dd-mm-yyyy

return 1 if it is a valid date else return-1.

Example:

input=12.03.2012

output=1

import java.util.ArrayList;

import java.util.Collections;

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Date;

public class Dd {

public static void main(String[] args) {

String s="14/09/91";

getvalues(s);

}

public static void getvalues(String s) {

if(s.matches("[0-9]{2}[.]{1}[0-9]{2}[.]{1}[0-9]{4}"))

{

SimpleDateFormat sdf=new SimpleDateFormat("dd.MM.yyyy");

sdf.setLenient(false);

try

{

Date d1=sdf.parse(s);

System.out.println(1);

} catch (ParseException e) {

System.out.println(-1);

}

}

else if(s.matches("[0-9]{2}[/]{1}[0-9]{2}[/][0-9]{2}"))

{

SimpleDateFormat sdf=new SimpleDateFormat("dd/MM/yyyy");

sdf.setLenient(false);

try

{

Date d1=sdf.parse(s);

System.out.println(1);

} catch (ParseException e) {

System.out.println(-1);

}

}

else if(s.matches("[0-9]{2}[-]{1}[0-9]{2}[-][0-9]{4}"))

{

SimpleDateFormat sdf=new SimpleDateFormat("dd-MM-yyyy");

sdf.setLenient(false);

try

{

Date d1=sdf.parse(s);

System.out.println(1);

} catch (ParseException e) {

System.out.println(-1);

}

}

else

System.out.println(-1);

}

}

......................................................................

17) Given a method with a date string in dd/mm/yyy format.

Write code to calculate the day which falls on the same date next year and return it as outputNote-return the outout in small case.

Example.

input=13/07/2012

output=saturday

package com.Packas;

import java.util.ArrayList;import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Calendar;

import java.util.Date;

public class Dd {

public static void main(String[] args) {

String s1="13/07/2012";

getvalues(s1);

}

public static void getvalues(String s1) {

SimpleDateFormat sdf=new SimpleDateFormat("dd/MM/yyyy");

sdf.setLenient(false);

try {

Date d1=sdf.parse(s1);

Calendar cal=Calendar.getInstance();

cal.setTime(d1);

cal.add(Calendar.YEAR, 1);

Date d2=cal.getTime();

SimpleDateFormat sdf1=new SimpleDateFormat("EEEEE");

String s=sdf1.format(d2);

System.out.println(s);

} catch (ParseException e) { e.printStackTrace();

}

}

}

...........................................................................

18) Given a method with a date string in dd/mm/yyy.Write code to convert the given date to the format dd-mm-yy.

Example:

input=12/11/1998

output=12-11-98

package com.Packas;

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Date;

public class Dd {

public static void main(String[] args) {

String s1="23/5/2014";

getvalues(s1);

}

public static void getvalues(String s1) {

SimpleDateFormat sdf=new SimpleDateFormat("dd/MM/yyyy");

sdf.setLenient(false);

try {

Date d1=sdf.parse(s1);

SimpleDateFormat sdf1=new SimpleDateFormat("dd-MM-yy");

String s2=sdf1.format(d1);

System.out.println(s2);

} catch (ParseException e) {

System.out.println(-1);

}

}

}

......................................................................

................................................................

59) Given a method with a date string in the format dd/mm/yyyy.

Write code to convert the given date to the format dd-mm-yy.

Example:

input=12/11/1998

output= 12-11-98

package shivam;

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Date;

public class B {

public static void main(String[] args) {

String s1="12/12/1998";

getvalues(s1);

}

public static void getvalues(String s1) {

SimpleDateFormat sdf=new SimpleDateFormat("dd/MM/yyyy");

sdf.setLenient(false);

try {

Date d1=sdf.parse(s1);

SimpleDateFormat sdf1=new SimpleDateFormat("dd-MM-yy");

String s2=sdf1.format(d1);

System.out.println(s2);

} catch (ParseException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}

........................................................................................

44. Return the number of days in a month, where month and year are given as input.

package Set1;

import java.util.\*;

public class ClassSet44 {

public static void main(String[] args){

String s1="02/2011";

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

}

public static int noOfDaysInAmonth(String s1){

int n=0;

StringTokenizer r=new StringTokenizer(s1,"/");

int n1=Integer.parseInt(r.nextToken());

int n2=Integer.parseInt(r.nextToken());

if(n1==1 || n1==3 || n1==5 ||n1==7 || n1==8 || n1==10 || n1==12)

n=31;

else if(n1==4 || n1==6 || n1==9 || n1==11)

n=30;

else if(n1==2){

if(n2%4==0)

n=29;

else

n=28; }

return n;

}

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

String shud be in "dd-MM-yyyy" format.Consider leap year conditions as well.

Solution:

package array;

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.\*;

public class met {

/\*\*

\* @param args

\* @throws IOException

\* @throws ParseException

\*/

public static void main(String[] args) throws IOException, ParseException {

// TODO Auto-generated method stub

String s1="12/02/2000";

SimpleDateFormat sdf=new SimpleDateFormat("dd/MM/yyyy");

Calendar cal=Calendar.getInstance();

Date d1=sdf.parse(s1);

cal.setTime(d1);

int n=cal.getActualMaximum(Calendar.DAY\_OF\_MONTH);

System.out.println(n);

}

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Q6) 2 Strings as input : Input1 : year, Input2 : 5

Find diff between current year and given year and check if its greater than or equal to input2

Solution:

package array;

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.\*;

public class met {

/\*\*

\* @param args

\* @throws IOException

\* @throws ParseException

\*/

public static void main(String[] args) throws IOException, ParseException {

// TODO Auto-generated method stub

String s1="12/06/2009";

Date d1=new Date();

int n=6;

SimpleDateFormat sdf=new SimpleDateFormat("dd/MM/yyyy");

Date d2=sdf.parse(s1);

int n1=d1.getYear();

int n2=d2.getYear();

if(Math.abs(n1-n2)>n)

System.out.println("true");

else

System.out.println("false");

}

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

5) string input1="13:45:15" input2="15:50:50" both in hh:mm:ss format and orginal hh:mm:ss is 23:59:59.

find interval between two inputs and output should be 1:5:36:05.

Solution:

package array;

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.\*;

public class met {

public static void main(String[] args) throws IOException, ParseException {

String s1="13:45:15";

String s2="15:50:50";

String s3="23:59:59";

SimpleDateFormat sdf=new SimpleDateFormat("k:m:s");

Date d1=sdf.parse(s1);

int n=d1.getHours();

int n1=d1.getMinutes();

int n2=d1.getSeconds();

Date d2=sdf.parse(s2);

//System.out.println(n2);

int p=d2.getHours();

int p1=d2.getMinutes();

int p2=d2.getSeconds();

int y=(n+p);

int y1=(n1+p1);

int y2=(n2+p2);

Date d3=sdf.parse(s3);

int q=d3.getHours();

int q1=d3.getMinutes();

int q2=d3.getSeconds();

int r=Math.abs(q-y);

int r1=Math.abs(q1-y1);

int r2=Math.abs(q2-y2);

StringBuffer sb=new StringBuffer();

sb.append("1").append(":").append(r).append(":").append(r1).append(":").append(r2);

System.out.println(sb);

}

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Q4) validate a date

input in a string format(MM/dd/yyyy)

check if month is between 1-12

check if date is between 1-31

year must be 4 digits.

Solution:

package array;

import java.util.\*;

public class met {

/\*\*

\* @param args

\* @throws ParseException

\*/

public static void main(String[] args) {

// TODO Auto-generated method stub

String s1="31/12/1997";

boolean b=false;

StringTokenizer t=new StringTokenizer(s1,"/");

String s2=t.nextToken();

String s3=t.nextToken();

String s4=t.nextToken();

int n1=Integer.parseInt(s2);

int n2=Integer.parseInt(s3);

int n3=Integer.parseInt(s4);

if(s4.length()==4)

{

if(n2>0 && n2<=12 && n1>0 && n1<=31)

{

b=true;

}

}

if(b)

System.out.println("true");

else

System.out.println("false");

}

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Add and Reverse

Given an int array and a number as input, write a program to add all the elements in the array greater than the given number. Finally reverse the digits of the obtained sum and print it.

Include a class UserMainCode with a static method ?addAndReverse? that accepts 2 arguments and returns an integer.The first argument corresponds to the integer array and the second argument corresponds to the number.

Create a class Main which would get the required input and call the static method addAndReverse present in the UserMainCode.

Example:

Input Array = {10,15,20,25,30,100}

Number = 15

sum = 20 + 25 + 30 + 100 = 175

output = 571

Input and Output Format:

The first line of the input consists of an integer that corresponds to the number of elements in the array.

The next n lines of the input consists of integers that correspond to the elements in the array.

The last line of the input consists of an integer that corresponds to the number.

Output consists of a single integer.

Sample Input

6

10

15

20

25

30

100

15

Sample Output

571

1.Add and Reverse

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 571

public class Main {

public static void main(String[] args) {

int[] a={10,15,20,25,30,100};

int b=15;

int i,sum=0,r=0;

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

{

if(a[i]>b)

{

sum=sum+a[i];

}

}

//System.out.println(sum);

while(sum!=0)

{

r=(r\*10)+(sum%10);

sum=sum/10;

}

System.out.println(r);;

}

}

???????????????????????????????????????????????????????????????????????????????????????????????

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner scan=new Scanner(System.in);

int size=scan.nextInt();

int a[]=new int[size];

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

{

a[i]=scan.nextInt();

}

int b=scan.nextInt();

System.out.println(UserMainCode.getvalues(a,b));

}

}

public class UserMainCode{

public static int getvalues( int a[],int b){

int i,sum=0,r=0;

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

{

if(a[i]>b)

{

sum=sum+a[i];

}

}

//System.out.println(sum);

while(sum!=0)

{

r=(r\*10)+(sum%10);

sum=sum/10;

}

//System.out.println(r);;

return r;

}

}

\

2..Add Time

Write a program to read two String variables containing time intervals in hh:mm:ss format.

Add the two time intervals and return a string in days:hours:minutes:seconds format where DD is number of days.

Hint: Maximum value for hh:mm:ss is 23:59:59

Include a class UserMainCode with a static method addTime which accepts the string values. The return type is the string.

Create a Class Main which would be used to accept the two string values and call the static method present in UserMainCode.

Input and Output Format:

Input consists of two string.

Output consists of a string.

Refer sample output for formatting specifications.

Sample Input 1:

12:45:30

13:50:45

Sample Output 1:

1:2:36:15

Sample Input 2:

23:59:59

23:59:59

Sample Output 2:

1:23:59:58

2.Add Time

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public class Main {

public static void main(String[] args) throws IOException, ParseException {

String s1="23:59:59";

String s2="23:59:59";

SimpleDateFormat sdf=new SimpleDateFormat("HH:mm:ss");

sdf.setTimeZone(TimeZone.getTimeZone("UTC"));

sdf.setTimeZone(TimeZone.getTimeZone("s1"));

sdf.setTimeZone(TimeZone.getTimeZone("s2"));

Date d1=sdf.parse(s1);

Date d2=sdf.parse(s2);

long add=d1.getTime()+d2.getTime();

String s=sdf.format(add);

Calendar cal=Calendar.getInstance();

cal.setTime(sdf.parse(s));

int FindDay=cal.get(Calendar.DAY\_OF\_MONTH);

if(FindDay>1)

FindDay=FindDay-1;

String op=FindDay+":"+s;

System.out.println(op);

}

}

?????????????????????????????????????????????????????????????????????????????

import java.io.IOException;

import java.text.ParseException;

import java.util.Scanner;

public class Main {

public static void main(String[] args) throws IOException, ParseException {

Scanner scan=new Scanner(System.in);

String s1=scan.nextLine();

String s2=scan.nextLine();

System.out.println(Usercode.Getdate(s1,s2));

}

}

import java.io.IOException;

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Calendar;

import java.util.Date; //import java.util.sql;xxxxxxxxxxxxx

import java.util.TimeZone;

public class Usercode {

public static String Getdate (String s1, String s2) throws IOException, ParseException {

SimpleDateFormat sdf=new SimpleDateFormat("HH:mm:ss");

sdf.setTimeZone(TimeZone.getTimeZone("UTC"));

sdf.setTimeZone(TimeZone.getTimeZone("s1"));

sdf.setTimeZone(TimeZone.getTimeZone("s2"));

Date d1=sdf.parse(s1);

Date d2=sdf.parse(s2);

long add=d1.getTime()+d2.getTime();

String s=sdf.format(add);

Calendar cal=Calendar.getInstance();

cal.setTime(sdf.parse(s));

int FindDay=cal.get(Calendar.DAY\_OF\_MONTH);

if(FindDay>1)

FindDay=FindDay-1;

String op=FindDay+":"+s;

//System.out.println(op);

return op;

}

}

3.Adjacent Swaps

Write a Program that accepts a string as a parameter and returns the string with each pair of adjacent letters reversed. If the string has an odd number of letters, the last letter is unchanged.

Include a class UserMainCode with a static method swapPairs which accepts a string. The return type is string which is reversed pair of letters.

Create a Class Main which would be used to accept two Input strings and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string with maximum size of 100 characters.

Output consists of a single string.

Refer sample output for formatting specifications.

Sample Input 1:

forget

Sample Output 1:

ofgrte

Sample Input 2:

New York

Sample Output 2:

eN woYkr

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public class Main

{

public static void main(String[] args) {

String s1="forget";

getvalues(s1);

}

public static void getvalues(String s1)

{

StringBuffer sb=new StringBuffer();

int l=s1.length();

if(l%2==0)

{

for(int i=0;i<s1.length()-1;i=i+2)

{

char a=s1.charAt(i);

char b=s1.charAt(i+1);

sb.append(b).append(a);

}

System.out.println(sb);

}

else

{

for(int i = 0;i<s1.length()-1;i=i+2)

{

char a=s1.charAt(i);

char b=s1.charAt(i+1);

sb.append(b).append(a);

System.out.println(sb);

}

sb.append(s1.charAt(l-1));

System.out.println(sb);

}

}

}

??????????????????????????????????????????????????????????????????????????????????????

public class Main

{

public static void main(String[] args) {

Scanner scan=new Scanner (System.in);

String s1=scan.nextLine();

System.out.println(Usercode.getvalues(s1));

}

}

public class Usercode

{

public static String getvalues(String s1)

{

StringBuffer sb=new StringBuffer();

int l=s1.length();

if(l%2==0)

{

for(int i=0;i<s1.length()-1;i=i+2)

{

char a=s1.charAt(i);

char b=s1.charAt(i+1);

sb.append(b).append(a);

}

return sb.toString();

}

else

{

for(int i = 0;i<s1.length()-1;i=i+2)

{

char a=s1.charAt(i);

char b=s1.charAt(i+1);

sb.append(b).append(a);

return sb.toString();

}

sb.append(s1.charAt(l-1));

return sb.toString();

}

}

}

4.Age for Voting(getAge)

Given a date of birth (dd/MM/yyyy) of a person in string, compute his age as of 01/01/2015.

If his age is greater than 18, then println eligible else println not-eligible.

Include a class UserMainCode with a static method getAge which accepts the string value. The return type is the string.

Create a Class Main which would be used to accept the two string values and call the static method present in UserMainCode.

Input and Output Format:

Input consists of two string.

Output consists of a string.

Refer sample output for formatting specifications.

Sample Input 1:

16/11/1991

Sample Output 1:

eligible

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.Scanner;

public class Main {

public static void main(String[] args)

{

Scanner sc=new Scanner(System.in);

String s =sc.nextLine();

System.out.println(UserMainCode.display(s));}}

import java.text.SimpleDateFormat;

import java.util.Date;

public class UserMainCode {

public static String display(String n)

{

int year=0;

String now="01/01/2015";

SimpleDateFormat sdf1=new SimpleDateFormat("dd/MM/yyyy");

try

{

Date d=sdf1.parse(n);

Date d1=sdf1.parse(now);

sdf1.setLenient(false);

int y=d.getYear();

int y1=d1.getYear();

int m=d.getMonth();

int m1=d1.getMonth();

int day=d.getDay();

int day1=d1.getDay();

year=y1-y;

if(m>m1)

year--;

else if(m==m1)

{if(day<day1)

year--;

}

}

catch(Exception e)

{

e.printStackTrace();

}

if(year>18)

return "eligible";

else

return "not-eligible";

}}

?????????????????????????????????????????????????????????

import java.util.Scanner;

public class Main {

public static void main(String[] args)

{

Scanner sc=new Scanner(System.in);

String s =sc.nextLine();

//int a=sc.nextInt();

//

System.out.println(Usercode.display(s));}}

import java.text.SimpleDateFormat;

import java.util.Date;

public class Usercode {

public static String display(String n)

{

int year=0;

String now="01/01/2015";

SimpleDateFormat sdf1=new SimpleDateFormat("dd/MM/yyyy");

try

{

Date d=sdf1.parse(n);

Date d1=sdf1.parse(now);

sdf1.setLenient(false);

int y=d.getYear();

int y1=d1.getYear();

int m=d.getMonth();

int m1=d1.getMonth();

int day=d.getDay();

int day1=d1.getDay();

year=y1-y;

if(m>m1)

year--;

else if(m==m1)

{if(day<day1)

year--;

}

}

catch(Exception e)

{

e.printStackTrace();

}

if(year>18)

return "eligible";

else

return "not-eligible";

}}

5.All Numbers(validateNumber)

Write a program to read a string array and return 1 if all the elements of the array are numbers, else return -1.

Include a class UserMainCode with a static method validateNumber which accepts a string aray. The return type (integer) should be -1 or 1 based on the above rules.

Create a Class Main which would be used to accept Input string array and call the static method present in UserMainCode.

The string array is said to be valid if all the elements in the array are numbers. Else it is invalid.

Input and Output Format:

Input consists of an integer specifying the size of string array followed by n strings.

Refer sample output for formatting specifications.

Sample Input 1:

4

123

24.5

23

one

Sample Output 1:

invalid

Sample Input 2:

2

123

24.5

Sample Output 2:

valid

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.Scanner;

public class kape {

public static void main(String args[]){

Scanner sc = new Scanner(System.in);

int size=sc.nextInt();

String []s1=new String[size];

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

{

s1[i]=sc.next();

}

System.out.println(kape1.display(s1));

}}

public class kape1 {

public static int display(String a[])

{int c=0;

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

{

String s=a[i];

int k=0;

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

{

char a1=s.charAt(j);

//System.out

if(a1>='0' && a1<='9' || a1=='.' )

k++;

}

if(k==s.length())

c++;

}

if(c==a.length)

return 1;

else

return -1;

}}

????????????????????????????????????????????????????????????????????????????

import java.util.Scanner;

public class Main {

public static void main(String args[]){

Scanner sc = new Scanner(System.in);

int size=sc.nextInt();

String []s1=new String[size];

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

{

s1[i]=sc.next();

}

int res = Usercode.display(s1);

if(res==1){

System.out.println("Valid");

}

else{

System.out.println("Invalid");

}

// System.out.println();

}}

public class Usercode {

public static int display(String a[])

{int c=0;

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

{

String s=a[i];

int k=0;

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

{

char a1=s.charAt(j);

//System.out

if(a1>='0' && a1<='9' || a1=='.' )

k++;

}

if(k==s.length())

c++;

}

if(c==a.length)

return 1;

else

return -1;

}}

6.All Vowels

Write a program to read a string and check if given string contains exactly five vowels in any order. Print ?Yes? if the condition satisfies, else print ?No?.

Assume there is no repetition of any vowel in the given string and all characters are lowercase.

Include a class UserMainCode with a static method testVowels which accepts a string. The return type (Integer) should return 1 if all vowels are present, else return 2.

Create a Class Main which would be used to accept a string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Output consists of a string (?Yes? or ?No?).

Refer sample output for formatting specifications.

Sample Input 1:

acbisouzze

Sample Output 1:

Yes

Sample Input 2:

cbisouzze

Sample Output 2:

No

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.Iterator;

import java.util.LinkedHashSet;

public class Main {

/\*\*

\* @param args

\*/

public static void main(String[] args) {

String s1="alphabet";

getvalues(s1);

}

public static void getvalues(String s1) {

String s2="aeiou";

StringBuffer sb=new StringBuffer();

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

{

for(int j=0;j<s2.length();j++)

{

if(s1.charAt(i)==s2.charAt(j))

{

sb.append(s1.charAt(i));

}

}

}

if(sb.toString().equals(s2))

{

System.out.println("true");

}

else

System.out.println("false");

}

}

????????????????????????????????????????????????????????????????

import java.util.Scanner;

public class Main

{

public static void main(String[] args)

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

String s1=sc.nextLine();

int res=Usercode.testVowel(s1);

if(res==1)

System.out.println("Yes");

else

System.out.println("No");

}

}

public class Usercode {

public static Integer testVowel(String s1) {

int a=0;

int e=0;

int s=0;

int o=0;

int u=0;

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

{

if(s1.charAt(i)=='a')

a++;

else if(s1.charAt(i)=='e')

e++;

else if(s1.charAt(i)=='i')

s++;

else if(s1.charAt(i)=='o')

o++;

else if(s1.charAt(i)=='u')

u++;

}

if(a==1&&e==1&&s==1&&o==1&&u==1)

return 1;

else

return 2;

}

}

7.8.Anagrams

Write a program to check whether the two given strings are anagrams.

Note: Rearranging the letters of a word or phrase to produce a new word or phrase, using all the original letters exactly once is called Anagram."

Include a class UserMainCode with a static method ?getAnagram? that accepts 2 strings as arguments and returns an int. The method returns 1 if the 2 strings are anagrams. Else it returns -1.

Create a class Main which would get 2 Strings as input and call the static method getAnagram present in the UserMainCode.

Input and Output Format:

Input consists of 2 strings. Assume that all characters in the string are lower case letters.

Output consists of a string that is either ?Anagrams? or ?Not Anagrams?.

Sample Input 1:

eleven plus two

twelve plus one

Sample Output 1:

Anagrams

Sample Input 2:

orchestra

carthorse

Sample Output 2:

Anagrams

Sample Input 3:

cognizant

technologies

Sample Output 3:

Not Anagrams

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.ArrayList;

import java.util.Collections;

import java.util.List;

public class Main {

public static void main(String[] args) {

String s1="Desperation";

String s2="A Rope Ends It";

List<Character> l1=new ArrayList<Character>();

List<Character> l2=new ArrayList<Character>();

List<Character> l3=new ArrayList<Character>();

String s3=s1.toUpperCase();

String s4=s2.toUpperCase();

for (int i = 0; i < s3.length(); i++)

{

l1.add(s3.charAt(i));

}

for (int i = 0; i < s4.length(); i++)

{

l2.add(s4.charAt(i));

}

l3.add(' ');

l1.removeAll(l3);

l2.removeAll(l3);

Collections.sort(l1);

Collections.sort(l2);

System.out.println(l1);

System.out.println(l2);

if(l1.equals(l2))

System.out.println("true");

else

System.out.println("false");

}

}

?????????????????????????????????????????????????????????????????????????????

import java.util.ArrayList;

import java.util.Collections;

import java.util.List;

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

//String s1="Desperation";

//String s2="A Rope Ends It";

Scanner sc=new Scanner(System.in);

String s1=sc.nextLine();

String s2=sc.nextLine();

int z=Usercode.Anagrams(s1,s2);

if(z==1){

System.out.println("Anagrams");

}

else{

System.out.println("Not anagrams");

}

}

}

import java.util.ArrayList;

import java.util.Collections;

import java.util.List;

public class Usercode {

public static int Anagrams(String s1, String s2) {

int res=0;

List<Character> l1=new ArrayList<Character>();

List<Character> l2=new ArrayList<Character>();

List<Character> l3=new ArrayList<Character>();

String s3=s1.toUpperCase();

String s4=s2.toUpperCase();

for (int i = 0; i < s3.length(); i++)

{

l1.add(s3.charAt(i));

}

for (int i = 0; i < s4.length(); i++)

{

l2.add(s4.charAt(i));

}

l3.add(' ');

l1.removeAll(l3);

l2.removeAll(l3);

Collections.sort(l1);

Collections.sort(l2);

//System.out.println(l1);

//System.out.println(l2);

if(l1.equals(l2))

//System.out.println("true");

res=1;

else {

//System.out.println("false");

res=-1;

}

return res;

}

}

9.Array List Sorting and Merging (sortMergedArrayList)

Write a code to read two int array lists of size 5 each as input and to merge the two arrayLists, sort the merged arraylist in ascending order and fetch the elements at 2nd, 6th and 8th index into a new arrayList and return the final ArrayList.

Include a class UserMainCode with a static method sortMergedArrayList which accepts 2 ArrayLists.

The return type is an ArrayList with elements from 2,6 and 8th index position .Array index starts from position 0.

Create a Main class which gets two array list of size 5 as input and call the static method sortMergedArrayList present in the UserMainCode.

Input and Output Format:

Input consists of two array lists of size 5.

Output is an array list .

Note - The first element is at index 0.

Refer sample output for formatting specifications.

Sample Input 1:

3

1

17

11

19

5

2

7

6

20

Sample Output 1:

3

11

19

Sample Input 2:

1

2

3

4

5

6

7

8

9

10

Sample Output 2:

3

7

9

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 3/1/17/11/19

public class kape {

public static void main(String[] args)

{

Scanner sc=new Scanner(System.in);

ArrayList<Integer> al1=new ArrayList<Integer>();

ArrayList<Integer> al2=new ArrayList<Integer>();

ArrayList<Integer> ans=new ArrayList<Integer>();

for(int i=0;i<5;i++)

al1.add(sc.nextInt());

for(int j=0;j<5;j++)

al2.add(sc.nextInt());

ans=kape1.answer(al1,al2);

//System.out.println(ans);

for(int k=0;k<3;k++)

System.out.println(ans.get(k));

}

}

import java.util.ArrayList;

import java.util.Collections;

public class kape1 {

public static ArrayList<Integer> answer (ArrayList<Integer> al1, ArrayList<Integer> al2)

{

al1.addAll(al2);

Collections.sort(al1);

ArrayList<Integer> ans=new ArrayList<Integer>();

ans.add(al1.get(2));

ans.add(al1.get(6));

ans.add(al1.get(8));

return ans;

}

?????????????????????????????????????????????????????????????????????????????????????////

import java.util.ArrayList;

import java.util.Scanner;

public class Main {

public static void main(String[] args)

{

Scanner sc=new Scanner(System.in);

ArrayList<Integer> al1=new ArrayList<Integer>();

ArrayList<Integer> al2=new ArrayList<Integer>();

ArrayList<Integer> ans=new ArrayList<Integer>();

for(int i=0;i<5;i++)

al1.add(sc.nextInt());

for(int j=0;j<5;j++)

al2.add(sc.nextInt());

ans=Usercode.answer(al1,al2);

//System.out.println(ans);

for(int k=0;k<3;k++)

System.out.println(ans.get(k));

}

}

import java.util.ArrayList;

import java.util.Collections;

public class Usercode {

public static ArrayList<Integer> answer (ArrayList<Integer> al1, ArrayList<Integer> al2)

{

al1.addAll(al2);

Collections.sort(al1);

ArrayList<Integer> ans=new ArrayList<Integer>();

ans.add(al1.get(2));

ans.add(al1.get(6));

ans.add(al1.get(8));

return ans;

}}

10.ArrayFront (scanArray)

Write a program to read a integer array and return true if one of the first 4 elements in the array is 9 else return false.

Note: The array length may be less than 4.

Include a class UserMainCode with a static method scanArray which accepts the integer array. The return type is true / false.

Create a Class Main which would be used to accept the integer array and call the static method present in UserMainCode.

Input and Output Format:

Input consists of an integer n which is the number of elements followed by n integer values.

Output consists of TRUE / FALSE.

Refer sample output for formatting specifications.

Sample Input 1:

6

1

2

3

4

5

6

Sample Output 1:

FALSE

Sample Input 2:

3

1

2

9

Sample Output 2:

TRUE

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 6/1/2/3/4/5/6

import java.util.Scanner;

public class kapes3 {

public static void main(String []args){

Scanner sc=new Scanner(System.in);

int s=sc.nextInt();

int []a=new int[s];

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

{

a[i]=sc.nextInt();

}

System.out.println(kapes4.display(a));

}

}

public class kapes4

{public static boolean display(int[] a)

{

int u=0,l=0;

boolean b=false;

if(a.length>=4)

l=4;

else

l=a.length;

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

if(a[i]==9)

u=10;

if(u==10)

b=true;

return b;

}

}

????????????????????????????????????????????????????????????????????????????????????????????????????/

import java.util.Scanner;

public class Main {

public static void main(String []args){

Scanner sc=new Scanner(System.in);

int s=sc.nextInt();

int []a=new int[s];

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

{

a[i]=sc.nextInt();

}

boolean z=Usercode.display(a);

if(z==false){

System.out.println("FALSE");

}

else{

System.out.println("TRUE");

}

}

}

public class Usercode

{public static boolean display(int[] a)

{

int u=0,l=0;

boolean b=false;

if(a.length>=4)

l=4;

else

l=a.length;

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

if(a[i]==9)

u=10;

if(u==10)

b=true;

return b;

}

}

11.ArrayList and Set Operations (performSetOperations)

Write a program that performs the following actions:

Read 2n integers as input & a set operator (of type char).

Create two arraylists to store n elements in each arraylist.

Write a function performSetOperations which accepts these two arraylist and the set operator as input.

The function would perform the following set operations:.

'+' for SET-UNION

'\*' for SET-INTERSECTION

'-' for SET-DIFFERENCE

Refer to sample inputs for more details.

Return the resultant arraylist.

Include a class UserMainCode with the static method performSetOperations which accepts two arraylist and returns an arraylist.

Create a Class Main which would be used to read 2n+1 integers and call the static method present in UserMainCode.

Note:

- The index of first element is 0.

Input and Output Format:

Input consists of 2n+2 integers. The first integer denotes the size of the arraylist, the next n integers are values to the first arraylist, and the next n integers are values to the second arraylist and the last input corresponds to that set operation type.

Output consists of a modified arraylist as per step 4.

Refer sample output for formatting specifications.

Sample Input 1:

3

1

2

3

3

5

7

+

Sample Output 1:

1

2

3

5

7

Sample Input 2:

4

10

9

8

7

2

4

6

8

\*

Sample Output 2:

8

Sample Input 3:

4

5

10

15

20

0

10

12

20

-

Sample Output 3:

5

15

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.ArrayList;

import java.util.Scanner;

public class Main {

public static void main(String args[]){

Scanner sc = new Scanner(System.in);

int n=Integer.parseInt(sc.nextLine());

ArrayList<Integer>a1=new ArrayList<Integer>();

ArrayList<Integer>a2=new ArrayList<Integer>();

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

a1.add(Integer.parseInt(sc.nextLine()));

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

a2.add(Integer.parseInt(sc.nextLine()));

char c=sc.nextLine().charAt(0);

System.out.println(UserMainCode.getvalues(a1,a2,c));

}

}

import java.util.ArrayList;

public class UserMainCode {

public static ArrayList<Integer> getvalues(ArrayList<Integer>a1,ArrayList<Integer>a2,char c)

{

ArrayList<Integer>op1=new ArrayList<Integer>();int k=0;

switch(c)

{

case '+':

a1.removeAll(a2);

a1.addAll(a2);

op1=a1;

break;

case '\*':

a1.retainAll(a2);

op1=a1;

break;

case '-':

for(int i=0;i<a1.size();i++)

{

k=0;

for(int j=0;j<a2.size();j++)

{

if(a1.get(i)==a2.get(j))

k=1;

}

if(k==0)

op1.add(a1.get(i));

}

break;

}

return op1;

}}

??????????????????????????????????????????????????????????????????????????????????????????//

import java.util.ArrayList;

import java.util.Scanner;

public class Main {

public static void main(String args[]){

Scanner sc = new Scanner(System.in);

int n=Integer.parseInt(sc.nextLine());

ArrayList<Integer>a1=new ArrayList<Integer>();

ArrayList<Integer>a2=new ArrayList<Integer>();

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

a1.add(Integer.parseInt(sc.nextLine()));

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

a2.add(Integer.parseInt(sc.nextLine()));

char c=sc.nextLine().charAt(0);

System.out.println(Usercode.getvalues(a1,a2,c));

}

}

import java.util.ArrayList;

public class Usercode {

public static ArrayList<Integer> getvalues(ArrayList<Integer>a1,ArrayList<Integer>a2,char c)

{

ArrayList<Integer>op1=new ArrayList<Integer>();int k=0;

switch(c)

{

case '+':

a1.removeAll(a2);

a1.addAll(a2);

op1=a1;

break;

case '\*':

a1.retainAll(a2);

op1=a1;

break;

case '-':

for(int i=0;i<a1.size();i++)

{

k=0;

for(int j=0;j<a2.size();j++)

{

if(a1.get(i)==a2.get(j))

k=1;

}

if(k==0)

op1.add(a1.get(i));

}

break;

}

return op1;

}}

12. ArrayList Manipulation (generateOddEvenList)

Write a program that performs the following actions:

Read 2n integers as input.

Create two arraylists to store n elements in each arraylist.

Write a function generateOddEvenList which accepts these two arraylist as input.

The function fetch the odd index elements from first array list and even index elements from second array list and add them to a new array list according to their index.

Return the arraylist.

Include a class UserMainCode with the static method generateOddEvenList which accepts two arraylist and returns an arraylist.

Create a Class Main which would be used to read 2n integers and call the static method present in UserMainCode.

Note:

- The index of first element is 0.

- Consider 0 as an even number.

- Maintain order in the output array list

Input and Output Format:

Input consists of 2n+1 integers. The first integer denotes the size of the arraylist, the next n integers are values to the first arraylist, and the last n integers are values to the second arraylist.

Output consists of a modified arraylist as per step 4.

Refer sample output for formatting specifications.

Sample Input 1:

5

12

13

14

15

16

2

3

4

5

6

Sample Output 1:

2

13

4

15

6

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.ArrayList;

import java.util.Iterator;

import java.util.Scanner;

public class main {

public static void main(String []args){

Scanner sc=new Scanner(System.in);

int s=Integer.parseInt(sc.nextLine());

ArrayList<Integer>al1=new ArrayList<Integer>();

ArrayList<Integer>al2=new ArrayList<Integer>();

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

al1.add(sc.nextInt());

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

al2.add(sc.nextInt());

ArrayList<Integer>al3=new ArrayList<Integer>();

al3=Usermaincode.display(al1,al2);

Iterator<Integer> it=al3.iterator();

while(it.hasNext())

{

int n=it.next();

System.out.println(n);

}

}

}

import java.util.ArrayList;

import java.util.Iterator;

public class Usermaincode

{

public static ArrayList<Integer> display (ArrayList<Integer>al1,ArrayList<Integer>al2)

{

ArrayList<Integer>al3=new ArrayList<Integer>();

for(int i=0;i<al1.size();i++)

{

if(i%2==0)

al3.add(al2.get(i));

else

al3.add(al1.get(i));

}

return al3;

}}

13. ArrayList to String Array

Write a program that performs the following actions:

Read n strings as input.

Create an arraylist to store the above n strings in this arraylist.

Write a function convertToStringArray which accepts the arraylist as input.

The function should sort the elements (strings) present in the arraylist and convert them into a string array.

Return the array.

Include a class UserMainCode with the static method convertToStringArray which accepts an arraylist and returns an array.

Create a Class Main which would be used to read n strings and call the static method present in UserMainCode.

Input and Output Format:

Input consists of n+1 integers. The first integer denotes the size of the arraylist, the next n strings are values to the arraylist.

Output consists of an arrayas per step 4.

Refer sample output for formatting specifications.

Sample Input 1:

4

a

d

c

b

Sample Output 1:

a

b

c

d

Close

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.\*;

public class Main {

public static void main(String[] args) {

List<String> l1=new ArrayList<String>();

l1.add("Apple");

l1.add("Chery");

l1.add("Grapes");

List<String> l2=new ArrayList<String>();

l2.add("Orange");

l2.add("Mango");

l2.add("Melon");

l2.add("Apple");

String[] s2=fruitsList(l1,l2);

for(String s3:s2)

System.out.println(s3);

}

public static String[] fruitsList(List<String> l1, List<String> l2){

List<String> l3=new ArrayList<String>();

for(int i=0;i<l1.size();i++){

String s1=l1.get(i);

if(s1.charAt(s1.length()-1)!='a' && s1.charAt(s1.length()-1)!='A' && s1.charAt(s1.length()-1)!='e' && s1.charAt(s1.length()-1)!='E')

l3.add(s1); }

for(int i=0;i<l2.size();i++){

String s1=l2.get(i);

if(s1.charAt(0)!='m' && s1.charAt(0)!='M' && s1.charAt(0)!='a' && s1.charAt(0)!='A')

l3.add(s1); }

Collections.sort(l3);

String[] s2=new String[l3.size()];

for(int i=0;i<s2.length;i++)

s2[i]=l3.get(i);

return s2;

}

14. ArrayList to String Array(convertToStringArray)

Write a program that performs the following actions:

1.Read m strings as input (fruit names).

2.Create an arraylist to store the above m strings in this arraylist.

3.Read n strings as input (fruit names).

4.Create an arraylist to store the above n strings in this arraylist.

5.Write a function fruitSelector which accepts the arraylists as input.

6.Remove all fruits whose name ends with 'a' or 'e' from first arrayList and remove all fruits whose name begins with 'm' or 'a' from second arrayList then combine the two lists and return the final output as a String array.

7.If the array is empty the program will print as ?No fruit found?

Include a class UserMainCode with the static method fruitSelector which accepts the two arraylists and returns an array.

Create a Class Main which would be used to read n strings and call the static method present in UserMainCode.

Input and Output Format:

Input consists of an integer (m) denoting the size of first arraylist. The next m elements would be the values of the first arraylist. The next input would be n denoting the size of the second arraylist. The next n elements would be the values of the second arraylist.

Output consists of an array as per step 6. Refer sample output for formatting specifications.

Sample Input 1:

3

Apple

Cherry

Grapes

4

Orange

Mango

Melon

Apple

Sample Output 1:

Cherry

Grapes

Orange

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.Collections;

import java.util.List;

public class UserMainCode{

public static String[]convertToString(List <String>l)

{

Collections.sort(l);

String answer[]=new String[l.size()];

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

{

answer[i]=l.get(i);

}

return answer;

}

}

15. Asterisk & Characters

Write a program to read a string and return true or false based on the below rule:

1. Return true if for every '\*' in the string, there are same characters both side immediately before and after the star, else return false.

Include a class UserMainCode with a static method scanStarNeighbors which accepts the string. The return type is the boolean TRUE or FALSE based on the rule.

Note: The check is case-insensitive.

Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Output consists of TRUE or FALSE.

Refer sample output for formatting specifications.

Sample Input 1:

Hello\*World

Sample Output 1:

FALSE

Sample Input 2:

Welcome\*elizabeth

Sample Output 2:

TRUE

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.\*;

public class Main {

public static void main(String[] args) {

String input="Welcome\*elizabeth";

System.out.println(characterCheck(input));

}

public static boolean characterCheck(String input) {

boolean b=false;

StringTokenizer t=new StringTokenizer(input,"\*");

String s1=t.nextToken();

String s2=t.nextToken();

String s3=s1.substring(s1.length()-1);

String s4=s2.substring(0,1);

if(s3.equalsIgnoreCase(s4))

b=true;

return b;

}

}

????????????????????????????????????????????????????????????????????????????/

import java.util.\*;

public class Main {

public static void main(String[] args) {

Scanner scan=new Scanner(System.in);

String s1=scan.nextLine();

boolean z=Usercode.characterCheck(s1);

if(z==false){

System.out.println("FALSE");

}

else

{

System.out.println("TRUE");

}

}

}

import java.util.StringTokenizer;

public class Usercode{

public static boolean characterCheck(String input) {

boolean b=false;

StringTokenizer t=new StringTokenizer(input,"\*");

String s1=t.nextToken();

String s2=t.nextToken();

String s3=s1.substring(s1.length()-1);

String s4=s2.substring(0,1);

if(s3.equalsIgnoreCase(s4))

b=true;

return b;

}

}

16. Average of Elements in Hashmap (avgOfEven)

Given a method with a HashMap as input. Write code to find out avg of all values whose keys are even numbers. Round the average to two decimal places and return as output.

[Hint : If the average is 5.901, the rounded average value is 5.9 . It the average is 6.333, the rounded average value is 6.33 . ]

Include a class UserMainCode with a static method avgOfEven which accepts a HashMap as input.

The return type of the output is a floating point value which is the average of all values whose key elements are even numbers.

Create a class Main which would get the input and call the static method avgOfEven present in the UserMainCode.

Input and Output Format:

Input consists of the number of elements in the HashMap and the HashMap.

Output is a floating point value that corresponds to the average.

Refer sample output for formatting specifications.

Sample Input 1:

3

1

2.3

2

4.1

6

6.2

Sample Output 1:

5.15

Sample Input 2:

3

9

3.1

4

6.3

1

2.6

Sample Output 2:

6.3

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 3/1/2.3/2/4.1/6/6.2

import java.util.HashMap;

import java.util.Scanner;

public class kapes3 {

public static void main(String []args){

Scanner sc=new Scanner(System.in);

int s=Integer.parseInt(sc.nextLine());

HashMap<Integer,Float>hm=new HashMap<Integer,Float>();

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

{

hm.put(Integer.parseInt(sc.nextLine()),Float.parseFloat(sc.nextLine()));

}

System.out.println(kapes4.display(hm));

}

}

import java.text.DecimalFormat;

import java.util.HashMap;

import java.util.Iterator;

public class kapes4

{public static String display(HashMap<Integer,Float>hm)

{

float sum=0;

int count=0;

DecimalFormat df=new DecimalFormat("#.00");

Iterator<Integer> it=hm.keySet().iterator();

while(it.hasNext())

{

int y=it.next();

if(y%2==0)

{

sum=(float) (sum+hm.get(y));

count++;

}}

float d=sum/count;

return df.format(d);

}

}

??????????????????????????????????????????????????????????????????????????????????????????????????????????/////

import java.util.HashMap;

import java.util.Scanner;

public class Main {

public static void main(String []args){

Scanner sc=new Scanner(System.in);

int s=Integer.parseInt(sc.nextLine());

HashMap<Integer,Float>hm=new HashMap<Integer,Float>();

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

{

hm.put(Integer.parseInt(sc.nextLine()),Float.parseFloat(sc.nextLine()));

}

System.out.println(Usercode.display(hm));

}

}

import java.text.DecimalFormat;

import java.util.HashMap;

import java.util.Iterator;

public class Usercode

{public static float display(HashMap<Integer,Float>hm)

{

float sum=0;

int count=0;

DecimalFormat df=new DecimalFormat("#.00");

Iterator<Integer> it=hm.keySet().iterator();

while(it.hasNext())

{

int y=it.next();

if(y%2==0)

{

sum=(float) (sum+hm.get(y));

count++;

}}

float d=sum/count;

float re=Float.parseFloat(df.format(d));

return re;

}

}

17. Average of Prime Locations (averageElements)

Write a program to read an integer array and find the average of the numbers located on the Prime location(indexes).

Round the avarage to two decimal places.

Assume that the array starts with index 0.

Include a class UserMainCode with a static method averageElements which accepts a single integer array. The return type (double) should be the average.

Create a Class Main which would be used to accept Input array and call the static method present in UserMainCode.

Input and Output Format:

Input consists of n+1 integers. The first integer corresponds to n, the number of elements in the array. The next 'n' integers correspond to the elements in the array.

Output consists of a single Double value.

Refer sample output for formatting specifications.

Assume that the maximum number of elements in the array is 20.

Sample Input 1:

8

4

1

7

6

5

8

6

9

Sample Output 1:

7.5

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 8/4/1/7/6/5/8/6/9

import java.util.Scanner;

public class kape{

public static void main(String[] args) {

Scanner sc=new Scanner(System.in);

int []a=new int[20];

int n=Integer.parseInt(sc.nextLine());

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

{

a[i]=Integer.parseInt(sc.nextLine());

}

System.out.print(kape1.display(n,a));

}}

public class kape1{

public static double display(int n,int[]a)

{

int count=0,sum=0,n1=0;

double avg=0;

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

{

count=0;

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

{

if(i%j==0)

{

count++;

}

}

if(count==2)

{

sum=sum+a[i];

n1=n1+1;

}

}

avg=(double)(sum)/n1;

return avg;

}}

?????????????????????????????????????????????????????????????????????????????????

import java.util.Scanner;

public class Main{

public static void main(String[] args) {

Scanner sc=new Scanner(System.in);

int []a=new int[20];

int n=Integer.parseInt(sc.nxtLine());

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

{

a[i]=Integer.parseInt(sc.nextLine());

}

System.out.print(Usercode.display(n,a));

}}

public class Usercode{

public static double display(int n,int[]a)

{

int count=0,sum=0,n1=0;

double avg=0;

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

{

count=0;

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

{

if(i%j==0)

{

count++;

}

}

if(count==2)

{

sum=sum+a[i];

n1=n1+1;

}

}

avg=(double)(sum)/n1;

return avg;

}}

18.Average of Primes (addPrimeIndex)

Write a program to read an array and find average of all elements located at index i, where i is a prime number. Type cast the average to an int and return as output. The index starts from 0.

Include a class UserMainCode with a static method addPrimeIndex which accepts a single integer array. The return type (integer) should be the average of all elements located at index i where i is a prime number.

Create a Class Main which would be used to accept Input array and call the static method present in UserMainCode.

Input and Output Format:

Input consists of n+1 integers. The first integer corresponds to n, the number of elements in the array. The next 'n' integers correspond to the elements in the array.

Output consists of a single Integer.

Refer sample output for formatting specifications.

Assume that the maximum number of elements in the array is 20 and minimum number of elements is 3.

Sample Input 1:

4

2

5

2

4

Sample Output 1:

3

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.Scanner;

public class kapes3{

public static void main (String[] args)

{

// your code goes here

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

int[] arr = new int[n];

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

arr[i] = sc.nextInt();

}

System.out.println(kapes4.getvalues(arr, n));

}}

public class kapes4{

public static int getvalues(int[] a,int n) {

int sum=0;

int count=0;

int sum\_count=0;

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

{

count=0;

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

{

if(i%j==0)

{

count++;

if(i<0)

break;

}

}

if(count==2)

{

sum=sum+a[i];

sum\_count++;

}

}

int avg=sum/sum\_count;

return avg;

}}

19. BOUNDARY AVERAGE (getBoundaryAverage)

Given an int array as input, write a program to compute the average of the maximum and minimum element in the array.

Include a class UserMainCode with a static method ?getBoundaryAverage? that accepts an integer array as argument and returns a float that corresponds to the average of the maximum and minimum element in the array.

Create a class Main which would get the input array and call the static method getBoundaryAverage present in the UserMainCode.

Input and Output Format:

The first line of the input consists of an integer n, that corresponds to the size of the array.

The next n lines consist of integers that correspond to the elements in the array.

Assume that the maximum number of elements in the array is 10.

Output consists of a single float value that corresponds to the average of the max and min element in the array.

Sample Input :

6

3

6

9

4

2

5

Sample Output:

5.5

BOUNDARY AVERAGE (getBoundaryAverage)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 6/3/6/9/4/2/5

import java.util.HashMap;

import java.util.Iterator;

import java.util.Scanner;

public class main {

public static void main(String []args){

Scanner sc=new Scanner(System.in);

int size=sc.nextInt();

int n[]=new int[size];

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

{

n[i]=sc.nextInt();

}

System.out.println(Usermaincode.display(n));

}

}

public class Usermaincode {

public static float display(int num[])

{

int max = num[0];

int min=num[num.length-1];

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

{

if (num[i] > max)

max = num[i];

if (num[i]<min)

min=num[i];

}

return (float)(max+min)/2;

}

}

20. Calculate Average – Hash Map

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Write amethod that accepts the input data as a hash map and finds out the avg of all values whose keys are odd numbers.

Include a class UserMainCode with a static method calculateAverage which accepts a HashMap and the size of the HashMap. The return type (Double) should return the calculated average. Round the average to two decimal places and return it.

Create a Class Main which would be used to accept Input values and store it as a hash map, and call the static method present in UserMainCode.

Input and Output Format:

Input consists of an integer n corresponds to number of hash map values, followed by 2n values. (index followed by value).

Output consists of a Double.

Refer sample input and output for formatting specifications.

Sample Input :

4

1

3.41

2

4.1

3

1.61

4

2.5

Sample Output :

2.51

import java.util.HashMap;

import java.util.Scanner;

public class main12 {

public static void main(String[] arg)

{

HashMap<Integer,Double> hm=new HashMap<Integer,Double>();

Scanner sc=new Scanner(System.in);

int n=sc.nextInt();

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

{

int a=sc.nextInt();

double s=sc.nextDouble();

hm.put(a,s);

}

System.out.println(main13.dis(hm));}}

import java.text.DecimalFormat;

import java.util.HashMap;

import java.util.Map;

class main13

{

public static double dis(HashMap<Integer,Double> h1)

{

double avg=0.0,sum=0.0;

int k=0;

for(Map.Entry m:h1.entrySet())

{

int a=(Integer)m.getKey();

if(a%2!=0)

{

Double d=(Double) m.getValue();

sum=sum+d;

k++;

}

}

avg = (double)sum/k;

DecimalFormat df = new DecimalFormat(".##");

String b1 = df.format(avg);

double b = Double.parseDouble(b1);

return b;}

}

21.22.

Calculate Electricity Bill (calculateElectricityBill)

ABC2012345

Given a method calculateElectricityBill() with three inputs. Write code to calculate the current bill.

Include a class UserMainCode with a static method calculateElectricityBill which accepts 3 inputs .The return type of the output should be an integer .

Create a class Main which would get the inputs and call the static method calculateElectricityBill present in the UserMainCode.

Input and Output Format:

Input consist of 3 integers.

First input is previous reading, second input is current reading and last input is per unit charge.

Reading Format - XXXXXAAAAA where XXXXX is consumer number and AAAAA is meter reading.

Output is a single integer corresponding to the current bill.

Refer sample output for formatting specifications.

Sample Input 1:

ABC2012345

ABC2012660

4

Sample Output 1:

1260

Sample Input 2:

ABCDE11111

ABCDE11222

3

Sample Output 2:

333

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

public static int meterReading(String input1, String input2, int input3)

{

int n1=Integer.parseInt(input1.substring(5, input1.length()));

int n2=Integer.parseInt(input2.substring(5, input2.length()));

int n=Math.abs((n2-n1)\*input3);

return n;

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

23.Character cleaning (removeCharacter) elephant/e

Write a program to input a String and a character, and remove that character from the given String. Print the final string.

Include a class UserMainCode with a static method removeCharacter which accepts a string and a character. The return type (string) should return the character cleaned string.

Create a Class Main which would be used to accept Input String and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string and a character.

Output consists of a string(the character cleaned string).

Refer sample output for formatting specifications.

Sample Input :

elephant

e

Sample Output:

lphant

public class PO {

public static void main(String[]args){

Scanner sc=new Scanner(System.in);

String name=sc.nextLine();

char ch=sc.nextLine().charAt(0);

StringBuffer sb=new StringBuffer(name);

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

{if(ch==sb.charAt(i))

{

sb.deleteCharAt(i);

i--;

}

}

System.out.print(sb.toString());}}

??????????????????????????????????????????????????????????????????????????????????????????????

import java.util.Scanner;

public class Main {

public static void main(String[]args){

Scanner sc=new Scanner(System.in);

String name=sc.nextLine();

char ch=sc.nextLine().charAt(0);

System.out.println(UserMainCode.getValues(name, ch));

}

}

public class UserMainCode{

public static String getValues(String name,char ch){

StringBuffer sb=new StringBuffer(name);

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

{if(ch==sb.charAt(i))

{

sb.deleteCharAt(i);

i--;

}

}

//System.out.print(sb.toString());

return sb.toString();}}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

24.Check Characters in a String (checkCharacters)the picture was great

Write a program to read a string and to test whether first and last character are same. The string is said to be be valid if the 1st and last character are the same. Else the string is said to be invalid.

Include a class UserMainCode with a static method checkCharacters which accepts a string as input .

The return type of this method is an int. Output should be 1 if the first character and last character are same . If they are different then return -1 as output.

Create a class Main which would get the input as a string and call the static method checkCharacters present in the UserMainCode.

Input and Output Format:

Input consists of a string.

Output is a string saying characters are same or not .

Refer sample output for formatting specifications.

Sample Input 1:

the picture was great

Sample Output 1:

Valid

Sample Input 1:

this

Sample Output 1:

Invalid

import java.util.StringTokenizer;

public class Main

{

public static void main(String[] args)

{

String input="this";

System.out.println(checkForFirstAndLastChar(input));

}

public static int checkForFirstAndLastChar(String input)

{

StringTokenizer t = new StringTokenizer(input," ");

String s = t.nextToken();

String s1 = " " ;

while(t.hasMoreTokens())

{

s1 = t.nextToken();

}

if(s.charAt(0) == s1.charAt(s1.length()-1))

return 1;

else

return 0;

}

}

??????????????????????????????????????????????????????????????????????????????????

import java.util.Scanner;

import java.util.StringTokenizer;

public class Main

{

public static void main(String[] args)

{

Scanner scan=new Scanner(System.in);

String s=scan.nextLine();

int z=Usercode.checkForFirstAndLastChar(s);

if(z==1){

System.out.println("Valid");

}

else{

System.out.println("Invalid");

}

}

}

import java.util.StringTokenizer;

public class Usercode{

public static int checkForFirstAndLastChar(String input)

{

StringTokenizer t = new StringTokenizer(input," ");

String s = t.nextToken();

String s1 = " " ;

while(t.hasMoreTokens())

{

s1 = t.nextToken();

}

if(s.charAt(0) == s1.charAt(s1.length()-1))

return 1;

else

return -1;

}

}

25.Check first and last word (check)

import java.util.HashMap;

import java.util.Iterator;

import java.util.Scanner;

public class main {

public static void main(String []args){

Scanner sc=new Scanner(System.in);

String age=sc.nextLine();

System.out.println(Usermaincode.display(age));

}

}

import java.util.StringTokenizer;

public class Usermaincode

{public static int display(String s)

{

int count=0;

String fin="";

StringTokenizer st=new StringTokenizer(s);

String ini=st.nextToken();

while(st.hasMoreTokens())

{ fin=st.nextToken();

}

if(ini.equals(fin))

count=ini.length();

else

count=ini.length()+fin.length();

return count;

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

26.Check Sum of Odd Digits (checkSum)56895/sum of odd digits is odd

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

27.28.Color Code ()

Write a program to read a number , calculate the sum of odd digits (values) present in the given number.

Include a class UserMainCode with a static method checkSum which accepts a positive integer . The return type should be 1 if the sum is odd . In case the sum is even return -1 as output.

Create a class Main which would get the input as a positive integer and call the static method checkSum present in the UserMainCode.

Input and Output Format:

Input consists of a positive integer n.

Refer sample output for formatting specifications.

Sample Input 1:

56895

Sample Output 1:

Sum of odd digits is odd.

Sample Input 2:

84228

Sample Output 2:

Sum of odd digits is even.

public class UserMainCode {

public static int SumOfOddsAndEvens(int n){

int n1,n2=0,n3;

while(n!=0)

{

n1=n%10;

if((n1%2)!=0)

n2+=n1;

n/=10;

}

if(n2%2==0)

n3=-1;

else

n3=1;

return n3;

}

public static void main(String[] args) {

int n=84882;

System.out.println(SumOfOddsAndEvens(n));

}

}

???????????????????????????????????????????????????????????????????????????????/

import java.util.Scanner;

public class Main{

public static void main(String[] args) {

Scanner scan=new Scanner(System.in);

int n=scan.nextInt();

System.out.println(Usercode.SumOfOddsAndEvens(n));

}

}

public class Usercode {

public static int SumOfOddsAndEvens(int n){

int n1,n2=0,n3;

while(n!=0)

{

n1=n%10;

if((n1%2)!=0)

n2+=n1;

n/=10;

}

if(n2%2==0)

n3=2;

else

n3=1;

return n3;

}

}

Write a program to read a string and validate whether the given string is a valid color code based on the following rules:

- Must start with "#" symbol

- Must contain six characters after #

- It may contain alphabets from A-F or digits from 0-9

Include a class UserMainCode with a static method validateColorCode which accepts a string. The return type (integer) should return 1 if the color is as per the rules else return -1.

Create a Class Main which would be used to accept a String and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Output consists of a string (Valid or Invalid).

Refer sample output for formatting specifications.

Sample Input 1:

#FF9922

Sample Output 1:

Valid

Sample Input 2:

#FF9(22

Sample Output 2:

Invalid

import java.util.\*;

public class B {

public static void main(String[] args) {

Scanner s=new Scanner(System.in);

String s1=s.next();

boolean b=colorCodeValidation(s1);

if(b==true)

System.out.println("valid color code");

else

System.out.println("invalid color code");

}

public static boolean colorCodeValidation(String s1) {

boolean b=false,b1=false;

String s2=s1.substring(1,s1.length());

if(s1.length()==7)

if(s1.charAt(0)=='#')

b1=true;

if(b1==true)

for(int i=0;i<s2.length();i++){

char c=s2.charAt(i);

if(c!='#'){

if((Character.isAlphabetic(c)&& Character.isUpperCase(c)) || Character.isDigit(c))

b=true;

else{

b=false;

break;}}}

return b;

}

29.Common Elements (sumCommonElements)

Write a program to find out sum of common elements in given two arrays. If no common elements are found print - ?No common elements?.

Include a class UserMainCode with a static method getSumOfIntersection which accepts two integer arrays and their sizes. The return type (integer) should return the sum of common elements.

Create a Class Main which would be used to accept 2 Input arrays and call the static method present in UserMainCode.

Input and Output Format:

Input consists of 2+m+n integers. The first integer corresponds to m (Size of the 1st array), the second integer corresponds to n (Size of the 2nd array), followed by m+n integers corresponding to the array elements.

Output consists of a single Integer corresponds to the sum of common elements or a string ?No common elements?.

Refer sample output for formatting specifications.

Assume the common element appears only once in each array.

Sample Input 1:

4

3

2

3

5

1

1

3

9

Sample Output 1:

4

Sample Input 2:

4

3

2

3

5

1

12

31

9

Sample Output 2:

No common elements

import java.util.Scanner;

public class Main {

public static void main(String [] args){

Scanner sc=new Scanner(System.in);

int n=Integer.parseInt(sc.nextLine());

int[] a=new int[n];

int[] b=new int[n];

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

a[i]=Integer.parseInt(sc.nextLine());

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

b[i]=Integer.parseInt(sc.nextLine());

System.out.println(UserMainCode.display(a,b));

}}

public class UserMainCode {

public static int display(int a[],int b[])

{

int sum=0;

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

{

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

{

if(a[i]==b[j])

sum=sum+a[i];

}}

if(sum==0)

return -1;

else

return sum;

}}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

30.Commons (countCommonStrings)

Given two arrays of strings,return the count of strings which is common in both arrays. Duplicate entries are counted only once.

Include a class UserMainCode with a static method countCommonStrings which accepts the string arrays. The return type is the integer formed based on rules.

Create a Class Main which would be used to accept the string arrays and integer and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a an integer indicating the number of elements in the string array followed by the array.

Output consists of a integer .

Refer sample output for formatting specifications.

Sample Input 1:

3

a

c

e

3

b

d

e

Sample Output 1:

1

Sample Input 2:

5

ba

ba

black

sheep

wool

5

ba

ba

have

any

wool

Sample Output 2:

2

public class kape {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int count=0;

int n1 = sc.nextInt();

String[] s1 = new String[n1];

for (int i = 0; i < n1; i++) {

s1[i] = sc.next();

}

int n2 = sc.nextInt();

String[] s2 = new String[n2];

for (int i = 0; i < n2; i++) {

s2[i] = sc.next();

}

ArrayList<String> al = new ArrayList<String>();

for (int i = 0; i < n1; i++) {

for (int j = 0; j < n2; j++) {

if(s1[i].equals(s2[j])){

if(!al.contains(s1[i])){

count++;

al.add(s1[i]);

}

}

}

}

System.out.println(count);

}

}

????????????????????????????????????????????????????????????????????????????????????????????????????

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int n1 = sc.nextInt();

String[] s1 = new String[n1];

for (int i = 0; i < n1; i++) {

s1[i] = sc.next();

}

int n2 = sc.nextInt();

String[] s2 = new String[n2];

for (int i = 0; i < n2; i++) {

s2[i] = sc.next();

}

System.out.println(UserMainCode.getvalues(s1,s2));

}

}

import java.util.ArrayList;

public class UserMainCode {

public static int getvalues (String s1[],String s2[]) {

int count=0;

ArrayList<String> al = new ArrayList<String>();

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

{

for (int j = 0; j < s2.length; j++)

{

if(s1[i].equals(s2[j]))

{

if(!al.contains(s1[i]))

{

count++;

al.add(s1[i]);

}

}

}

}

//System.out.println(count);

return count;

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

31.Concatenate Characters (concatCharacter)

Given an array of Strings, write a program to take the last character of each string and make a new String by concatenating it.

Include a class UserMainCode with a static method ?concatCharacter? that accepts a String array as input and returns the new String.

Create a class Main which would get the String array as input and call the static method concatCharacter present in the UserMainCode.

Input and Output Format:

The first line of the input consists of an integer n that corresponds to the number of strings in the input string array.

The next n lines of the input consist of the strings in the input string array.

Output consists of a string.

Sample Input:

3

ab

a

abcd

Sample Output:

bad

import java.util.Scanner;

public class kapes3 {

public static void main(String []args){

Scanner sc=new Scanner(System.in);

int s=Integer.parseInt(sc.nextLine());

String []a=new String[s];

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

{

a[i]=sc.nextLine();

}

System.out.println(kapes4.display(a));

}

}

public class kapes4

{public static String display(String[] a)

{

StringBuffer sb=new StringBuffer();

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

sb.append(a[i].charAt(a[i].length()-1));

return sb.toString();

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

32.Convert Format

Given a 10 digit positive number in the format XXX-XXX-XXXX as a string input, write a program to convert this number to the format XX-XX-XXX-XXX.

Include a class UserMainCode with a static method ?convertFormat? that accepts a String argument and returns a String.

Create a class Main which would get a String as input and call the static method convertFormat present in the UserMainCode.

Input and Output Format:

Input consists of a String.

Output consists of a String.

Sample Input:

555-666-1234

Sample Output:

55-56-661-234

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

import java.util.StringTokenizer;

public class Main {

public static void main(String[] args) {

String s="555-666-1234";

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

}

public static String display(String s) {

StringTokenizer t=new StringTokenizer(s,"-");

String s1=t.nextToken();

String s2=t.nextToken();

String s3=t.nextToken();

StringBuffer sb=new StringBuffer();

sb.append(s1.substring(0, s1.length()-1)).append('-');

sb.append(s1.charAt(s1.length()-1)).append(s2.charAt(0)).append('-');

sb.append(s2.substring(1, s2.length())).append(s3.charAt(0)).append('-');

sb.append(s3.substring(1, s3.length()));

return sb.toString();

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

33.Count Sequential Characters (countSequentialChars)abcxxxabc/1

Get a string as input and write code to count the number of characters which gets repeated 3 times consecutively and return that count (ignore case). If no character gets repeated 3 times consecutively return -1.

Include a class UserMainCode with a static method countSequentialChars which accepts a string as input.

The return type of the output is the repeat count.

Create a class Main which would get the input and call the static method countSequentialChars present in the UserMainCode.

Input and Output Format:

Input consists a string.

Output is an integer.

Refer sample output for formatting specifications.

Sample Input 1:

abcXXXabc

Sample Output 1:

1

Sample Input 2:

aaaxxyzAAAx

Sample Output 2:

2

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

public class Main {

public static void main(String[] args) {

String input1="aaxxyzAAx";

System.out.println(consecutiveRepeatitionOfChar(input1));

}

public static int consecutiveRepeatitionOfChar(String input1) {

int c=0;

int n=0;

for(int i=0;i<input1.length()-1;i++){

if(input1.charAt(i)==input1.charAt(i+1))

n++;

else

n=0;

if(n==2)

c++; }

return c;

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

34.Count Vowels (“countVowels”) avinash/3

Given a string input, write a program to find the total number of vowels in the given string.

Include a class UserMainCode with a static method ?countVowels? that accepts a String argument and returns an int that corresponds to the total number of vowels in the given string.

Create a class Main which would get the String as input and call the static method countVowels present in the UserMainCode.

`

Input and Output Format:

Input consists of a string.

Output consists of an integer..

Sample Input:

avinash

Sample Output:

3

public class Main {

public static void main(String[] args) {

String s1="avinash";

getvalues(s1);

}

public static void getvalues(String s1) {

String s2=s1.toLowerCase();

String s3="aeiou";

int count=0;

for(int i=0;i<s2.length();i++)

{

for(int j=0;j<s3.length();j++)

{

if(s2.charAt(i)==s3.charAt(j))

{

count++;

}

}

}

System.out.println(count);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

35.Dash Check

Write a program to read two strings and check whether or not they have dashes in the same places. Print ?Yes? if the condition satisfies, else print ?No?.

Include a class UserMainCode with a static method compareDashes which accepts two strings. The return type (Integer) should return 1 if all dashes are placed correctly, else return 2.

Create a Class Main which would be used to accept two strings and call the static method present in UserMainCode.

Note: The strings must have exactly the same number of dashes in exactly the same positions. The strings might be of different length.

Input and Output Format:

Input consists of two strings.

Output consists of a string (?Yes? or ?No?).

Refer sample output for formatting specifications.

Sample Input 1:

hi?there-you.

12--(134)-7539

Sample Output 1:

Yes

Sample Input 2:

-15-389

-xyw-zzy

Sample Output 2:

No

import java.util.ArrayList;

public class Main {

public static void main(String[] args) {

String s1="hi-there-you.";

String s2="12-(134)-7539";

getvalues(s1,s2);

}

public static void getvalues(String s1, String s2) {

ArrayList<Integer>l1=new ArrayList<Integer>

();

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

{

if(s1.charAt(i)=='-')

{

l1.add(i);

}

}

ArrayList<Integer>l2=new ArrayList<Integer>

();

for(int i=0;i<s2.length();i++)

{

if(s2.charAt(i)=='-')

{

l2.add(i);

}

}

//System.out.println(l1);

//System.out.println(l2);

if(l1.equals(l2))

{

System.out.println(1);

}

else

System.out.println(-1);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

36.Date Format Conversion

Given a date string in the format dd/mm/yyyy, write a program to convert the given date to the format dd-mm-yy.

Include a class UserMainCode with a static method ?convertDateFormat? that accepts a String and returns a String.

Create a class Main which would get a String as input and call the static method convertDateFormat present in the UserMainCode.

Input and Output Format:

Input consists of a String.

Output consists of a String.

Sample Input:

12/11/1998

Sample Output:

12-11-98

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Date;

public class Main {

public static void main(String[] args) {

String s1="12/12/1998";

getvalues(s1);

}

public static void getvalues(String s1) {

SimpleDateFormat sdf=new SimpleDateFormat("dd/MM/yyyy");

sdf.setLenient(false);

try {

Date d1=sdf.parse(s1);

SimpleDateFormat sdf1=new SimpleDateFormat("dd-MM-yy");

String s2=sdf1.format(d1);

System.out.println(s2);

} catch (ParseException e) {

e.printStackTrace();

}

}

}

???????????????????????????????????????????????????????????????????????????????????????????????????

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Date;

public class Main {

public static void main(String[] args) {

String s1="12/12/1998";

System.out.println(UserMainCode.getvalues(s1));

}

}

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Date;

public class UserMainCode{

public static String getvalues(String s1) {

SimpleDateFormat sdf=new SimpleDateFormat("dd/MM/yyyy");

sdf.setLenient(false);

try {

Date d1=sdf.parse(s1);

SimpleDateFormat sdf1=new SimpleDateFormat("dd-MM-yy");

String s2=sdf1.format(d1);

System.out.println(s2);

} catch (ParseException e) {

e.printStackTrace();

}

return s1;

}

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

37.Date Format

Write a program to read two String variables in DD-MM-YYYY.Compare the two dates and return the older date in 'MM/DD/YYYY' format.

Include a class UserMainCode with a static method findOldDate which accepts the string values. The return type is the string.

Create a Class Main which would be used to accept the two string values and call the static method present in UserMainCode.

Input and Output Format:

Input consists of two string.

Output consists of a string.

Refer sample output for formatting specifications.

Sample Input 1:

05-12-1987

8-11-2010

Sample Output 1:

12/05/1987

import java.text.SimpleDateFormat;

import java.util.Calendar;

import java.util.Date;

class Usercode

{

public static void getvalues(String s1,String s2)throws Exception

{

SimpleDateFormat sdf=new SimpleDateFormat("dd-MM-yyyy");

SimpleDateFormat sdf1=new SimpleDateFormat("dd/MM/yyyy");

Date d1=sdf.parse(s1);

Date d2=sdf.parse(s2);

Calendar cal=Calendar.getInstance();

cal.setTime(d1);

long y=cal.getTimeInMillis();

cal.setTime(d2);

long y1=cal.getTimeInMillis();

String s3=sdf1.format(d1);

String s4=sdf1.format(d2);

if(y<y1)

System.out.println(s3);

else

System.out.println(s4);

}

}

import java.util.\*;

public class Main {

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

{

Scanner sc=new Scanner(System.in);

String s1=sc.nextLine();

String s2=sc.nextLine();

Usercode.getvalues(s1,s2);

}

}

??????????????????????????????????????????????????????????????????????????????????

import java.util.\*;

public class Main {

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

{

Scanner sc=new Scanner(System.in);

String s1=sc.nextLine();

String s2=sc.nextLine();

System.out.println(UserMainCode.getvalues(s1,s2));

}

}

import java.text.SimpleDateFormat;

import java.util.Calendar;

import java.util.Date;

public class UserMainCode{

public static String getvalues(String s1,String s2)throws Exception

{ String s=null;

SimpleDateFormat sdf=new SimpleDateFormat("dd-MM-yyyy");

SimpleDateFormat sdf1=new SimpleDateFormat("dd/MM/yyyy");

Date d1=sdf.parse(s1);

Date d2=sdf.parse(s2);

Calendar cal=Calendar.getInstance();

cal.setTime(d1);

long y=cal.getTimeInMillis();

cal.setTime(d2);

long y1=cal.getTimeInMillis();

String s3=sdf1.format(d1);

String s4=sdf1.format(d2);

if(y<y1)

//System.out.println(s3);

s=s3;

else

//System.out.println(s4);

s=s4;

return s;

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

38.Date Validation (getValidDate)

Write a program to read a string representing a date. The date can be in any of the three formats

1:dd-MM-yyyy 2: dd/MM/yyyy 3: dd.MM.yyyy

If the date is valid, print valid else print invalid.

Include a class UserMainCode with a static method getValidDate which accepts a string. The return type (integer) should be based on the validity of the date.

Create a Class Main which would be used to accept Input string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Output consists of a string.

Refer sample output for formatting specifications.

Sample Input 1:

03.12.2013

Sample Output 1:

valid

Sample Input 2:

03$12$2013

Sample Output 3:

invalid

import java.util.ArrayList;

import java.util.Collections;

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Date;

public class Main {

public static void main(String[] args) {

String s="14#09/1991";

getvalues(s);

}

public static void getvalues(String s) {

if(s.matches("[0-9]{2}[.]{1}[0-9]{2}[.]{1}[0-9]{4}"))

{

SimpleDateFormat sdf=new SimpleDateFormat("dd.MM.yyyy");

sdf.setLenient(false);

try

{

Date d1=sdf.parse(s);

System.out.println(1);

} catch (ParseException e) {

System.out.println(-1);

}

}

else if(s.matches("[0-9]{2}[/]{1}[0-9]{2}[/][0-9]{4}"))

{

SimpleDateFormat sdf=new SimpleDateFormat("dd/MM/yyyy");

sdf.setLenient(false);

try

{

Date d1=sdf.parse(s);

System.out.println(1);

} catch (ParseException e) {

System.out.println(-1);

}

}

else if(s.matches("[0-9]{2}[-]{1}[0-9]{2}[-][0-9]{4}"))

{

SimpleDateFormat sdf=new SimpleDateFormat("dd-MM-yyyy");

sdf.setLenient(false);

try

{

Date d1=sdf.parse(s);

System.out.println(1);

} catch (ParseException e) {

System.out.println(-1);

}

}

else

System.out.println(-1);

}

}

???????????????????????????????????????????????????????????????????????????????????????????

import java.util.ArrayList;

import java.util.Collections;

import java.util.Scanner;

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Date;

public class Main {

public static void main(String[] args) {

Scanner scan=new Scanner(System.in);

String s=scan.nextLine();;

int z=UserMainCode.getvalues(s);

if(z==1){

System.out.println("Valid");

}

else

System.out.println("Invalid");

}

}

import java.util.ArrayList;

import java.util.Collections;

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Date;

public class UserMainCode {

public static int getvalues(String s) {

int res=0;

if(s.matches("[0-9]{2}[.]{1}[0-9]{2}[.]{1}[0-9]{4}"))

{

SimpleDateFormat sdf=new SimpleDateFormat("dd.MM.yyyy");

sdf.setLenient(false);

try

{

Date d1=sdf.parse(s);

//System.out.println(1);

res=1;

} catch (ParseException e) {

//System.out.println(-1);

res=-1;

}

}

else if(s.matches("[0-9]{2}[/]{1}[0-9]{2}[/][0-9]{4}"))

{

SimpleDateFormat sdf=new SimpleDateFormat("dd/MM/yyyy");

sdf.setLenient(false);

try

{

Date d1=sdf.parse(s);

//System.out.println(1);

res=1;

} catch (ParseException e) {

//System.out.println(-1);

res=-1;

}

}

else if(s.matches("[0-9]{2}[-]{1}[0-9]{2}[-][0-9]{4}"))

{

SimpleDateFormat sdf=new SimpleDateFormat("dd-MM-yyyy");

sdf.setLenient(false);

try

{

Date d1=sdf.parse(s);

//System.out.println(1);

res=1;

} catch (ParseException e) {

//System.out.println(-1);

res=-1;

}

}

else

//System.out.println(-1);

res=-1;

return res;

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 03.12.2013/VALID

39.Day of the Week

Write a program to read a date as string (MM-dd-yyyy) and return the day of week on that date.

Include a class UserMainCode with a static method getDay which accepts the string. The return type (string) should be the day of the week.

Create a Class Main which would be used to accept Input string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Output consists of a string.

Refer sample output for formatting specifications.

Sample Input 1:

07-13-2012

Sample Output 1:

Friday

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Date;

import java.util.Scanner;

public class Main {

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

Scanner sc=new Scanner(System.in);

String s1=sc.nextLine();

System.out.println(User.calculateBornDay(s1));

}

}

import java.text.SimpleDateFormat;

import java.text.ParseException;

import java.util.Date;

public class User {

public static String calculateBornDay(String s1) throws ParseException

{

SimpleDateFormat sdf=new SimpleDateFormat("MM-dd-yyyy");

SimpleDateFormat sdf1=new SimpleDateFormat("EEEEE");

Date d=sdf.parse(s1);

String s=sdf1.format(d);

return s;

}

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

40.Day of Week

Write a program to read a date as string (MM-dd-yyyy) and return the day of week on that date.

Include a class UserMainCode with a static method getDay which accepts the string. The return type (string) should be the day of the week.

Create a Class Main which would be used to accept Input string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Output consists of a string.

Refer sample output for formatting specifications.

Sample Input 1:

07-13-2012

Sample Output 1:

Friday

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Date;

import java.util.Scanner;

public class Main {

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

Scanner sc=new Scanner(System.in);

String s1=sc.nextLine();

System.out.println(User.calculateBornDay(s1));

}

}

import java.text.SimpleDateFormat;

import java.text.ParseException;

import java.util.Date;

public class User {

public static String calculateBornDay(String s1) throws ParseException

{

SimpleDateFormat sdf=new SimpleDateFormat("dd/MM/yyyy");

SimpleDateFormat sdf1=new SimpleDateFormat("EEEEE");

Date d=sdf.parse(s1);

String s=sdf1.format(d);

return s.toLowerCase();

}

}

?????????????????????????????????????????????????????????????????????????????????????????????????????

import java.text.ParseException;

import java.util.Scanner;

public class Main {

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

Scanner sc=new Scanner(System.in);

String s1=sc.nextLine();

System.out.println(UserMainCode.calculateBornDay(s1));

}

}

import java.text.SimpleDateFormat;

import java.text.ParseException;

import java.util.Date;

public class UserMainCode {

public static String calculateBornDay(String s1) throws ParseException

{

SimpleDateFormat sdf=new SimpleDateFormat("MM-dd-yyyy");

SimpleDateFormat sdf1=new SimpleDateFormat("EEEEE");

Date d=sdf.parse(s1);

String s=sdf1.format(d);

return s;

} }

41.Decimal to Binary Conversion (convertDecimalToBinary)

Write a Program that accepts a decimal number n, and converts the number to binary.

Include a class UserMainCode with a static method convertDecimalToBinary which accepts an integer. The return type is long representing the binary number.

Create a Class Main which would be used to accept the input integer and call the static method present in UserMainCode.

Input and Output Format:

Input consists of single integer.

Output consists of a single long.

Refer sample output for formatting specifications.

Sample Input 1:

5

Sample Output 1:

101

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 5/101

public class Main {

public static void main(String[] args) {

int n=8;

String s1=Integer.toBinaryString(n);

long y=Long.parseLong(s1);

System.out.println(y);

}

}

?????????????????????????????????????????????????????????????????????????????????????????

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner scan=new Scanner(System.in);

int n=scan.nextInt();

System.out.println(UserMainCode.binary(n));

}

}

public class UserMainCode {

public static long binary (int n)

{ String s1=Integer.toBinaryString(n);

long y=Long.parseLong(s1);

//System.out.println(y);

return y;

} }

42.Difference between largest and smallest elements in an array (getBigDiff)

Given a method taking an int array having size more than or equal to 1 as input. Write code to return the difference between the largest and smallest elements in the array. If there is only one element in the array return the same element as output.

Include a class UserMainCode with a static method getBigDiff which accepts a integer array as input.

The return type of the output is an integer which is the difference between the largest and smallest elements in the array.

Create a Main class which gets integer array as an input and call the static method getBigDiff present in the UserMainCode.

Input and Output Format:

Input is an integer array.First element in the input represents the number of elements in an array.

Size of the array must be >=1

Output is an integer which is the difference between the largest and smallest element in an array.

Sample Input 1:

4

3

6

2

1

Sample Output 1:

5

Sample Input 2:

4

5

3

7

2

Sample Output 2:

5

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 4/3/6/2/1

import java.util.Arrays;

public class kape1 {

public static int display(int []array)

{

Arrays.sort(array);

int n=array[array.length-1]-array[0];

int b=array.length;

if(b==1)

{

n=array[0];

}

return n;

}

}

?????????????????????????????????????????????????????????????????

import java.lang.reflect.Array;

import java.text.ParseException;

import java.util.Scanner;

public class Main{

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

Scanner scan=new Scanner(System.in);

int size=scan.nextInt();

int []s1=new int[size];

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

{

s1[i]=scan.nextInt();

}

System.out.println(Usercode.display(s1));

}

}

import java.util.Arrays;

public class Usercode {

public static int display(int []array)

{

Arrays.sort(array);

int n=array[array.length-1]-array[0];

int b=array.length;

if(b==1)

{

n=array[0];

}

return n;

}

}

43.Difference between two dates in days (getDateDifference)

Get two date strings as input and write code to find difference between two dates in days.

Include a class UserMainCode with a static method getDateDifference which accepts two date strings as input.

The return type of the output is an integer which returns the diffenece between two dates in days.

Create a class Main which would get the input and call the static method getDateDifference present in the UserMainCode.

Input and Output Format:

Input consists of two date strings.

Format of date : yyyy-mm-dd.

Output is an integer.

Refer sample output for formatting specifications.

Sample Input 1:

2012-03-12

2012-03-14

Sample Output 1:

2

Sample Input 2:

2012-04-25

2012-04-28

Sample Output 2:

3

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 2012-03-12/2012-03-14/2

import java.text.\*;

import java.util.\*;

public class Main {

public static int dateDifference(String s1,String s2) throws ParseException{

SimpleDateFormat sd=new SimpleDateFormat("yyyy-MM-dd");

Date d=sd.parse(s1);

Calendar c=Calendar.getInstance();

c.setTime(d);

long d1=c.getTimeInMillis();

d=sd.parse(s2);

c.setTime(d);

long d2=c.getTimeInMillis();

int n=Math.abs((int) ((d1-d2)/(1000\*3600\*24)));

return n;

}

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

String s1="2012-03-12";

String s2="2012-03-14";

System.out.println(dateDifference(s1,s2));

}

}

??????????????????????????????????????????????????????????????????????????????????????????

import java.text.\*;

import java.util.\*;

public class Main {

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

Scanner scan=new Scanner(System.in);

String s1=scan.nextLine();

String s2=scan.nextLine();

System.out.println(UserMainCode.dateDifference(s1,s2));

}

}

import java.text.\*;

import java.util.\*;

public class UserMainCode {

public static int dateDifference(String s1,String s2) throws ParseException{

SimpleDateFormat sd=new SimpleDateFormat("yyyy-MM-dd");

Date d=sd.parse(s1);

Calendar c=Calendar.getInstance();

c.setTime(d);

long d1=c.getTimeInMillis();

d=sd.parse(s2);

c.setTime(d);

long d2=c.getTimeInMillis();

int n=Math.abs((int) ((d1-d2)/(1000\*3600\*24)));

return n;

}

}

44.Digit Comparison (compareLastDigit)

Write a program to read two integers and return true if they have the same last digit.

Include a class UserMainCode with a static method compareLastDigit which accepts two integers and returns boolean. (true / false)

Create a Class Main which would be used to accept two integers and call the static method present in UserMainCode.

Input and Output Format:

Input consists of two integer.

Output consists TRUE / FALSE.

Refer sample output for formatting specifications.

Sample Input 1:

59

29

Sample Output 1:

TRUE

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 59/29/TRUE

import java.text.ParseException;

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc=new Scanner(System.in);

int a=sc.nextInt();

int b=sc.nextInt();

System.out.println(Usercode.characterCheck(a,b));

} }

import java.util.Date;

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Calendar;

class Usercode

{

public static boolean characterCheck(int q,int w) {

boolean b=false;

int c=q%10;

int d=w%10;

if(c==d)

b=true;

else

b=false;

return b;

}

}

???????????????????????????????????????????????????????????????????????????????????????????

import java.text.ParseException;

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc=new Scanner(System.in);

int a=sc.nextInt();

int b=sc.nextInt();

boolean z=UserMainCode.characterCheck(a,b);

if(z==true){

System.out.println("TRUE");

}

else

System.out.println("FALSE");

} }

import java.util.Date;

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Calendar;

class UserMainCode

{

public static boolean characterCheck(int q,int w) {

boolean b=false;

int c=q%10;

int d=w%10;

if(c==d)

b=true;

else

b=false;

return b;

}

}

45.Digits - II

Write a program to read a non-negative integer n, compute the sum of its digits. If sum is greater than 9 repeat the process and calculate the sum once again until the final sum comes to single digit.Return the single digit.

Include a class UserMainCode with a static method getDigitSum which accepts the integer value. The return type is integer.

Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a integer.

Output consists of integer.

Refer sample output for formatting specifications.

Sample Input 1:

9999

Sample Output 1:

9

Sample Input 2:

698

Sample Output 2:

5

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public class Main

{

public static void main(String[] args)

{

int n=698;

System.out.println(conversiontoaSingleDigit(n));

}

public static int conversiontoaSingleDigit(int n)

{

int sum = 0 ;

int n1=n;

while(n>10)

{

int a = 0 ; sum = 0;

while(n!=0)

{

a = n%10;

sum+=a;

n=n/10;

}

n=sum;

}

return sum;

}

}

46.Digits (countSeven)

Write a program to read a non-negative integer n, that returns the count of the occurances of 7 as digit.

Write a program to read a non-negative integer n, that returns the count of the occurances of 7 as digit.

Include a class UserMainCode with a static method countSeven which accepts the integer value. The return type is integer which is the count value.

Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a integer.

Output consists of integer.

Refer sample output for formatting specifications.

Sample Input 1:

717

Sample Output 1:

2

Sample Input 2:

4534

Sample Output 2:

0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 717/2

public class UserMainCode

{

public static int display(int n)

{

int num=0,count=0;

while(n!=0)

{

num=n%10;

if(num==7)

{

count++;

}

n=n/10;

}

return count;

}

}

47.Discount Rate Calculation

Write a program to calculate discount of the acccount holders based on the transaction amount and registration date using below mentioned prototype:

1. Read account details from the User. The details would include id, DOR (date of registration) and transaction amount in the given order. The datatype for id is string, DOR is string and transaction amount is integer.

2. You decide to build two hashmaps. The first hashmap contains employee id as key and DOR as value, and the second hashmap contains same employee ids as key and amount as value.

3. Discount Amount as on 01/01/2015:

a. If the transaction amount greater than or equal to 20000 and registration greater than or equal to 5 year then discount rate is 20% of transaction amount.

b. If the transaction amount greater than or equal to 20000 and registration less then to 5 year then discount rate is 10% of transaction amount.

c. If the transaction amount less than to 20000 and registration greater than or equal to 5 year then discount rate is 15% of transaction amount.

d. If the transaction amount less than to 20000 and registration less then to 5 year then discount rate is 5% of transaction amount.

4. You decide to write a function calculateDiscount which takes the above hashmaps as input and returns the treemap as output. Include this function in class UserMainCode.

Create a Class Main which would be used to read employee details in step 1 and build the two hashmaps. Call the static method present in UserMainCode.

Input and Output Format:

Input consists of transaction details. The first number indicates the size of the employees. The next three values indicate the user id, user DOR and transaction amount. The DOR (Date of Registration) format is ?dd-mm-yyyy?

Output consists of a string which has the user id and discount amount one in a line for each user.

Refer sample output for formatting specifications.

Sample Input 1:

4

A-1010

20-11-2007

25000

B-1011

04-12-2010

30000

C-1012

11-11-2005

15000

D-1013

02-12-2012

10000

Sample Output 1:

A-1010:5000

B-1011:3000

C-1012:2250

D-1013:500

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.HashMap;

import java.util.Iterator;

import java.util.TreeMap;

import java.util.Scanner;

public class main {

public static void main(String []args){

Scanner sc=new Scanner(System.in);

int s=Integer.parseInt(sc.nextLine());

HashMap<String,String>hm=new HashMap<String,String>();

HashMap<String,Integer>hm1=new HashMap<String,Integer>();

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

{

String id=sc.nextLine();

hm.put(id, sc.nextLine());

hm1.put(id,Integer.parseInt(sc.nextLine()));

}

TreeMap<String,Integer>tm=new TreeMap<String,Integer>();

tm=Usermaincode.display(hm,hm1);

Iterator<String> it=tm.keySet().iterator();

while(it.hasNext())

{

String n=it.next();

int fac=tm.get(n);

System.out.println(n+":"+fac);

}

}

}

import java.text.SimpleDateFormat;

import java.util.Date;

import java.util.HashMap;

import java.util.Iterator;

import java.util.HashMap;

import java.util.TreeMap;

public class Usermaincode

{

public static TreeMap<String,Integer> display (HashMap<String,String>hm,HashMap<String,Integer>hm1)

{

int year=0,amount=0;

double dis=0;

String now="01/01/2015";

TreeMap<String,Integer>tm=new TreeMap<String,Integer>();

Iterator<String> it=hm.keySet().iterator();

while(it.hasNext())

{

String id=it.next();

String dor=hm.get(id);

amount=hm1.get(id);

SimpleDateFormat sdf=new SimpleDateFormat("dd-MM-yyyy");

SimpleDateFormat sdf1=new SimpleDateFormat("dd/MM/yyyy");

try

{

Date d=sdf.parse(dor);

Date d1=sdf1.parse(now);

sdf.setLenient(false);

int y=d.getYear();

int y1=d1.getYear();

int m=d.getMonth();

int m1=d1.getMonth();

int day=d.getDay();

int day1=d1.getDay();

year=y1-y;

if(m>m1)

year--;

else if(m==m1)

{if(day<day1)

year--;

}

if(year>=5 && amount>=20000)

dis=0.2\*amount;

else if(year<5 && amount>=20000)

dis=0.1\*amount;

else if(year>=5 && amount<20000)

dis=0.15\*amount;

else

dis=0.05\*amount;

tm.put(id,(int)dis);

}

catch(Exception e)

{

e.printStackTrace();

}

}

return tm;

}

}

48.DOB - Validation(ValidateDOB)

Write a program to validate the Date of Birth given as input in String format (MM/dd/yyyy) as per the validation rules given below. Return true for valid dates else return false.

1. Value should not be null

2. month should be between 1-12, date should be between 1-31 and year should be a four digit number.

Include a class UserMainCode with a static method ValidateDOB which accepts the string. The return type is TRUE / FALSE.

Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Output consists of TRUE / FALSE.

Refer sample output for formatting specifications.

Sample Input 1:

12/23/1985

Sample Output 1:

TRUE

Sample Input 2:

31/12/1985

Sample Output 2:

FALSE

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.text.SimpleDateFormat;

import java.util.Date;

import java.util.Scanner;

public class UserMainCode {

public static void main(String[] args)

{

String str=new String();

Scanner sc=new Scanner(System.in);

str=sc.nextLine();

SimpleDateFormat sdf=new SimpleDateFormat("MM/dd/yyyy");

sdf.setLenient(false);

try

{

Date d1=sdf.parse(str);

System.out.println("TRUE");

}

catch(Exception e)

{

System.out.println("FALSE");

}

}

}

???????????????????????????????????????????????????????????????????????

import java.text.SimpleDateFormat;

import java.util.Date;

import java.util.Scanner;

public class Main {

public static void main(String[] args)

{

String str=new String();

Scanner sc=new Scanner(System.in);

str=sc.nextLine();

boolean z=Usercode.DobValidate(str);

if(z==true)

{

System.out.println("TRUE");

}

else

{

System.out.println("FALSE");

}

}}

import java.text.SimpleDateFormat;

import java.util.Date;

import java.util.Scanner;

public class Usercode {

public static Boolean DobValidate(String s1)

{ boolean b=false;

SimpleDateFormat sdf=new SimpleDateFormat("MM/dd/yyyy");

sdf.setLenient(false);

try

{

Date d1=sdf.parse(s1);

//System.out.println("TRUE");

b=true;

}

catch(Exception e)

{

//System.out.println("FALSE");

b=false;

}

return b;

}

}

49.Duplicate Characters

Write a Program which removes duplicate characters from the string. Your program should read a sentence (string) as input from user and return a string removing duplicate characters. Retain the first occurance of the duplicate character. Assume the characters are case ? sensitive.

Include a class UserMainCode with a static method removeDuplicates which accepts a string. The return type is the modified sentence of type string.

Create a Class Main which would be used to accept the input string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string with maximum size of 100 characters.

Output consists of a single string.

Refer sample output for formatting specifications.

Sample Input 1:

hi this is sample test

Sample Output 1:

hi tsample

Sample Input 2:

ABC DEF

Sample Output 2:

ABC DEF

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.HashSet;

import java.util.Iterator;

import java.util.LinkedHashSet;

import java.util.StringTokenizer;

public class Main {

public static void main(String[] args) {

String s1="hi this is sample test";

getvalues(s1);

}

public static void getvalues(String s1) {

char a[]=s1.toCharArray();

StringBuffer sb=new StringBuffer();

LinkedHashSet<Character>hs=new LinkedHashSet<Character>();

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

{

hs.add(a[i]);

}

Iterator<Character>itr=hs.iterator();

while(itr.hasNext())

{

char o=itr.next();

if(o!=' ');

{

sb.append(o);

}

}

System.out.println(sb);

}}

????????????????????????????????????????????????????????????????????????????///

import java.util.HashSet;

import java.util.Iterator;

import java.util.LinkedHashSet;

import java.util.Scanner;

import java.util.StringTokenizer;

public class Main {

public static void main(String[] args) {

Scanner scan = new Scanner(System.in);

String s1=scan.nextLine();

Usercode.getvalues(s1);

}

}

import java.util.HashSet;

import java.util.Iterator;

import java.util.LinkedHashSet;

import java.util.StringTokenizer;

public class Usercode {

public static String getvalues(String s1) {

char a[]=s1.toCharArray();

StringBuffer sb=new StringBuffer();

LinkedHashSet<Character>hs=new LinkedHashSet<Character>();

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

{

hs.add(a[i]);

}

Iterator<Character>itr=hs.iterator();

while(itr.hasNext())

{

char o=itr.next();

if(o!=' ');

{

sb.append(o);

}

}

System.out.println(sb);

return s1;

}}

50.Duplicates (getDistinctSum)

GIven three integers (a,b,c) find the sum. However, if one of the values is the same as another, both the numbers do not count towards the sum and the third number is returned as the sum.

Include a class UserMainCode with a static method getDistinctSum which accepts three integers and returns integer.

Create a Class Main which would be used to accept three integers and call the static method present in UserMainCode.

Input and Output Format:

Input consists of three integers.

Output consists of a integer.

Refer sample output for formatting specifications.

Sample Input 1:

1

2

1

Sample Output 1:

2

Sample Input 2:

1

2

3

Sample Output 2:

6

Close

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1/2/1/2

import java.util.\*;

public class User {

public static void main(String[] args) {

int a=1;

int b=1;

int c=1;

System.out.println(characterCheck(a,b,c));

}

public static int characterCheck(int q,int w,int e) {

int s=0;

if(q!=w&&q!=e)

{

s=q+w+e;

}

else if(q==w&&q==e)

{

s=0;

}

else if(q==w&&q!=e)

{

s=e;

}

else if(q!=w&&q==e)

{

s=w;

}

return s;

}

}

50.Duplicates (getDistinctSum)

GIven three integers (a,b,c) find the sum. However, if one of the values is the same as another, both the numbers do not count towards the sum and the third number is returned as the sum.

Include a class UserMainCode with a static method getDistinctSum which accepts three integers and returns integer.

Create a Class Main which would be used to accept three integers and call the static method present in UserMainCode.

Input and Output Format:

Input consists of three integers.

Output consists of a integer.

Refer sample output for formatting specifications.

Sample Input 1:

1

2

1

Sample Output 1:

2

Sample Input 2:

1

2

3

Sample Output 2:

6

Close

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1/2/1/2

import java.util.\*;

public class User {

public static void main(String[] args) {

int a=1;

int b=1;

int c=1;

System.out.println(characterCheck(a,b,c));

}

public static int characterCheck(int q,int w,int e) {

int s=0;

if(q!=w&&q!=e)

{

s=q+w+e;

}

else if(q==w&&q==e)

{

s=0;

}

else if(q==w&&q!=e)

{

s=e;

}

else if(q!=w&&q==e)

{

s=w;

}

return s;

}

}

51.Elements in ArrayList(arrayListSubtractor)

Use Collection Methods.

Write a program that takes two ArrayLists as input and finds out all elements present either in A or B, but not in both.

Include a class UserMainCode with the static method arrayListSubtractor which accepts the two arraylists and returns an array.

Create a Class Main which would be used to read the inputs and call the static method present in UserMainCode.

Input and Output Format:

Input consists of an integer (m) denoting the size of first arraylist. The next m elements would be the values of the first arraylist. The next input would be n denoting the size of the second arraylist. The next n elements would be the values of the second arraylist.

Output consists of an array. The elements in the output array need to be printed in sorted order.

Refer sample output for formatting specifications.

Sample Input 1:

4

1

8

3

5

2

3

5

Sample Output 1:

1

8

Sample Input 2:

4

9

1

3

5

4

1

3

5

6

Sample Output 2:

6

9

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.\*;

public class Main

{

public static void main(String[] args)

{

int n,m;

Scanner sin = new Scanner(System.in);

n = sin.nextInt();

ArrayList<Integer> a1 = new ArrayList<Integer>(n);

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

{

int k = sin.nextInt();

a1.add(k);

}

m = sin.nextInt();

ArrayList<Integer> a2 = new ArrayList<Integer>(m);

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

{

int k = sin.nextInt();

a2.add(k);

}

int[] result = UserMainCode.arrayListSubtractor(a1,a2);

Arrays.sort(result);

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

System.out.println(result[i]);

}

}

import java.util.\*;

public class UserMainCode

{

public static int[] arrayListSubtractor(ArrayList<Integer> arrlist1,ArrayList<Integer> arrlist2)

{

int count=0,key;

int max = arrlist1.size();

if(arrlist1.size() < arrlist2.size())

max = arrlist2.size();

ArrayList<Integer> temp = new ArrayList<Integer>(max);

for(int i=0;i<arrlist1.size();i++)

{

key = (int)arrlist1.get(i);

if(arrlist2.indexOf(key) == -1)

{

++count;

temp.add(key);

}

}

for(int i=0;i<arrlist2.size();i++)

{

key = (int)arrlist2.get(i);

if(arrlist1.indexOf(key) == -1)

{

if(!temp.contains(key))

{

++count;

temp.add(key);

}

}

}

int[] result = new int[count];

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

result[i] = (int)temp.get(i);

return result;

}

}

52.E-Mail Validation (ValidateEmail)

Write a program to read a string and validate the given email-id as input.

Validation Rules:

1. Ensure that there are atleast 5 characters between '@' and '.'

2. There should be only one '.' and one '@' symbol.

3. The '.' should be after the '@' symbol.

4. There must be atleast three characters before '@'.

5. The string after '.' should only be 'com'

Include a class UserMainCode with a static method ValidateEmail which accepts the string. The return type is TRUE / FALSE as per problem.

Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Output consists of TRUE / FALSE.

Refer sample output for formatting specifications.

Sample Input 1:

test@gmail.com

Sample Output 1:

TRUE

Sample Input 2:

academy@xyz.com

Sample Output 2:

FALSE

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.\*;

public class Main {

public static void main(String[] args) {

Scanner scan=new Scanner(System.in);

String ip=scan.nextLine();

boolean b=UserMainCode.emailIdValidation(ip);

if(b==true)

System.out.println("valid mail Id");

else

System.out.println("not a valid Id");

}}

import java.util.StringTokenizer;

public class UserMainCode{

public static boolean emailIdValidation(String ip) {

int i=0;

boolean b=false;

StringTokenizer t=new StringTokenizer(ip,"@");

String s1=t.nextToken();

String s2=t.nextToken();

StringTokenizer t1=new StringTokenizer(s2,".");

String s3=t1.nextToken();

String s4=t1.nextToken();

if(ip.contains("@") && ip.contains("."))

i++;

if(i==1)

if(s3.length()==5)

if(s1.length()>=3)

if(s4.equals("com"))

b=true;

return b;

}

}

53.Employee Bonus

A Company wants to give away bonus to its employees. You have been assigned as the programmer to automate this process. You would like to showcase your skills by creating a quick prototype. The prototype consists of the following steps:

Read Employee details from the User. The details would include id, DOB (date of birth) and salary in the given order. The datatype for id is integer, DOB is string and salary is integer.

You decide to build two hashmaps. The first hashmap contains employee id as key and DOB as value, and the second hashmap contains same employee ids as key and salary as value.

If the age of the employee in the range of 25 to 30 years (inclusive), the employee should get bonus of 20% of his salary and in the range of 31 to 60 years (inclusive) should get 30% of his salary. store the result in TreeMap in which Employee ID as key and revised salary as value. Assume the age is caculated based on the date 01-09-2014. (Typecast the bonus to integer).

Other Rules:

a. If Salary is less than 5000 store -100.

b. If the age is less than 25 or greater than 60 store -200.

c. a takes more priority than b i.e both if a and b are true then store -100.

You decide to write a function calculateRevisedSalary which takes the above hashmaps as input and returns the treemap as output. Include this function in class UserMainCode.

Create a Class Main which would be used to read employee details in step 1 and build the two hashmaps. Call the static method present in UserMainCode.

Input and Output Format:

Input consists of employee details. The first number indicates the size of the employees. The next three values indicate the employee id, employee DOB and employee salary. The Employee DOB format is ?dd-mm-yyyy?

Output consists of a single string.

Refer sample output for formatting specifications.

Sample Input 1:

2

1010

20-12-1987

10000

2020

01-01-1985

14400

Sample Output 1:

1010

12000

2020

17280

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.HashMap;

import java.util.Iterator;

import java.util.Scanner;

import java.util.TreeMap;

public class Main

{

public static void main(String args[])

{

HashMap<Integer,String>hm=new HashMap<Integer,String>();

HashMap<Integer,Integer>hm1=new HashMap<Integer,Integer>();

TreeMap<Integer,Integer>hm2=new TreeMap<Integer,Integer>();

Scanner sc=new Scanner(System.in);

int s=Integer.parseInt(sc.nextLine());

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

{

int id=Integer.parseInt(sc.nextLine());

String ss=sc.nextLine();

int salary=Integer.parseInt(sc.nextLine());

hm.put(id,ss);

hm1.put(id,salary);

}

hm2=UserMainCode.display(hm,hm1);

Iterator<Integer> it=hm2.keySet().iterator();

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

{

int id=it.next();

int bonus=hm2.get(id);

System.out.println(id);

System.out.println(bonus);

}

}}

import java.text.SimpleDateFormat;

import java.util.Calendar;

import java.util.Date;

import java.util.HashMap;

import java.util.Iterator;

import java.util.TreeMap;

public class UserMainCode

{ public static TreeMap<Integer,Integer> display(HashMap<Integer,String>hm,HashMap<Integer,Integer>hm1)

{

TreeMap<Integer,Integer>hm2=new TreeMap<Integer,Integer>();

Iterator<Integer> it=hm.keySet().iterator();

while(it.hasNext())

{

int y=it.next();

String dob=hm.get(y);

SimpleDateFormat sdf=new SimpleDateFormat("dd-MM-yyyy");

sdf.setLenient(false);

String now="01-09-2014";

try

{ Date d=sdf.parse(now);

Date d1=sdf.parse(dob);

int y1=d.getYear();

int y2=d1.getYear();

int m1=d.getMonth();

int m2=d1.getMonth();

int day1=d.getDay();

int day2= d1.getDay();

int age=y1-y2;

if(m1<m2)

age--;

else if(m1==m2 && day1<day2)

age--;

if(age>=25 && age<=30)

{float bonus=(float)0.2\*hm1.get(y)+hm1.get(y);

hm2.put(y,(int)bonus );

}

else if(age>30 && age<=60)

{ float bonus=(float) (0.3\*hm1.get(y))+hm1.get(y);

hm2.put(y,(int)bonus );

}

else if(age<25 || age>60)

hm2.put(y,-200);

else

hm2.put(y,-100);

}

catch(Exception e)

{e.printStackTrace();

}}

return hm2;

}

54.Employees & Designations(obtainDesignation)

A Company wants to obtain employees of a particular designation. You have been assigned as the programmer to build this package. You would like to showcase your skills by creating a quick prototype. The prototype consists of the following steps:

Read Employee details from the User. The details would include name and designaton in the given order. The datatype for name and designation is string.

Build a hashmap which contains the name as key and designation as value.

You decide to write a function obtainDesignation which takes the hashmap and designation as input and returns a string List of employee names who belong to that designation as output. Include this function in class UserMainCode. Display employee name's in ascending order.

Create a Class Main which would be used to read employee details in step 1 and build the hashmap. Call the static method present in UserMainCode.

Input and Output Format:

Input consists of employee details. The first number indicates the size of the employees. The next two values indicate the employee name employee designation. The last string would be the designation to be searched.

Output consists of a array values containing employee names.

Refer sample output for formatting specifications.

Sample Input 1:

4

Manish

MGR

Babu

CLK

Rohit

MGR

Viru

PGR

MGR

Sample Output 1:

Manish

Rohit

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.Iterator;

import java.util.LinkedHashMap;

import java.util.Scanner;

public class Main {

public static void main(String[] args)

{

Scanner sc=new Scanner(System.in);

int k1=Integer.parseInt(sc.nextLine());

LinkedHashMap<String,String> hm=new LinkedHashMap<String,String>();

for(int i=0;i<k1;i++)

{

String k=sc.nextLine();

String s=sc.nextLine();

hm.put(k,s);

}

String n=sc.nextLine();

LinkedHashMap<String,String> hm1=new LinkedHashMap<String,String>();

hm1=UserMainCode.dis(hm,n);

Iterator<String> it=hm1.keySet().iterator();

while(it.hasNext())

{

String s2=it.next();

System.out.println(s2);

}

}

}

import java.util.HashMap;

import java.util.Iterator;

import java.util.LinkedHashMap;

import java.util.Map;

import java.util.Scanner;

public class UserMainCode

{

public static LinkedHashMap<String,String> dis(LinkedHashMap<String,String> h1,String n)

{

int k=0;

LinkedHashMap<String,String> hm1=new LinkedHashMap<String,String>();

Iterator<String>it=h1.keySet().iterator();

while(it.hasNext())

{

String s2=it.next();

String s3=h1.get(s2);

if(s3.equals(n))

hm1.put(s2,s3);

}

return hm1;

}}

55.Even and Odd Index Sum

Write a program that accepts a positive number as input and calculates the sum of digits at even indexes (say evenSum) and sum of digits at odd indexes (say oddSum) in the given number. If both the sums are equal , print 'yes', else print no.

Example:

input = 23050

evenSum = 2 + 0 + 0 = 2

oddSum = 3 + 5 = 8

output = no

Include a class UserMainCode with a static method ?sumOfOddEvenPositioned? that accepts an integer and returns an integer. The method returns 1 if the 2 sums are equal. Else the method returns -1.

Create a class Main which would get an integer as input and call the static method sumOfOddEvenPositioned present in the UserMainCode.

Input and Output Format:

Input consists of an integer.

Output consists of a string that is either ?yes? or ?no?.

Sample Input 1:

23050

Sample Output 1:

no

Sample Input 2:

231

Sample Output 2:

yes

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

int res=UserMainCode.sumOfOddEvenPositioned(n);

if(res==1)

System.out.println("yes");

else

System.out.println("no");

}

}

public class UserMainCode

{

public static int sumOfOddEvenPositioned(int n) {

int rem = 0, i = 0;

int a[] = new int[10];

while (n > 0) {

rem = n % 10;

a[i] = rem;

n = n / 10;

i++;

}

int sume = 0, sumo = 0;

for (int j = i - 1; j >= 0; j -= 2) {

sumo = sumo + a[j];

}

for (int j = i - 2; j >= 0; j -= 2) {

sume = sume + a[j];

}

if (sume == sumo) {

return 1;

} else

return -1;

}

}

56.Even Sum & Duplicate Elements(sumElements)

Write a program to read a integer array, Remove the duplicate elements and display sum of even numbers in the output. If input array contain only odd number then return -1.

Include a class UserMainCode with a static method sumElements which accepts the integer array. The return type is integer.

Create a Class Main which would be used to accept the integer array and call the static method present in UserMainCode.

Input and Output Format:

Input consists of an integer n which is the number of elements followed by n integer values.

Output consists of integer.

Refer sample output for formatting specifications.

Sample Input 1:

7

2

3

54

1

6

7

7

Sample Output 1:

62

Sample Input 2:

6

3

7

9

13

17

21

Sample Output 2:

-1

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.HashMap;

import java.util.LinkedHashMap;

import java.util.LinkedHashSet;

import java.util.Scanner;

public class Main

{

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int n=sc.nextInt();

int a[]=new int[n];

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

{

a[i]=sc.nextInt();

}

System.out.println(UserMainCode.display(a));

}}

import java.util.Iterator;

import java.util.LinkedHashSet;

public class UserMainCode {

public static int display(int a[])

{

LinkedHashSet<Integer>h1=new LinkedHashSet<Integer>();

int s=0;

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

{

h1.add(a[i]);

}

Iterator<Integer> it=h1.iterator();

while(it.hasNext())

{

int k=it.next();

if(k%2==0)

{

s=s+k;

}

}

if(s>0)

return s;

else

return -1;

}}

57.Experience Calculator (calculateExperience)

Write a program to read Date of Joining and current date as Strings and Experience as integer and validate whether the given experience and calculated experience are the same. Print ?true? if same, else ?false?.

Include a class UserMainCode with a static method calculateExperience which accepts 2 strings and an integer. The return type is boolean.

Create a Class Main which would be used to accept 2 string (dates) and an integer and call the static method present in UserMainCode.

Input and Output Format:

Input consists of 2 strings and an integer, where the 2 strings corresponds to the date of joining and current date, and the integer is the experience.

Output is either ?true? or ?false?.

Refer sample output for formatting specifications.

Sample Input 1:

11/01/2010

01/09/2014

4

Sample Output 1:

true

Sample Input 2:

11/06/2009

01/09/2014

4

Sample Output 2:

false

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*11/01/2010/01/09/2014/4

import java.util.Date;

import java.text.SimpleDateFormat;

public class Usermaincode

{public static boolean display(String s,String s1,int n)

{

boolean b=false;

SimpleDateFormat sdf=new SimpleDateFormat("dd/MM/yyyy");

try{

Date d=sdf.parse(s);

Date d1=sdf.parse(s1);

int y=d.getYear();

int y1=d1.getYear();

int m=d.getMonth();

int m1=d1.getMonth();

int day=d.getDay();

int day1=d1.getDay();

int age=y1-y;

if(m>m1)

age--;

else if(m==m1)

{if(day<day1)

age--;

}

if(age==n)

b=true;

else

b=false;

}

catch(Exception e)

{e.printStackTrace();

}

return b;

}

}

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

String s=sc.nextLine();

String s1=sc.nextLine();

int n = sc.nextInt();

System.out.println(UserMainCode.validateExp(s,s1,n));

}

}

58.Experience Validator(validateExp)

Write a program to valiadate the experience of an employee.

An employee who has recently joined the organization provides his year of passing and total number of years of experience in String format. Write code to validate his experience against the current date.

1) Input consists of two String first represent the year of passed out and the second string reperesent the year of experience.

2) create a function with name validateExp which accepts two string as input and boolean as output.

3) The difference between current year and year of pass should be more than or equal to Experience

Return true if all condition are true.

Note:Conside 2015 as the current year.

Include a class UserMainCode with the static function validateExp

Create a Class Main which would be used to accept the boolean and call the static method present in UserMainCode.

Input and Output Formate:

Input consists of two Strings.

output will display true if the given data are correct.

Sample Input:

2001

5

Sample Output:

TRUE

Close

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.ArrayList;

import java.util.HashMap;

import java.util.Scanner;

public class Main {

public static void main(String args[]){

Scanner sc = new Scanner(System.in);

String s=sc.nextLine();

String s1=sc.nextLine();

System.out.print(UserMainCode.validateExp(s,s1));

}

}

import java.util.Calendar;

import java.util.Date;

public class UserMainCode {

public static boolean validateExp(String s,String s1)

{

int y1=Integer.parseInt(s);

Date d=new Date();

Calendar c=Calendar.getInstance();

int y2=c.get(Calendar.YEAR);

int y=Math.abs(y1-y2);

int e=Integer.parseInt(s1);

if(y>=e)

return true;

else

return false;

}}

59.Fetching Middle Characters from String (getMiddleChars)

Write a program to read a string of even length and to fetch two middle most characters from the input string and return it as string output.

Include a class UserMainCode with a static method getMiddleChars which accepts a string of even length as input . The return type is a string which should be the middle characters of the string.

Create a class Main which would get the input as a string and call the static method getMiddleChars present in the UserMainCode.

Input and Output Format:

Input consists of a string of even length.

Output is a string .

Refer sample output for formatting specifications.

Sample Input 1:

this

Sample Output 1:

hi

Sample Input 1:

Hell

Sample Output 1:

el

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* this/hi

public class UserMainCode {

public static String getMiddleChars(String str)

{

StringBuffer sb=new StringBuffer();

if(str.length()%2==0)

{

sb.append(str.substring((str.length()/2)-1,(str.length()/2)+1));

}

return sb.toString();

}

}

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

String s=sc.nextLine();

System.out.println(UserMainCode.getMiddleChars(s));

}

}

60.Fibonacci Sum

Write a program to read an integer n, generate fibonacci series and calculate the sum of first n numbers in the series. Print the sum.

Include a class UserMainCode with a static method getSumOfNfibos which accepts an integer n. The return type (Integer) should return the sum of n fibonacci numbers.

Create a Class Main which would be used to accept an integer and call the static method present in UserMainCode.

Note: First two numbers in a Fibonacci series are 0, 1 and all other subsequent numbers are sum of its previous two numbers. Example - 0, 1, 1, 2, 3, 5...

Input and Output Format:

Input consists of an integer, which corresponds to n.

Output consists of an Integer (sum of fibonacci numbers).

Refer sample output for formatting specifications.

Sample Input :

5

Sample Output :

7

[0 + 1 + 1 + 2 + 3 = 7]

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public class UserMainCode {

public static int sumOfFibonacci(int n){

int a=0,b=1,c=0,d=1;

for(int i=3;i<=n;i++){

c=a+b;

a=b; b=c;

d=d+c;

}

return d;

}

}

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int s=sc.nextInt();

System.out.println(UserMainCode.sumOfFibonacci(s));

}

}

61.File Extension (fileIdentifier) sun.gif

Write a program to read a file name as a string and find out the file extension and return it as output. For example, the file sun.gif has the extension gif.

Include a class UserMainCode with a static method fileIdentifier which accepts a string. The return type (string) should return the extension of the input string (filename).

Create a Class Main which would be used to accept Input String and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string that corresponds to a file name.

Output consists of a string(extension of the input string (filename)).

Refer sample output for formatting specifications.

Sample Input 1:

sun.gif

Sample Output 1:

gif

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

String s=sc.nextLine();

System.out.println(UserMainCode.extensionString(s));

}

}

import java.util.StringTokenizer;

public class UserMainCode {

public static String extensionString(String s1){

StringTokenizer t=new StringTokenizer(s1,".");

t.nextToken();

String s2=t.nextToken();

return s2;

}

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

62.Find common characters and unique characters in string (commonChars)a black cow/battle ship/2

Given a method with two strings as input. Write code to count the common and unique letters in the two strings.

Note:

- Space should not be counted as a letter.

- Consider letters to be case sensitive. ie, "a" is not equal to "A".

Include a class UserMainCode with a static method commonChars which accepts two strings as input.

The return type of the output is the count of all common and unique characters in the two strings.

Create a class Main which would get the inputs and call the static method commonChars present in the UserMainCode.

Input and Output Format:

Input consists of two strings.

Output is an integer.

Refer sample output for formatting specifications.

Sample Input 1:

a black cow

battle ship

Sample Output 1:

2

[Explanation : b, l and a are the common letters between the 2 input strings. But 'a' appears more than once in the 1st string. So 'a' should not be considered while computing the count value.]

Sample Input 2:

australia

sri lanka

Sample Output 2:

4

import java.util.Arrays;

import java.util.StringTokenizer;

public class UserMainCode

{

public static int commonChars(String s,String s1)

{

int c=0,m=0;String t=null;

char a[]=s.toCharArray();

char b[]=s1.toCharArray();

Arrays.sort(a);

Arrays.sort(b);

s=new String(a);

s1=new String(b);

StringTokenizer st=new StringTokenizer(s);

StringTokenizer st1=new StringTokenizer(s1);

s=st.nextToken();

s1=st1.nextToken();

if(s.length()>s1.length())

{t=s1;

s1=s;

96s=t;

}

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

{

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

{

if(s.charAt(i)==s1.charAt(j))

{

if((s.indexOf(s.charAt(i))==s.lastIndexOf(s.charAt(i)))&&(s1.indexOf(s1.charAt(j))==s1.lastIndexOf(s1.charAt(j))))

{

c++;

}

}}}

return c;

}

}

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

String s=sc.nextLine();

String s1=sc.nextLine();

System.out.println(UserMainCode.commonChars(s,s1));

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

63.Find Digits

For a given double number with atleast one decimal value, Write a program to compute the number of digits before and after the decimal point in the following format ? noOfDigitsBeforeDecimal:noOfDigitsAfterDecimal.

Note: Ignore zeroes at the end of the decimal (Except if zero is the only digit after decimal. Refer Example 2 and 3)

Include a class UserMainCode with a static method findNoDigits which accepts the decimal value. The return type is string.

Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a double.

Output consists of string.

Refer sample output for formatting specifications.

Sample Input 1:

843.21

Sample Output 1:

3:2

Sample Input 2:

20.130

Sample Output 2:

2:2

Sample Input 3:

20.130

Sample Output 3:

2:2

import java.util.StringTokenizer;

public class UserMainCode {

public static String findNoDigits(double d) {

int n1=0,n2=0;

String s=String.valueOf(d);

StringTokenizer t=new StringTokenizer(s,".");

String s1=t.nextToken();

String s2=t.nextToken();

n1=s1.length();

n2=s2.length();

if(s1.charAt(0)=='0')

n1=s1.length()-1;

if(n2!=1)

if(s2.charAt(s2.length()-1)=='0')

n2=s2.length()-1;

String s3=String.valueOf(n1)+":"+String.valueOf(n2);

return s3;

}

}

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

double d=sc.nextDouble();

System.out.println(UserMainCode.findNoDigits(d));

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

64.Find Distance (findDistance)3/4/5/2

Write a Program that accepts four int inputs(x1,y1,x2,y2) as the coordinates of two points. Calculate the distance between the two points using the below formula.

Formula : square root of((x1-x2)\*(x1-x2)+(y1-y2)\*(y1-y2))

Then, Round the result to return an int

Include a class UserMainCode with a static method findDistance which accepts four integers. The return type is integer representing the formula.

Create a Class Main which would be used to accept the input integers and call the static method present in UserMainCode.

Input and Output Format:

Input consists of four integers.

Output consists of a single integer.

Refer sample output for formatting specifications.

Sample Input 1:

3

4

5

2

Sample Output 1:

3

Sample Input 2:

3

1

5

2

Sample Output 2:

2

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int a=sc.nextInt();

int b=sc.nextInt();

int c=sc.nextInt();

int d=sc.nextInt();

System.out.println(UserMainCode.findDistance(a,b,c,d));

}

}

public class UserMainCode {

public static int findDistance(int a,int b,int c,int d) {

long q=(int)Math.round(Math.sqrt(((a-c)\*(a-c))+((b-d)\*(b-d))));

return (int) q;

}

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

65.Find the difference between Dates in months (getMonthDifference)2012-03-01/2012-04-12/1

Given a method with two date strings in yyyy-mm-dd format as input. Write code to find the difference between two dates in months.

Include a class UserMainCode with a static method getMonthDifference which accepts two date strings as input.

The return type of the output is an integer which returns the diffenece between two dates in months.

Create a class Main which would get the input and call the static method getMonthDifference present in the UserMainCode.

Input and Output Format:

Input consists of two date strings.

Format of date : yyyy-mm-dd.

Output is an integer.

Refer sample output for formatting specifications.

Sample Input 1:

2012-03-01

2012-04-16

Sample Output 1:

1

Sample Input 2:

2011-03-01

2012-04-16

Sample Output 2:

13

import java.text.\*;

public class Main

{

public static void main(String[] args) throws ParseException

{

String s1="2011-03-01";

String s2="2012-04-16";

System.out.println(UserMainCode.getMonthDifferences(s1,s2));

}

}

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Calendar;

import java.util.Date;

public class UserMainCode {

public static int getMonthDifferences(String s1, String s2) throws ParseException {

SimpleDateFormat sdf=new SimpleDateFormat("yyyy-MM-dd");

Date d1=sdf.parse(s1);

Date d2=sdf.parse(s2);

Calendar cal=Calendar.getInstance();

cal.setTime(d1);

int months1=cal.get(Calendar.MONTH);

int year1=cal.get(Calendar.YEAR);

cal.setTime(d2);

int months2=cal.get(Calendar.MONTH);

int year2=cal.get(Calendar.YEAR);

int n=((year2-year1)\*12)+(months2-months1);

return n;

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

66.Find the element position in a reversed string array (getElementPosition)4/red/green/blue

Given a method with an array of strings and one string variable as input. Write code to sort the given array in reverse alphabetical order and return the postion of the given string in the array.

Include a class UserMainCode with a static method getElementPosition which accepts an array of strings and a string variable as input.

The return type of the output is an integer which is the position of given string value from the array.

Create a Main class which gets string array and a string variable as an input and call the static method getElementPosition present in the UserMainCode.

Input and Output Format:

Input is an string array. First element in the input represents the size the array

Assume the position of first element is 1.

Output is an integer which is the position of the string variable

Sample Input 1:

4

red

green

blue

ivory

ivory

Sample Output 1:

2

Sample Input 2:

3

grape

mango

apple

apple

Sample Output 2:

3

import java.util.ArrayList;

import java.util.Collections;

public class UserMainCode {

public static int getElementPosition(String[] a, String b) {

int r=0;

ArrayList<String>al=new ArrayList<String>();

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

{

al.add(a[i]);

}

//System.out.println(al);

Collections.sort(al);

//System.out.println(al);

Collections.reverse(al);

//System.out.println(al);

for(int i=0;i<al.size();i++)

{

if(b.equals(al.get(i)))

{ r=i+1;

//System.out.println(i+1);

}

}

return r;

}

}

import java.text.\*;

public class Main

{

public static void main(String[] args) throws ParseException

{

String []a={"grape","mango","apple"};

String b="apple";

UserMainCode.getElementPosition(a,b);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

67.Finding the day of birth (calculateBornDay)29-07-2013/MONDAY

iven an input as date of birth of person, write a program to calculate on which day (MONDAY,TUESDAY....) he was born store and print the day in Upper Case letters.

Include a class UserMainCode with a static method calculateBornDay which accepts a string as input.

The return type of the output is a string which should be the day in which the person was born.

Create a class Main which would get the input and call the static method calculateBornDay present in the UserMainCode.

Input and Output Format:

NOTE: date format should be(dd-MM-yyyy)

Input consists a date string.

Output is a string which the day in which the person was born.

Refer sample output for formatting specifications.

Sample Input 1:

29-07-2013

Sample Output 1:

MONDAY

Sample Input 2:

14-12-1992

Sample Output 2:

MONDAY

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Date;

import java.util.Scanner;

public class Main {

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

Scanner sc=new Scanner(System.in);

String s1=sc.nextLine();

System.out.println(UserMainCode.calculateBornDay(s1));

}

}

import java.text.SimpleDateFormat;

import java.text.ParseException;

import java.util.Date;

public class UserMainCode {

public static String calculateBornDay(String s1) throws ParseException

{

SimpleDateFormat sdf=new SimpleDateFormat("dd-MM-yyyy");

SimpleDateFormat sdf1=new SimpleDateFormat("EEEEE");

Date d=sdf.parse(s1);

String s=sdf1.format(d);

return s.toUpperCase();

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

68.Flush Characters (getSpecialChar) cogniz$#45Ant

Write a program to read a string from the user and remove all the alphabets and spaces from the String, and only store special characters and digit in the output String. Print the output string.

Include a class UserMainCode with a static method getSpecialChar which accepts a string. The return type (String) should return the character removed string.

Create a Class Main which would be used to accept a string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a strings.

Output consists of an String (character removed string).

Refer sample output for formatting specifications.

Sample Input :

cogniz$#45Ant

Sample Output :

$#45

public class UserMainCode {

public static String getSpecialChar(String s1) {

StringBuffer sb= new StringBuffer();

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

{

char a=s1.charAt(i);

if(!Character.isAlphabetic(a))

sb.append(a);

}

return sb.toString();

}

}

import java.text.\*;

public class Main

{

public static void main(String[] args) throws ParseException

{

String s="cogniz$#45Ant";

String s1=UserMainCode.getSpecialChar(s);

System.out.println(s1);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

69.Forming New Word from a String (formNewWord)california/3/calnia

Write a program to read a string and a positive integer n as input and construct a string with first n and last n characters in the given string.

Include a class UserMainCode with a static method formNewWord which accepts a string and positive integer .

The return type of the output should be a string (value) of first n character and last n character.

Create a class Main which would get the input as a string and integer n and call the static method formNewWord present in the UserMainCode.

Input and Output Format:

Input consists of a string of even length.

Output is a string .

Note: The given string length must be >=2n.

Refer sample output for formatting specifications.

Sample Input 1:

California

3

Sample Output 1:

Calnia

Sample Input 2:

this

1

Sample Output 2:

ts

public class UserMainCode {

public static String subStringOfgivenString(String s1, int n1)

{

int n = 2\*n1;

String s = new String();

if(s1.length()>n)

{

s = s1.substring(0,n1) + s1.substring(s1.length()-n1, s1.length());

return s;

}

else

return null;

}

}

public class Main

{

public static void main(String[] args)

{

String s1="this";

int n1=1;

System.out.println(UserMainCode.subStringOfgivenString(s1,n1));

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

70.Generate the series (addSeries)9/-3

Given a method taking an odd positive Integer number as input. Write code to evaluate the following series:

1+3-5+7-9?+/-n.

Include a class UserMainCode with a static method addSeries which accepts a positive integer .

The return type of the output should be an integer .

Create a class Main which would get the input as a positive integer and call the static method addSeries present in the UserMainCode.

Input and Output Format:

Input consists of a positive integer n.

Output is a single integer .

Refer sample output for formatting specifications.

Sample Input 1:

9

Sample Output 1:

-3

Sample Input 2:

11

Sample Output 2:

8

import java.util.\*;

public class Main {

public static int consecutiveSumSubofOddNos(int n){

List<Integer> l1=new ArrayList<Integer>();

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

if(i%2!=0)

l1.add(i);

int n1=l1.get(0);

for(int i=1;i<l1.size();i++)

if(i%2!=0)

n1=n1+l1.get(i);

else

n1=n1-l1.get(i);

return n1;

}

public static void main(String[] args) {

Scanner s=new Scanner(System.in);

int n=s.nextInt();

System.out.println(consecutiveSumSubofOddNos(n));

}}

c

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

71.Grade Calculator I (calculateGrade)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.LinkedHashMap;

import java.util.Map;

import java.util.Scanner;

public class Main

{

public static void main(String[]arg)

{

LinkedHashMap<String,Double>hm=new LinkedHashMap<String,Double>();

LinkedHashMap<String,String>hm1=new LinkedHashMap<String,String>();

Scanner sc=new Scanner(System.in);

int n=sc.nextInt();

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

{

String s=sc.next();

double d=sc.nextDouble();

hm.put(s,d);

}

LinkedHashMap<String,String>hm2=UserMainCode.dis(hm);

for(Map.Entry<String,String>entry:hm2.entrySet())

{

System.out.println(entry.getKey());

System.out.println(entry.getValue());

}}}

import java.util.LinkedHashMap;

import java.util.Map;

class UserMainCode

{

public static LinkedHashMap<String,String>dis(LinkedHashMap<String,Double>h1)

{

LinkedHashMap<String,String>h2=new LinkedHashMap<String,String>();

for(Map.Entry m:h1.entrySet())

{

double d=(Double)m.getValue();

if(d>60)

{

String s=(String)m.getKey();

h2.put(s,"pass");

}

else

{

String s=(String)m.getKey();

h2.put(s,"fail");

}

}

return h2;

}

}

72.Grade Calculator

A School wants to assign grades to its students based on their marks. You have been assigned as the programmer to automate this process. You would like to showcase your skills by creating a quick prototype. The prototype consists of the following steps:

Read student details from the User. The details would include roll no, mark in the given order. The datatype for id is integer, mark is integer.

You decide to build a hashmap. The hashmap contains roll no as key and mark as value.

BUSINESS RULE:

1. If Mark is greater than or equal to 80 store medal as ""GOLD"".

2. If Mark is less then to 80 and greater than or equal to 60 store medal as ""SILVER"".

3 .If Mark is less then to 60 and greater than or equal to 45 store medal as ""BRONZE"" else store ""FAIL"".

Store the result in TreeMap in which Roll No as Key and grade as value.

4. You decide to write a function calculateGrade which takes the above hashmaps as input and returns the treemap as output. Include this function in class UserMainCode.

Create a Class Main which would be used to read employee details in step 1 and build the two hashmaps. Call the static method present in UserMainCode.

Input and Output Format:

Input consists of employee details. The first number indicates the size of the students. The next two values indicate the roll id, mark.

Output consists of a single string.

Refer sample output for formatting specifications.

Sample Input 1:

2

1010

80

100

40

Sample Output 1:

100

FAIL

1010

GOLD

import java.util.HashMap;

import java.util.Iterator;

import java.util.HashMap;

import java.util.TreeMap;

import java.util.Scanner;

public class main {

public static void main(String []args){

Scanner sc=new Scanner(System.in);

int s=sc.nextInt();

HashMap<Integer,Integer>hm=new HashMap<Integer,Integer>();

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

{

hm.put(sc.nextInt(),sc.nextInt());

}

TreeMap<Integer,String>tm=new TreeMap<Integer,String>();

tm=Usermaincode.display(hm);

Iterator<Integer> it=tm.keySet().iterator();

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

{

int n=it.next();

String fac=tm.get(n);

System.out.println(n);

System.out.println(fac);

}

}

}

import java.util.Iterator;

import java.util.HashMap;

import java.util.TreeMap;

public class Usermaincode

{

public static TreeMap<Integer,String> display (HashMap<Integer,Integer>hm)

{

TreeMap<Integer,String>tm=new TreeMap<Integer,String>();

Iterator<Integer> it=hm.keySet().iterator();

while(it.hasNext())

{

int id=it.next();

int mark=hm.get(id);

if(mark>=80)

tm.put(id,"GOLD");

else if(mark<80 && mark>=60)

tm.put(id,"SILVER");

else if(mark<60 && mark>=45)

tm.put(id,"BRONZE");

else

tm.put(id,"FAIL");

}

return tm;

}}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

73.ID Validation (validateIDLocations)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.\*;

public class Main {

public static void main(String[] args) {

String s1="CTS-hyd-1234";

String s2="hyderabad";

boolean b=formattingString(s1,s2);

if(b==true)

System.out.println("String format:CTS-LLL-XXXX");

else

System.out.println("not in required format");

}

public static boolean formattingString(String s1, String s2) {

String s3=s2.substring(0, 3);

boolean b=false;

StringTokenizer t=new StringTokenizer(s1,"-");

String s4=t.nextToken();

String s5=t.nextToken();

String s6=t.nextToken();

if(s4.equals("CTS") && s5.equals(s3) && s6.matches("[0-9]{4}"))

b=true;

else{

b=false;}

return b;

}

}

74.Initial Format (nameFormatter)Jessica Miller

Write a program to input a person's name in the format "FirstName LastName" and return the person name in the following format - "LastName, InitialOfFirstName".

Include a class UserMainCode with a static method nameFormatter which accepts a string. The return type (string) should return the expected format.

Create a Class Main which would be used to accept Input String and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string that corresponds to a Person's name.

Output consists of a string(person's name in expected format).

Refer sample output for formatting specifications.

Sample Input :

Jessica Miller

Sample Output:

Miller, J

import java.util.StringTokenizer;

class UserMainCode

{

public static String nameFormatter(String s1)

{

StringBuffer sb=new StringBuffer();

StringTokenizer st=new StringTokenizer(s1," ");

String s2=st.nextToken();

String s3=st.nextToken();

sb.append(s3).append(",");

sb.append(s2.substring(0,1).toUpperCase());

return sb.toString();

}

}

public class Main

{

public static void main(String[] args) {

String s1="vishal jadiya";

String z=UserMainCode.nameFormatter(s1);

System.out.println(z);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

75.Integer Factorial(getFactorial)

Give an array of integer as input, store the numbers and their factorials in an hashmap and print the same.

Include a class UserMainCode with a static method getFactorial which accepts the integer array. The return type is the hashmap which is printed key:value.

Create a Class Main which would be used to accept the integer array and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a number denoting the size of the array and followed by the elements.

Output consists of a hashmap printed in the output format .

Refer sample output for formatting specifications.

Sample Input1:

4

2

3

5

4

Sample Output1:

2:2

3:6

5:120

4:24

import java.util.HashMap;

import java.util.Iterator;

import java.util.LinkedHashMap;

import java.util.Scanner;

public class Main {

public static void main(String []args){

Scanner sc=new Scanner(System.in);

int s=Integer.parseInt(sc.nextLine());

int []a=new int[s];

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

{

a[i]=sc.nextInt();

}

LinkedHashMap<Integer,Integer>hm2=new LinkedHashMap<Integer,Integer>();

hm2=UserMainCode.getFactorial(a);

Iterator<Integer> it=hm2.keySet().iterator();

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

{

int n=it.next();

int fac=hm2.get(n);

System.out.println(n+":"+fac);

}

}

}

import java.text.DecimalFormat;

import java.util.HashMap;

import java.util.Iterator;

import java.util.LinkedHashMap;

public class UserMainCode

{public static LinkedHashMap<Integer,Integer> getFactorial(int[] a)

{

LinkedHashMap<Integer,Integer>hm=new LinkedHashMap<Integer,Integer>();

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

{

int u=1;

for(int j=1;j<=a[i];j++)

{u=u\*j;}

hm.put(a[i],u);

}

return hm;

}}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

76.Interest Calculation

Write a program to calculate amount of the acccount holders based on the below mentioned prototype:

1. Read account details from the User. The details would include id, DOB (date of birth) and amount in the given order. The datatype for id is string, DOB is string and amount is integer.

2. You decide to build two hashmaps. The first hashmap contains employee id as key and DOB as value, and the second hashmap contains same employee ids as key and amount as value.

3. Rate of interest as on 01/01/2015:

a. If the age greater than or equal to 60 then interest rate is 10% of Amount.

b.If the age less then to 60 and greater than or equal to 30 then interest rate is 7% of Amount.

v. If the age less then to 30 interest rate is 4% of Amount.

4. Revised Amount= principle Amount + interest rate.

5. You decide to write a function calculateInterestRate which takes the above hashmaps as input and returns the treemap as output. Include this function in class UserMainCode.

Create a Class Main which would be used to read employee details in step 1 and build the two hashmaps. Call the static method present in UserMainCode.

Input and Output Format:

Input consists of account details. The first number indicates the size of the acoount. The next three values indicate the user id, DOB and amount. The Employee DOB format is ?dd-mm-yyyy?

Output consists of the user id and the amount for each user one in a line.

Refer sample output for formatting specifications.

Sample Input 1:

4

SBI-1010

20-01-1987

10000

SBI-1011

03-08-1980

15000

SBI-1012

05-11-1975

20000

SBI-1013

02-12-1950

30000

Sample Output 1:

SBI-1010:10400

SBI-1011:16050

SBI-1012:21400

SBI-1013:33000

import java.util.HashMap;

import java.util.Iterator;

import java.util.Scanner;

import java.util.TreeMap;

public class main {

public static void main(String []args){

Scanner sc=new Scanner(System.in);

int s=Integer.parseInt(sc.nextLine());

HashMap<String,String>hm=new HashMap<String,String>();

HashMap<String,Integer>hm1=new HashMap<String,Integer>();

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

{

String id=sc.nextLine();

hm.put(id, sc.nextLine());

hm1.put(id,Integer.parseInt(sc.nextLine()));

}

TreeMap<String,Integer>tm=new TreeMap<String,Integer>();

tm=Usermaincode.calculateInterestRate(hm,hm1);

Iterator<String> it=tm.keySet().iterator();

while(it.hasNext())

{

String n=it.next();

int fac=tm.get(n);

System.out.println(n+":"+fac);

}

}

}

import java.text.DecimalFormat;

import java.text.SimpleDateFormat;

import java.util.Date;

import java.util.HashMap;

import java.util.Iterator;

import java.util.HashMap;

import java.util.TreeMap;

public class Usermaincode

{

public static TreeMap<String,Integer> calculateInterestRate(HashMap<String,String>hm,HashMap<String,Integer>hm1)

{

int year=0,amount=0;

double dis=0;

String now="01/01/2015";

TreeMap<String,Integer>tm=new TreeMap<String,Integer>();

Iterator<String> it=hm.keySet().iterator();

while(it.hasNext())

{

String id=it.next();

String dor=hm.get(id);

amount=hm1.get(id);

SimpleDateFormat sdf=new SimpleDateFormat("dd-MM-yyyy");

SimpleDateFormat sdf1=new SimpleDateFormat("dd/MM/yyyy");

try

{

Date d=sdf.parse(dor);

Date d1=sdf1.parse(now);

sdf.setLenient(false);

int y=d.getYear();

int y1=d1.getYear();

int m=d.getMonth();

int m1=d1.getMonth();

int day=d.getDay();

int day1=d1.getDay();

year=y1-y;

if(m>m1)

year--;

else if(m==m1)

{if(day<day1)

year--;

}

if(year>=60)

dis=0.1\*amount+amount;

else if(year<60 && year>=30 )

dis=0.07\*amount+amount;

else

dis=0.04\*amount+amount;

tm.put(id,(int)dis);

}

catch(Exception e)

{

e.printStackTrace();

}

}

return tm;

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

77.IP Validator (ipValidator) 132.145.184.210

Write a program to read a string and validate the IP address. Print ?Valid? if the IP address is valid, else print ?Invalid?.

Include a class UserMainCode with a static method ipValidator which accepts a string. The return type (integer) should return 1 if it is a valid IP address else return 2.

Create a Class Main which would be used to accept Input String and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string that corresponds to an IP.

Output consists of a string(?Valid? or ?Invalid?).

Refer sample output for formatting specifications.

Note: An IP address has the format a.b.c.d where a,b,c,d are numbers between 0-255.

Sample Input 1:

132.145.184.210

Sample Output 1:

Valid

Sample Input 2:

132.145.184.290

Sample Output 2:

Invalid

import java.util.StringTokenizer;

class UserMainCode

{

public static boolean ipValidator(String ipAddress) {

boolean b1=false;

StringTokenizer t=new StringTokenizer(ipAddress,".");

int a=Integer.parseInt(t.nextToken());

int b=Integer.parseInt(t.nextToken());

int c=Integer.parseInt(t.nextToken());

int d=Integer.parseInt(t.nextToken());

if((a>=0 && a<=255)&&(b>=0 && b<=255)&&(c>=0 && c<=255)&&(d>=0 && d<=255))

b1=true;

return b1;

}

}

public class Main

{

public static void main(String[] args) {

String ipAddress="132.145.184.210";

boolean b=UserMainCode.ipValidator(ipAddress);

if(b==true)

System.out.println("Valid");

else

System.out.println("Invalid");

}

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

78.ISBN Validation

Write a program to read a string and validate the given ISBN as input.

Validation Rules:

1. An ISBN (International Standard Book Number) is a ten digit code which uniquely identifies a book.

2. To verify an ISBN you calculate 10 times the first digit, plus 9 times the second digit, plus 8 times the third ..all the way until you add 1 times the last digit.

If the final number leaves no remainder when divided by 11 the code is a valid ISBN.

Example 1:

Input:0201103311

Calculation: 10\*0 + 9\*2 + 8\*0 + 7\*1 + 6\*1 + 5\*0 + 4\*3 + 3\*3 + 2\*1 + 1\*1 = 55.

55 mod 11 = 0

Hence the input is a valid ISBN number

Output: true

Include a class UserMainCode with a static method validateISBN which accepts the string. The return type is TRUE / FALSE as per problem.

Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Output consists of TRUE / FALSE.

Refer sample output for formatting specifications.

Sample Input 1:

0201103311

Sample Output 1:

TRUE

class UserMainCode

{

public static boolean validateISBN(String ip) {

boolean b=false;

int sum=0;

for(int i=0,j=ip.length();i<ip.length();i++,j--){

String s=String.valueOf(ip.charAt(i));

int n=Integer.parseInt(s);

sum+=(n\*j); }

//System.out.println(sum);

if(sum%11==0)

b=true;

return b;

}

}

public class Main

{

public static void main(String[] args) {

String ip="0201103311";

boolean b=UserMainCode.validateISBN(ip);

if(b==true)

System.out.println("TRUE");

else

System.out.println("check ur data");

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

79.Kaprekar Number (getKaprekarNumber)

Write a program to check whether the given input number is a Kaprekar number or not.

Note : A positive whole number ?n? that has ?d? number of digits is squared and split into two pieces, a right-hand piece that has ?d? digits and a left-hand piece that has remaining ?d? or ?d-1? digits. If the sum of the two pieces is equal to the number, then ?n? is a Kaprekar number.

If its Kaprekar number assign to output variable 1 else -1.

Example 1:

Input1:9

9^2 = 81, right-hand piece of 81 = 1 and left hand piece of 81 = 8

Sum = 1 + 8 = 9, i.e. equal to the number. Hence, 9 is a Kaprekar number.

Example 2:

Input1:45

Hint:

45^2 = 2025, right-hand piece of 2025 = 25 and left hand piece of 2025 = 20

Sum = 25 + 20 = 45, i.e. equal to the number. Hence, 45 is a Kaprekar number."

Include a class UserMainCode with a static method ?getKaprekarNumber? that accepts an integer argument and returns an integer. The method returns 1 if the input integer is a Kaprekar number. Else the method returns -1.

Create a class Main which would get the an Integer as input and call the static method getKaprekarNumber present in the UserMainCode.

Input and Output Format:

Input consists of an integer.

Output consists of a single string that is either ?Kaprekar Number? or ?Not A Kaprekar Number?

Sample Input 1:

9

Sample Output 1:

Kaprekar Number

Sample Input 2:

45

Sample Output 2:

Kaprekar Number

Sample Input 3:

4

Sample Output 3:

Not A Kaprekar Number

public class UserMainCode{

public static int getKaprekarNumber(int a)

{

int count=0,j=0;

int a1=a;

while(a1!=0)

{

count=count+1;

a1=a1/10;

}

int square=a\*a;

String s=Integer.toString(square);

String s1=s.substring(0,count);

String s2=s.substring(count);

int x=Integer.parseInt(s1);

int y=Integer.parseInt(s2);

int result =x+y;

if(result==a)

return 1;

else

return 2;

}}

import java.util.Scanner;

public class Main{

public static void main(String[] args)

{

Scanner sc=new Scanner(System.in);

int n=sc.nextInt();

int res=UserMainCode.getKaprekarNumber(n);

if(res==1)

System.out.println("Kaprekar Number");

else

System.out.println("Not A Kaprekar Number");

}}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

80.Largest Chunk

Write a program to read a string and find the length of the largest chunk in the string. If there are no chunk print ?No chunks? else print the length.

NOTE: chunk is the letter which is repeating 2 or more than 2 times.

Include a class UserMainCode with a static method largestChunk which accepts a string. The return type (Integer) should return the length of the largest chunk if the chunk is present, else return -1.

Create a Class Main which would be used to accept Input String and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Refer sample output for formatting specifications.

Sample Input 1:

You are toooo good

Sample Output 1:

4

(Because the largest chunk is letter 'o' which is repeating 4 times)

Sample Input 2:

who are u

Sample Output 2:

No chunks

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.StringTokenizer;

public class UserMainCode{

public static int largestChunk(String s1) {

int max=0;

StringTokenizer t=new StringTokenizer(s1," ");

while(t.hasMoreTokens()){

String s2=t.nextToken();

int n=0;

for(int i=0;i<s2.length()-1;i++)

if(s2.charAt(i)==s2.charAt(i+1))

n++;

if(n>max)

max=n;

}

return (max+1);

} }

public class Main{

public static void main(String[] args) {

String s1="who are u";

System.out.println(UserMainCode.largestChunk(s1));

}

}

81.Largest Difference (checkDifference)

Write a program to read a integer array, find the largest difference between adjacent elements and display the index of largest difference.

EXAMPLE:

input1: {2,4,5,1,9,3,8}

output1: 4 (here largest difference 9-1=8 then return index of 9 ie,4)

Include a class UserMainCode with a static method checkDifference which accepts the integer array. The return type is integer.

Create a Class Main which would be used to accept the integer array and call the static method present in UserMainCode.

Input and Output Format:

Input consists of an integer n which is the number of elements followed by n integer values.

Output consists of integer.

Refer sample output for formatting specifications.

Sample Input 1:

7

2

4

5

1

9

3

8

Sample Output 1:

4

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 7/2/4/5/1/9/3/8

public class UserMainCode{

public static int checkDifference(int[] n1){

int n2,n3=0,n4=0,i;

for(i=0;i<n1.length-1;i++){

n2=Math.abs(n1[i]-n1[i+1]);

if(n2>n3){

n3=n2;

n4=i+1; }}

return n4;

} }

public class Main{

public static void main(String[] args) {

int[] n1={2,4,5,1,9,3,8};

System.out.println(UserMainCode.checkDifference(n1));

}

}

82.Largest Element (checkLargestAmongCorner)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Write a program to read an int array of odd length, compare the first, middle and the last elements in the array and return the largest. If there is only one element in the array return the same element.

Include a class UserMainCode with a static method checkLargestAmongCorner which accepts an int arrayThe return type (integer) should return the largest element among the first, middle and the last elements.

Create a Class Main which would be used to accept Input array and call the static method present in UserMainCode.

Assume maximum length of array is 20.

Input and Output Format:

Input consists of n+1 integers. The first integer corresponds to n, the number of elements in the array. The next 'n' integers correspond to the elements in the array.

Output consists of a single Integer.

Refer sample output for formatting specifications.

Sample Input 1:

5

2

3

8

4

5

Sample Output 1:

8

public class UserMainCode{

public static int checkLargestAmongCorner(int []a)

{

int max=0;

int x,y,z;

x=a[0];

y=a[a.length/2];

z=a[a.length-1];

if(x>y&&x>z)

max=x;

else if(y>x&&y>z)

max=y;

else if(z>x&&z>y)

max=z;

return max;

}}

import java.util.Scanner;

public class Main{

public static void main(String[] args) {

Scanner ip=new Scanner(System.in);

int n=ip.nextInt();

int a[]=new int[n];

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

a[i]=ip.nextInt();

System.out.println(UserMainCode.checkLargestAmongCorner(a));

}

}

83.Largest Key in HashMap

Write a program that construts a hashmap and returns the value corresponding to the largest key.

Include a class UserMainCode with a static method getMaxKeyValue which accepts a string. The return type (String) should be the value corresponding to the largest key.

Create a Class Main which would be used to accept Input string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of 2n+1 values. The first value corresponds to size of the hashmap. The next n pair of numbers equals the integer key and value as string.

Output consists of a string which is the value of largest key.

Refer sample output for formatting specifications.

Sample Input 1:

3

12

amron

9

Exide

7

SF

Sample Output 1:

amron

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.HashMap;

import java.util.Iterator;

public class UserMainCode{

public static String getMaxKeyValue(HashMap<Integer, String> hm) {

int max=0;

String s2=new String();

Iterator<Integer>itr=hm.keySet().iterator();

while(itr.hasNext())

{

int a=itr.next();

if(a>max)

{

max=a;

String s3=hm.get(a);

s2=s3;

}

}

return s2;

} }

import java.util.HashMap;

public class Main{

public static void main(String[] args) {

HashMap<Integer, String>hm=new HashMap<Integer, String>();

hm.put(12,"amron");

hm.put(9, "excide");

hm.put(7,"SF");

System.out.println(UserMainCode.getMaxKeyValue(hm));

}

}

84.Largest Span(getLargestSpan)

Write a program to read an array and find the size of largest span in the given array

""span"" is the number of elements between two repeated numbers including both numbers. An array with single element has a span of 1.

.

Include a class UserMainCode with a static method getMaxSpan which accepts a single integer array. The return type (integer) should be the size of largest span.

Create a Class Main which would be used to accept Input array and call the static method present in UserMainCode.

Input and Output Format:

Input consists of n+1 integers. The first integer corresponds to n, the number of elements in the array. The next 'n' integers correspond to the elements in the array.

Output consists of a single Integer.

Refer sample output for formatting specifications.

Assume that the maximum number of elements in the array is 20.

Sample Input 1:

5

1

2

1

1

3

Sample Output 1:

4

Sample Input 2:

7

1

4

2

1

4

1

5

Sample Output 2:

6

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.Scanner;

public class Main {

public static void main(String[] args)

{

Scanner sc=new Scanner(System.in);

int n=sc.nextInt();

int []a=new int[n];

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

{

a[i]=sc.nextInt();

}

System.out.print(UserMainCode.getMaxSpan(a,n));

}}

public class UserMainCode {

public static int getMaxSpan(int[] x,int n)

{

int gap=0,max=0;

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

{

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

{

if(x[i]==x[j])

{

gap=j;

}

}

if(gap-i>max)

max=gap-i;

}

return max+1;

}

}

85.Largest Span (getMaxSpan)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public class Main {

public static void main(String[] args) {

int[]a={1,2,1,1,3};

System.out.println(maxSpan(a));

}

public static int maxSpan(int[] a) {

String s2 = null;

int n=0;

StringBuffer sb=new StringBuffer();

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

sb.append(String.valueOf(a[i]));

String s1=sb.toString();

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

for(int j=i+1;j<s1.length();j++)

if(s1.charAt(i)==s1.charAt(j))

s2=String.valueOf(s1.charAt(j));

int n1=s1.indexOf(s2);

int n2=s1.lastIndexOf(s2);

for(int i=n1+1;i<n2;i++)

n++;

return (n+2);

}

}

86.Last letters(getLastLetter)

Write a program to read a sentence as a string and store only the last letter of each word of the sentence in capital letters separated by $. Print the final string.

Include a class UserMainCode with a static method getLastLetter which accepts a string. The return type (string) should return the final string.

Create a Class Main which would be used to read a string, and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Output consists of a string (the final string).

Refer sample output for formatting specifications.

Smaple Input :

This is a cat

Sample Output :

S$S$A$T

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.Scanner;

public class main

{

public static void main(String[] args)

{

Scanner sc=new Scanner(System.in);

String number=sc.nextLine();

System.out.print(Usermaincode.getLastLetter(number));

}}

import java.util.StringTokenizer;

public class Usermaincode {

public static String getLastLetter(String input){

StringTokenizer st=new StringTokenizer(input);

String str=st.nextToken();

String solution=Character.toUpperCase(str.charAt(str.length()-1))+"";

while(st.hasMoreTokens()){

str=st.nextToken();

solution+="$"+Character.toUpperCase(str.charAt(str.length()-1));

}

return solution;

}

}

87.Leap Year

Write a program to read a string containing date in DD/MM/YYYY format and check if its a leap year. If so, return true else return false.

Include a class UserMainCode with a static method isLeapYear which accepts the string. The return type is the boolean indicating TRUE / FALSE.

Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Output consists of TRUE / FALSE.

Refer sample output for formatting specifications.

Sample Input 1:

23/02/2012

Sample Output 1:

TRUE

Sample Input 2:

12/12/2011

Sample Output 2:

FALSE

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.io.IOException;

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Date;

import java.util.GregorianCalendar;

import java.util.StringTokenizer;

public class UserMainCode {

public static boolean isLeapYear(String s1) throws IOException, ParseException

{

//boolean flag=false;

SimpleDateFormat sdf=new SimpleDateFormat("dd/MM/yyyy");

GregorianCalendar g=new GregorianCalendar();

StringTokenizer t=new StringTokenizer(s1,"/");

String s2=t.nextToken();

String s3=t.nextToken();

String s4=t.nextToken();

int n1=Integer.parseInt(s4);

Date d1=sdf.parse(s1);

boolean b=g.isLeapYear(n1);

return b;

}

}

import java.util.Scanner;

public class Main

{

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

{

Scanner sc=new Scanner(System.in);

String number=sc.nextLine();

boolean res=UserMainCode.isLeapYear(number);

System.out.println(res);

}}

88.Length of same word

Write a program to read a string containing multiple words find the first and last words, if they are same, return the length and if not return the sum of length of the two words.

Include a class UserMainCode with a static method compareLastWords which accepts the string. The return type is the length as per problem.

Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Output consists of a integer.

Refer sample output for formatting specifications.

Sample Input 1:

This is Cognizant Academy

Sample Output 1:

11

Sample Input 2:

Hello World Hello

Sample Output 2:

5

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.io.IOException;

import java.text.ParseException;

import java.util.ArrayList;

import java.util.List;

import java.util.StringTokenizer;

public class UserMainCode {

public static void compareLastWords(String s1) throws IOException, ParseException

{

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

StringTokenizer t=new StringTokenizer(s1," ");

while(t.hasMoreTokens())

{

String s2=t.nextToken();

l.add(s2);

}

String s3=l.get(0);

String s4=l.get(l.size()-1);

if(s3.equals(s4))

{

int n=s3.length();

System.out.println(n);

}

else

{

int n1=s3.length();

int n2=s4.length();

System.out.println(n1+n2);

}

}

}

import java.util.Scanner;

public class Main

{

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

{

Scanner sc=new Scanner(System.in);

String number=sc.nextLine();

UserMainCode.compareLastWords(number);

}}

???????????????????????????????????????????????????????????????????????????????????????????????????

import java.util.Scanner;

public class Main

{

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

{

Scanner sc=new Scanner(System.in);

String number=sc.nextLine();

System.out.println(UserMainCode.compareLastWords(number));

}}

import java.io.IOException;

import java.text.ParseException;

import java.util.ArrayList;

import java.util.List;

import java.util.StringTokenizer;

public class UserMainCode {

public static int compareLastWords(String s1) throws IOException, ParseException

{ int r=0;

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

StringTokenizer t=new StringTokenizer(s1," ");

while(t.hasMoreTokens())

{

String s2=t.nextToken();

l.add(s2);

}

String s3=l.get(0);

String s4=l.get(l.size()-1);

if(s3.equals(s4))

{

int n=s3.length();

//System.out.println(n);

r=n;

}

else

{

int n1=s3.length();

int n2=s4.length();

//System.out.println(n1+n2);

r=n1+n2;

}

return r;

}

}

89.Length of the Largest Chunk (largestChunk)

Write a program to read a string and find the length of the largest chunk in the string. If there are no chunk print ?No chunks? else print the length.

NOTE: chunk is the letter which is repeating 2 or more than 2 times.

Include a class UserMainCode with a static method largestChunk which accepts a string. The return type (Integer) should return the length of the largest chunk if the chunk is present, else return -1.

Create a Class Main which would be used to accept Input String and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Refer sample output for formatting specifications.

Sample Input 1:

You are toooo good

Sample Output 1:

4

(Because the largest chunk is letter 'o' which is repeating 4 times)

Sample Input 2:

who are u

Sample Output 2:

No chunks

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* You are toooo good/4

import java.util.StringTokenizer;

public class UserMainCode {

public static int largestChunk(String s1) {

int max=0;

StringTokenizer t=new StringTokenizer(s1," ");

while(t.hasMoreTokens()){

String s2=t.nextToken();

int n=0;

for(int i=0;i<s2.length()-1;i++)

if(s2.charAt(i)==s2.charAt(i+1))

n++;

if(n>max)

max=n;

}

return (max+1);

} }

import java.util.Scanner;

public class Main

{

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

{

Scanner sc=new Scanner(System.in);

String number=sc.nextLine();

int res=UserMainCode.largestChunk(number);

if(res==1)

System.out.println("No Chunks");

else

System.out.println(res);

}}

90.Longest Word

Write a Program which finds the longest word from a sentence. Your program should read a sentence as input from user and return the longest word. In case there are two words of maximum length return the word which comes first in the sentence.

Include a class UserMainCode with a static method getLargestWord which accepts a string The return type is the longest word of type string.

Create a Class Main which would be used to accept two Input strings and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string with maximum size of 100 characters.

Output consists of a single string.

Refer sample output for formatting specifications.

Sample Input 1:

Welcome to the world of Programming

Sample Output 1:

Programming

Sample Input 2:

ABC DEF

Sample Output 2:

ABC

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.StringTokenizer;

public class UserMainCode {

public static String getLargestWord(String s1) {

int max=0;

String s2=new String();

StringTokenizer t=new StringTokenizer(s1," ");

loop:

while(t.hasMoreTokens()){

String s3=t.nextToken();

int n=s3.length();

if(n>max){

max=n;

s2=s3;}

if(n==max)

for(int i=0;i<s3.length();i++){

char c1=s2.charAt(i);

char c2=s3.charAt(i);

if(c1!=c2){

if(c2<c1)

s2=s3;

continue loop;} }

}

return s2;

}}

import java.util.Scanner;

public class Main

{

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

{

Scanner sc=new Scanner(System.in);

String number=sc.nextLine();

System.out.println(UserMainCode.getLargestWord(number));

}}

91.Managers & Hashmaps

A Company wants to automate its payroll process. You have been assigned as the programmer to build this package. You would like to showcase your skills by creating a quick prototype. The prototype consists of the following steps:

Read Employee details from the User. The details would include id, designation and salary in the given order. The datatype for id is integer, designation is string and salary is integer.

You decide to build two hashmaps. The first hashmap contains employee id as key and designation as value, and the second hashmap contains same employee ids as key and salary as value.

The company decides to hike the salary of managers by 5000. You decide to write a function increaseSalaries which takes the above hashmaps as input and returns a hashmap with only managers id and their increased salary as output. Include this function in class UserMainCode.

Create a Class Main which would be used to read employee details in step 1 and build the two hashmaps. Call the static method present in UserMainCode.

Input and Output Format:

Input consists of employee details. The first number indicates the size of the employees. The next three values indicate the employee id, employee designation and employee salary.

Output consists of a single string.

Refer sample output for formatting specifications.

Sample Input 1:

2

2

programmer

3000

8

manager

50000

Sample Output 1:

8

55000

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.HashMap;

import java.util.Iterator;

import java.util.Scanner;

public class Main {

public static void main(String []args){

Scanner sc=new Scanner(System.in);

int s=Integer.parseInt(sc.nextLine());

HashMap<Integer,String>hm=new HashMap<Integer,String>();

HashMap<Integer,Integer>hm1=new HashMap<Integer,Integer>();

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

{

int id=Integer.parseInt(sc.nextLine());

hm.put(id, sc.nextLine());

hm1.put(id,Integer.parseInt(sc.nextLine()));

}

HashMap<Integer,Integer>hm2=new HashMap<Integer,Integer>();

hm2=Usermaincode.display(hm,hm1);

Iterator<Integer> it=hm2.keySet().iterator();

while(it.hasNext())

{

int n=it.next();

int fac=hm2.get(n);

System.out.println(n);

System.out.println(fac);

}}}

import java.util.HashMap;

import java.util.Iterator;

import java.util.HashMap;

public class Usermaincode

{public static HashMap<Integer,Integer> display(HashMap<Integer,String>hm,HashMap<Integer,Integer>hm1)

{

HashMap<Integer,Integer>hm3=new HashMap<Integer,Integer>();

Iterator<Integer> it=hm.keySet().iterator();

while(it.hasNext())

{

int id=it.next();

String name=hm.get(id);

if(name.equals("manager"))

{int salary=hm1.get(id)+5000;

hm3.put(id,salary);

}}

return hm3;

}

}

???????????????????????????????????????????????????????????????????????

import java.util.HashMap;

import java.util.Iterator;

import java.util.Scanner;

public class Main {

public static void main(String []args){

Scanner sc=new Scanner(System.in);

int s=Integer.parseInt(sc.nextLine());

HashMap<Integer,String>hm=new HashMap<Integer,String>();

HashMap<Integer,Integer>hm1=new HashMap<Integer,Integer>();

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

{

int id=Integer.parseInt(sc.nextLine());

hm.put(id, sc.nextLine());

hm1.put(id,Integer.parseInt(sc.nextLine()));

}

HashMap<Integer,Integer>hm2=new HashMap<Integer,Integer>();

hm2=MainUsercode.display(hm,hm1);

Iterator<Integer> it=hm2.keySet().iterator();

while(it.hasNext())

{

int n=it.next();

int fac=hm2.get(n);

System.out.println(n);

System.out.println(fac);

}}}

import java.util.HashMap;

import java.util.Iterator;

import java.util.HashMap;

public class MainUsercode

{

public static HashMap<Integer,Integer> display(HashMap<Integer,String>hm,HashMap<Integer,Integer>hm1)

{

HashMap<Integer,Integer>hm3=new HashMap<Integer,Integer>();

Iterator<Integer> it=hm.keySet().iterator();

while(it.hasNext())

{

int id=it.next();

String name=hm.get(id);

if(name.equals("manager"))

{int salary=hm1.get(id)+5000;

hm3.put(id,salary);

}}

return hm3;

}

}

92.Mastering Hashmap

You have recently learnt about hashmaps and in order to master it, you try and use it in all of your programs.

Your trainer / teacher has given you the following exercise:

Read 2n numbers as input where the first number represents a key and second one as value. Both the numbers are of type integers.

Write a function getAverageOfOdd to find out average of all values whose keys are represented by odd numbers. Assume the average is an int and never a decimal number. Return the average as output. Include this function in class UserMainCode.

Create a Class Main which would be used to read 2n numbers and build the hashmap. Call the static method present in UserMainCode.

Input and Output Format:

Input consists of a 2n+ 1 integers. The first integer specifies the value of n (essentially the hashmap size). The next pair of n numbers denote the key and value.

Output consists of an integer representing the average.

Refer sample output for formatting specifications.

Sample Input 1:

4

2

34

1

4

5

12

4

22

Sample Output 1:

8

Close

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.HashMap;

import java.util.Scanner;

public class Main

{

public static void Main(String args[])

{

Scanner sc=new Scanner(System.in);

int n=sc.nextInt();

HashMap<Integer,Integer> h1=new HashMap<Integer,Integer>();

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

{

h1.put(sc.nextInt(),sc.nextInt());

}

System.out.println(UserMainCode.getAverageOfOdd(h1));

}}

import java.util.HashMap;

import java.util.Iterator;

public class UserMainCode {

public static int getAverageOfOdd(HashMap<Integer,Integer>h1)

{

int av=0,c=0,s=0;

Iterator<Integer> it=h1.keySet().iterator();

while(it.hasNext())

{

int a=it.next();

if(a%2!=0)

{

int b=h1.get(a);

s=s+b;

c++;

}

}

av=s/c;

return av;

}}

93.Math Calculator (calculator)

Write a program that accepts three inputs, first two inputs are operands in int form and third one being one of the following five operators: +, -, \*, /, %. Implement calculator logic and return the result of the given inputs as per the operator provided. In case of division, Assume the result would be integer.

Include a class UserMainCode with a static method calculator which accepts two integers, one operand and returns the integer.

Create a Class Main which would be used to accept three integers and call the static method present in UserMainCode.

Input and Output Format:

Input consists of two integers and a character.

Output consists of a integer.

Refer sample output for formatting specifications.

Sample Input 1:

23

2

\*

Sample Output 1:

46

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 23/2/\*

public class UserMainCode {

public static int calculator(int a,int b,char c)

{

int a1=0;

if(c=='\*')

{

a1=a\*b;

}

else if(c=='+')

{

a1=a+b;

}

else if(c=='-')

{

a1=a-b;

}

else if(c=='/')

{

a1=a/b;

}

else if(c=='%')

{

a1=a%b;

}

return a1;

}}

94.Max Admissions

Write a program that reads details about number of admissions per year of a particular college, return the year which had maximum admissions. The details are stored in an arraylist with the first index being year and next being admissions count.

Include a class UserMainCode with a static method getYear which accepts a arraylist. The return type is an integer indicating the year of max admissions.

Create a Class Main which would be used to accept Input string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of 2n+1 values. The first value corresponds to size of the data (year & admissions). The next n pair of numbers contains the year and admissions count.

Output consists of an integer as mentioned in the problem statement.

Refer sample output for formatting specifications.

Sample Input 1:

4

2010

200000

2011

300000

2012

45000

2013

25000

Sample Output 1:

2011

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.util.ArrayList;

public class Main {

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

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

int n = Integer.parseInt(br.readLine());

ArrayList<Integer> inpList = new ArrayList<Integer>();

for(int i=0;i<n\*2;i++) {

inpList.add(Integer.parseInt(br.readLine()));

}

System.out.println(UserMainCode.getYear(inpList));

}

}

import java.util.ArrayList;

public class UserMainCode {

public static Integer getYear(ArrayList<Integer> inpList) {

int i,maxyear=0,maxcount=0;

for(i=0;i<inpList.size();i++)

{

int num=inpList.get(i);

if(maxcount<num)

maxcount=num;

}

maxyear=inpList.get((inpList.indexOf(maxcount))-1);

return maxyear;

}

}

95.Max Scorer

Write a program that performs the following actions:

Read n strings as input and stores them as an arraylist. The string consists of student information like name and obtained marks of three subjects. Eg: name-mark1-mark2-mark3 [suresh-70-47-12] The mark would range between 0 ? 100 (inclusive).

Write a function highestScorer which accepts these the arraylist and returns the name of the student who has scored the max marks. Assume the result will have only one student with max mark.

Include a class UserMainCode with the static method highestScorer which accepts the arraylist and returns the name (string) of max scorer.

Create a Class Main which would be used to read n strings into arraylist and call the static method present in UserMainCode.

Input and Output Format:

Input consists of 1 integer and n strings. The first integer denotes the size of the arraylist, the next n strings are score pattern described above.

Output consists of a string with the name of the top scorer.

Refer sample output for formatting specifications.

Sample Input 1:

3

sunil-56-88-23

bindul-88-70-10

john-70-49-65

Sample Output 1:

john

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.StringTokenizer;

public class Main {

public static String retrieveMaxScoredStudent(String[] s1){

int max=0;

String s4=null;

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

String s2=s1[i];

StringTokenizer t=new StringTokenizer(s2,"#");

String s3=t.nextToken();

int n1=Integer.parseInt(t.nextToken());

int n2=Integer.parseInt(t.nextToken());

int n3=Integer.parseInt(t.nextToken());

int n=n1+n2+n3;

if(n>max)

{

max=n;

s4=s3;

}

}

return s4;

}

public static void main(String[] args) {

String[] s1={"arun#12#12#12","deepak#13#12#12","puppy#12#11#12"};

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

}

}

96.Max Substring

Write a program to accept two string inputs. The first being a source string and second one a delimiter. The source string contains the delimiter at various locations. Your job is to return the substring with maximum number of characters. If two or more substrings have maximim number of characters return the substring which appears first. The size of the delimiter is 1.

Include a class UserMainCode with a static method extractMax which accepts the string. The return type (string) should be the max substring.

Create a Class Main which would be used to accept Input string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a source string and delimiter.

Output consists of a string.

Refer sample output for formatting specifications.

Sample Input 1:

delhi-pune-patna

-

Sample Output 1:

delhi

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.StringTokenizer;

public class UserMainCode {

public static String extractMax(String s,String s1){

int max=0;

String s3 = null;

StringTokenizer st=new StringTokenizer(s,"-");

while( st.hasMoreTokens())

{

String s2=st.nextToken();

int n=s2.length();

if(n>max)

{

max=n;

s3=s2;

}

}return s3;

}

}

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc=new Scanner(System.in);

String s=sc.nextLine();

String s1=sc.nextLine();

String res=UserMainCode.extractMax(s, s1);

System.out.println(res);

}

}

97.Max Vowels

Write a Program which fetches the word with maximum number of vowels. Your program should read a sentence as input from user and return the word with max number of vowels. In case there are two words of maximum length return the word which comes first in the sentence.

Include a class UserMainCode with a static method getWordWithMaximumVowels which accepts a string The return type is the longest word of type string.

Create a Class Main which would be used to accept two Input strings and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string with maximum size of 100 characters.

Output consists of a single string.

Refer sample output for formatting specifications.

Sample Input 1:

Appreciation is the best way to motivate

Sample Output 1:

Appreciation

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.StringTokenizer;

public class Main {

public static void main(String[] args) {

String s1 = "Appreciation is the best way to motivate aaaaeeeiii";

getWordWithMaximumVowels(s1);

}

public static void getWordWithMaximumVowels(String s1) {

int i = 0;

StringTokenizer st = new StringTokenizer(s1," ");

int len = 0;

int count = 0;

int count2 = 0;

String s6 = null;

while (st.hasMoreTokens()) {

String s5 = st.nextToken();

len = s5.length();

count=0;

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

if (s5.charAt(i) == 'a' || s5.charAt(i) == 'e'|| s5.charAt(i) == 'i' || s5.charAt(i) == 'o'|| s5.charAt(i) == 'u'

||s5.charAt(i) == 'A' ||s5.charAt(i) == 'E' ||s5.charAt(i) == 'I' ||s5.charAt(i) == 'O' ||s5.charAt(i) == 'U')

count++;

}

if (count > count2) {

count2 = count;

s6 = s5;

}

}

System.out.println(s6);

}

}

98.Maximum Difference

Write a program to read an integer array and find the index of larger number of the two adjacent numbers with largest difference. Print the index.

Include a class UserMainCode with a static method findMaxDistance which accepts an integer array and the number of elements in the array. The return type (Integer) should return index.

Create a Class Main which would be used to accept an integer array and call the static method present in UserMainCode.

Input and Output Format:

Input consists of n+1 integers, where n corresponds the size of the array followed by n integers.

Output consists of an Integer (index).

Refer sample output for formatting specifications.

Sample Input :

6

4

8

6

1

9

4

Sample Output :

4

[In the sequence 4 8 6 1 9 4 the maximum distance is 8 (between 1 and 9). The function should return the index of the greatest of two. In this case it is 9 (which is at index 4). output = 4.]

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public class UserMainCode {

public static int findMaxDistance(int[] n1){

int n2,n3=0,n4=0,i;

for(i=0;i<n1.length-1;i++){

n2=Math.abs(n1[i]-n1[i+1]);

if(n2>n3){

n3=n2;

n4=i+1; }}

return n4;

}

public static void main(String[] args) {

int[] n1={4,8,6,1,9,4};

System.out.println(findMaxDistance(n1));

}

}

99.Median Calculation(calculateMedian)

Write a program to accept an int array as input, and calculate the median of the same.

Median Calculation Procedure:

Sort the sequence of numbers.

The total number count is odd, Median will be the middle number.

The total number count is even, Median will be the average of two middle numbers, After calculating the average, round the number to nearest integer.

Include a class UserMainCode with a static method calculateMedian which accepts the int array. The return type is the integer which would be the median.

Create a Class Main which would be used to accept the integer array and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a an integer which denotes the size of the array followed by the array of integers.

Output consists of a integer.

Refer sample output for formatting specifications.

Sample Input 1:

7

1

2

1

4

7

1

2

Sample Output 1:

2

Sample Input 2:

6

52

51

81

84

60

88

Sample Output 2:

71

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.\*;

public class Main

{

public static void main(String[] args)

{

int n,m;

Scanner sin = new Scanner(System.in);

n = sin.nextInt();

int[] a1 = new int[n];

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

{

a1[i] = sin.nextInt();

}

System.out.println(""+UserMainCode.calculateMedian(a1));

}

}

import java.util.\*;

public class UserMainCode

{

public static int calculateMedian(int[] a)

{

Arrays.sort(a);

int length = a.length;

int result=0,mid=0,midNext=0;

if((length%2) != 0)

{

mid = (length/2)+1;

result = a[mid];

}

else

{

mid = length/2;

midNext = mid+1;

float add = a[mid-1]+a[midNext-1];

float div = add/2;

result = Math.round(div);

}

return result;

}

}

100.Middle of Array (getMiddleElement)

Write a program to read an integer array and return the middle element in the array. The size of the array would always be odd.

Include a class UserMainCode with a static method getMiddleElement which accepts a single integer array. The return type (integer) should be the middle element in the array.

Create a Class Main which would be used to accept Input array and call the static method present in UserMainCode.

Input and Output Format:

Input consists of n+1 integers. The first integer corresponds to n, the number of elements in the array. The next 'n' integers correspond to the elements in the array.

Output consists of a single Integer value.

Refer sample output for formatting specifications.

Assume that the maximum number of elements in the array is 19.

Sample Input 1:

5

1

5

23

64

9

Sample Output 1:

23

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 5/1/5/23/64/9

import java.util.Scanner;

public class kape {

public static void main(String []args){

Scanner scan=new Scanner(System.in);

int size=scan.nextInt();

int []a=new int[size];

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

{

a[i]=scan.nextInt();

}

System.out.println(kape1.display(a));

}

}

public class kape1

{public static int display(int[] a)

{

int y=a.length/2;

return a[y];

}

101.Month : Number of Days

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.\*;

public class MainDate{

public static void main(String []args){

Scanner sc=new Scanner(System.in);

int year=sc.nextInt();

int month=sc.nextInt();

System.out.println(main13.display(year, month));

}

}

import java.util.Calendar;

public class main13{

public static int display(int year,int month)

{

Calendar cal=Calendar.getInstance();

cal.set(Calendar.YEAR,year);

cal.set(Calendar.MONTH,month);

int DAY\_OF\_MONTH=cal.getActualMaximum(cal.DAY\_OF\_MONTH);

return DAY\_OF\_MONTH;

}}

102.Month Name (“getMonthName”)

Given a date as a string input in the format dd-mm-yy, write a program to extract the month and to print the month name in upper case.

Include a class UserMainCode with a static method ?getMonthName? that accepts a String argument and returns a String that corresponds to the month name.

Create a class Main which would get the String as input and call the static method getMonthName present in the UserMainCode.

The month names are {JANUARY, FEBRUARY, MARCH, APRIL, MAY, JUNE, JULY, AUGUST, SEPTEMBER, OCTOBER, NOVEMBER, DECEMBER}

Input and Output Format:

Input consists of a String.

Output consists of a String.

Sample Input:

01-06-82

Sample Output:

JUNE

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 01-06-82/JUNE

import java.text.ParseException;

import java.util.Scanner;

public class Main {

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

Scanner sc=new Scanner(System.in);

String s1=sc.nextLine();

System.out.println(User.calculateBornDay(s1));

}

}

import java.text.SimpleDateFormat;

import java.text.ParseException;

import java.util.Date;

public class User {

public static String calculateBornDay(String s1) throws ParseException

{

SimpleDateFormat sdf=new SimpleDateFormat("dd-MM-yy");

SimpleDateFormat sdf1=new SimpleDateFormat("MMMM");

Date d=sdf.parse(s1);

String s=sdf1.format(d);

return s.toUpperCase();

}

}

103.Name Shrinking (getFormatedString)

Write a program that accepts a string as input and converts the first two names into dot-separated initials and printa the output.

Input string format is 'fn mn ln'. Output string format is 'ln [mn's 1st character].[fn's 1st character]'

Include a class UserMainCode with a static method getFormatedString which accepts a string. The return type (String) should return the shrinked name.

Create a Class Main which would be used to accept Input String and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Output consists of a String.

Refer sample output for formatting specifications.

Sample Input:

Sachin Ramesh Tendulkar

Sample Output:

Tendulkar R.S

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Sachin Ramesh

import java.util.StringTokenizer;

public class Main {

public static void main(String[] args) {

String s1="Sachin Ramesh Tendulkar";

String z =UserMain.getvalues(s1);

System.out.println(z);

}

}

import java.util.StringTokenizer;

public class UserMain {

public static String getvalues(String s1) {

StringBuffer sb=new StringBuffer();

StringTokenizer st=new StringTokenizer(s1," ");

String s2=st.nextToken();

String s3=st.nextToken();

String s4=st.nextToken();

sb.append(s4).append(" ");

sb.append(s3.substring(0,1));

sb.append(".");

sb.append(s2.substring(0,1));

//System.out.println(sb);

return(sb.toString());

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

104.nCr (calculateNcr)

Write a program to calculate the ways in which r elements can be selected from n population, using nCr formula nCr=n!/r!(n-r)! where first input being n and second input being r.

Note1 : n! factorial can be achieved using given formula n!=nx(n-1)x(n-2)x..3x2x1.

Note2 : 0! = 1.

Example 5!=5x4x3x2x1=120

Include a class UserMainCode with a static method calculateNcr which accepts two integers. The return type (integer) should return the value of nCr.

Create a Class Main which would be used to accept Input elements and call the static method present in UserMainCode.

Input and Output Format:

Input consists of 2 integers. The first integer corresponds to n, the second integer corresponds to r.

Output consists of a single Integer.

Refer sample output for formatting specifications.

Sample Input 1:

4

3

Sample Output 1:

4

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 4/3/4

public class Main {

public static void main(String[] args) {

int n=4;

int r=3;

System.out.println(UserMain.getvalues(n,r));

}

}

public class UserMain{

public static int getvalues(int n, int r) {

int m;

int fact=1,fact1=1,fact2=1;

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

{

fact=fact\*i;

}

//System.out.println(fact);

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

{

fact1=fact1\*i;

}

//System.out.println(fact1);

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

{

fact2=fact2\*i;

}

//System.out.println(fact2);

int res=fact/(fact1\*fact2);

return res;

}

}

=====================================================================================

105.Negative String (“negativeString”)

Given a string input, write a program to replace every appearance of the word "is" by "is not".

If the word "is" is immediately preceeded or followed by a letter no change should be made to the string .

Include a class UserMainCode with a static method ?negativeString? that accepts a String arguement and returns a String.

Create a class Main which would get a String as input and call the static method negativeString present in the UserMainCode.

Input and Output Format:

Input consists of a String.

Output consists of a String.

Sample Input 1:

This is just a misconception

Sample Output 1:

This is not just a misconception

Sample Input 2:

Today is misty

Sample Output 2:

Today is not misty

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* This is just a misconception

import java.util.StringTokenizer;

public class Main {

public static void main(String[] args) {

String s1="this is just a misconception";

String z=UserMain.getvalues(s1);

System.out.println(z);

}

}

import java.util.StringTokenizer;

public class UserMain{

public static String getvalues(String s1)

{

StringBuffer sb=new StringBuffer();

StringTokenizer st=new StringTokenizer(s1," ");

while(st.hasMoreTokens())

{

String s2=st.nextToken();

if(s2.equals("is"))

{

String s3=s2.replace("is", "is not");

sb.append(s3);

//sb.append(" ");

}

else

sb.append(s2);

sb.append(" ");

}

sb.delete(sb.length()-1, sb.length());

//System.out.println(sb);

return sb.toString();

}

}

=========================================================================================

106.Next Year day

Given a date string in dd/mm/yyyy format, write a program to calculate the day which falls on the same date next year. Print the output in small case.

The days are sunday, monday, tuesday, wednesday, thursday, friday and saturday.

Include a class UserMainCode with a static method ?nextYearDay? that accepts a String and returns a String.

Create a class Main which would get a String as input and call the static method nextYearDay present in the UserMainCode.

Input and Output Format:

Input consists of a String.

Output consists of a String.

Sample Input:

13/07/2012

Sample Output:

saturday

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.ArrayList;import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Calendar;

import java.util.Date;

public class Main

{

public static void main(String[] args) {

String s1="13/07/2012";

String z=UserMain.getvalues(s1);

System.out.println(z);

} }

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Calendar;

import java.util.Date;

public class UserMain

{

public static String getvalues(String s1) {

SimpleDateFormat sdf=new SimpleDateFormat("dd/MM/yyyy");

sdf.setLenient(false);

try {

Date d1=sdf.parse(s1);

Calendar cal=Calendar.getInstance();

cal.setTime(d1);

cal.add(Calendar.YEAR, 1);

Date d2=cal.getTime();

SimpleDateFormat sdf1=new SimpleDateFormat("EEEEE");

String s=sdf1.format(d2);

//System.out.println(s);

return s;

}

catch (ParseException e)

{

e.printStackTrace();

}

return s1;

}}

===========================================================================================

107.Number Validation (validateNumber)

Write a program to read a string of 10 digit number , check whether the string contains a 10 digit number in the format XXX-XXX-XXXX where 'X' is a digit.

Include a class UserMainCode with a static method validateNumber which accepts a string as input .

The return type of the output should be 1 if the string meets the above specified format . In case the number does not meet the specified format then return -1 as output.

Create a class Main which would get the input as a String of numbers and call the static method validateNumber present in the UserMainCode.

Input and Output Format:

Input consists of a string.

Output is a string specifying the given string is valid or not .

Refer sample output for formatting specifications.

Sample Input 1:

123-456-7895

Sample Output 1:

Valid number format

Sample Input 2:

-123-12344322

Sample Output 2:

Invalid number format

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 123-456-7895

import java.util.\*;

public class Main {

public static void main(String[] args) {

Scanner s=new Scanner(System.in);

String pan=s.next();

int b=UserMain.panNumberValidation(pan);

if(b==1)

System.out.println("valid Number Format");

else

System.out.println("invalid Number Format");

}

}

public class UserMain{

public static int panNumberValidation(String input) {

int b=0;

if(input.matches("[0-9]{3}[-]{1}[0-9]{3}[-]{1}[0-9]{4}"))

{b=1;}

else

b=0;

return b;

}

}

=====================================================================================

108.Occurance Count

Write a program to read a string that contains a sentence and read a word. Check the number of occurances of that word in the sentence.

Include a class UserMainCode with a static method countWords which accepts the two strings. The return type is the integer giving the count.

Note: The check is case-sensitive.

Create a Class Main which would be used to accept the two strings and call the static method present in UserMainCode.

Input and Output Format:

Input consists of two strings.

Output consists of count indicating the number of occurances.

Refer sample output for formatting specifications.

Sample Input 1:

Hello world Java is best programming language in the world

world

Sample Output 1:

2

Sample Input 2:

hello world

World

Sample Output 2:

0

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.io.IOException;

import java.text.ParseException;

import java.util.\*;

public class Main {

public static void main(String[] args) throws IOException, ParseException {

// TODO Auto-generated method stub

String s1="hello world hello";

String s2="World";

int z=UserMain.getvalue(s1,s2);

System.out.println(z);

}

}

import java.util.StringTokenizer;

public class UserMain

{

public static int getvalue(String s1,String s2)

{

int c=0;

StringTokenizer t=new StringTokenizer(s1," ");

while(t.hasMoreTokens())

{

String s3=t.nextToken();

if(s3.equals(s2))

c++;

}

//System.out.println(c);

return c;

}

}

=======================================================================================

109.Odd Digit Sum (oddDigitSum)

Write a program to input a String array. The input may contain digits and alphabets (?de5g4G7R?). Extract odd digits from each string and find the sum and print the output.

For example, if the string is "AKj375A" then take 3+7+5=15 and not as 375 as digit.

Include a class UserMainCode with a static method oddDigitSum which accepts a string array and the size of the array. The return type (Integer) should return the sum.

Create a Class Main which would be used to accept Input Strings and call the static method present in UserMainCode.

Assume maximum length of array is 20.

Input and Output Format:

Input consists of an integer n, corresponds to the number of strings, followed by n Strings.

Output consists of an Integer.

Refer sample output for formatting specifications.

Sample Input :

3

cog2nizant1

al33k

d2t4H3r5

Sample Output :

15

(1+3+3+3+5)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 3/cog2nizant1/al33k/d2t4H3r5

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int s1=sc.nextInt();

String[] s2 = new String[s1];

for (int i = 0; i < s1; i++) {

s2[i] = sc.next();

}

System.out.println(UserMain.getSum(s2));

}}

public class UserMain {

public static int getSum(String[] s1) {

int sum=0;

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

for(int j=0;j<s1[i].length();j++){

char c=s1[i].charAt(j);

if(Character.isDigit(c)){

if(c%2!=0)

{

String t=String.valueOf(c);

int n=Integer.parseInt(t);

sum=sum+n; } }}

return sum;

}

}

==============================================================================

110.Palindrome - In Range

Write a program to input two integers, which corresponds to the lower limit and upper limit respectively, and find the sum of all palindrome numbers present in the range including the two numbers. Print the sum.

Include a class UserMainCode with a static method addPalindromes which accepts two integers. The return type (Integer) should return the sum if the palindromes are present, else return 0.

Create a Class Main which would be used to accept two integer and call the static method present in UserMainCode.

Note1 : A palindrome number is a number which remains same after reversing its digits.

Note2 : A single digit number is not considered as palindrome.

Input and Output Format:

Input consists of 2 integers, which corresponds to the lower limit and upper limit respectively.

Output consists of an Integer (sum of palindromes).

Refer sample output for formatting specifications.

Sample Input :

130

150

Sample Output :

272

(131+141 = 272)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.\*;

public class Main {

public static void main(String[] args) {

Scanner s=new Scanner(System.in);

System.out.println("enter the range:");

int n1=s.nextInt();

int n2=s.nextInt();

System.out.println("sum of palindrome nos.within given range is:"+UserMain.sumOfPalindromeNos(n1, n2));

}

}

import java.util.\*;

public class UserMain {

public static int sumOfPalindromeNos(int n1,int n2){

List<Integer> l1=new ArrayList<Integer>();

for(int i=n1;i<=n2;i++){

int r=0,n3=i;

while(n3!=0){

r=(r\*10)+(n3%10);

n3=n3/10; }

if(r==i)

l1.add(i); }

System.out.println(l1);

int s=0;

for(int i=0;i<l1.size();i++)

s+=l1.get(i);

return s;

} }

============================================================================

111.Palindrome & Vowels (checkPalindrome)

Write a program to check if a given string is palindrome and contains at least two different vowels.

Include a class UserMainCode with a static method checkPalindrome which accepts a string. The return type (integer) should be 1 if the above condition is satisfied, otherwise return -1.

Create a Class Main which would be used to accept Input string and call the static method present in UserMainCode.

Note ? Case Insensitive while considering vowel, i.e a & A are same vowel, But Case sensitive while considering palindrome i.e abc CbA are not palindromes.

Input and Output Format:

Input consists of a string with maximum size of 100 characters.

Output consists of a single Integer.

Refer sample output for formatting specifications.

Sample Input 1:

abceecba

Sample Output 1:

valid

Sample Input 2:

abcd

Sample Output 2:

invalid

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.Scanner;

public class Main {

public static void main(String[] args){

Scanner sc = new Scanner(System.in);

String s=sc.nextLine();

int z=UserMain.display(s);

if (z==1 )

System.out.println("valid");

else

System.out.println("invalid");

}

}

import java.util.Iterator;

import java.util.LinkedHashSet;

public class UserMain {

public static int display(String s) {

StringBuffer sb=new StringBuffer(s);

int k=0;

LinkedHashSet<Character>l1=new LinkedHashSet<Character>();

String s2=sb.reverse().toString();

if(s2.equals(s))

{

String s3=s2.toLowerCase();

for(int i=0;i<s3.length();i++)

{

l1.add(s3.charAt(i));

}

Iterator<Character> it=l1.iterator();

while(it.hasNext())

{

char a=it.next();

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

k++;

}

}

if(k>=2)

return 1;

else

return -1;

}}

============================================================================================

112.PAN Card

Write a program to read a string and validate PAN no. against following rules:

1. There must be eight characters.

2. First three letters must be alphabets followed by four digit number and ends with alphabet

3. All alphabets should be in capital case.

Print ?Valid? if the PAN no. is valid, else print ?Invalid?.

Include a class UserMainCode with a static method validatePAN which accepts a string. The return type (Integer) should return 1 if the string is a valid PAN no. else return 2.

Create a Class Main which would be used to accept a string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string, which corresponds to the PAN number.

Output consists of a string - "Valid" or "Invalid"

Refer sample output for formatting specifications.

Sample Input 1:

ALD3245E

Sample Output 1:

Valid

Sample Input 2:

OLE124F

Sample Output 2:

Invalid

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public class Main {

public static void main(String[] args) {

String s1="OLE124F";

int z=UserMain.getvalues(s1);

if (z==1)

System.out.println("Valid");

else

System.out.println("Invalid");

} }

public class UserMain

{

public static int getvalues(String s1) {

if(s1.matches("[A-Z]{3}[0-9]{4}[A-Z]{1}"))

{

//System.out.println(1);

return 1;

}

else

//System.out.println(-1);

return -1;

}

}

=======================================================================================

113.Password (validatePassword)

Given a String , write a program to find whether it is a valid password or not.

Validation Rule:

Atleast 8 characters

Atleast 1 number(1,2,3...)

Atleast 1 special character(@,#,%...)

Atleast 1 alphabet(a,B...)

Include a class UserMainCode with a static method ?validatePassword? that accepts a String argument and returns a boolean value. The method returns true if the password is acceptable. Else the method returns false.

Create a class Main which would get a String as input and call the static method validatePassword present in the UserMainCode.

Input and Output Format:

Input consists of a String.

Output consists of a String that is either ?Valid? or ?Invalid?.

Sample Input 1:

cts@1010

Sample Output 1:

Valid

Sample Input 2:

punitha3

Sample Output 2:

Invalid

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.\*;

public class Main {

public static void main(String[] args) {

Scanner s=new Scanner(System.in);

String s1=s.next();

boolean b=User.passwordValidation(s1);

if(b==true)

System.out.println("valid password");

else

System.out.println("not a valid password");

}

}

public class User{

public static boolean passwordValidation(String s1) {

boolean b=false,b1=false,b2=false;

if(s1.length()>=8)

b1=true;

if(b1==true)

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

if(Character.isAlphabetic(s1.charAt(i)) || Character.isDigit(s1.charAt(i)) || s1.charAt(i)=='#' || s1.charAt(i)=='@' || s1.charAt(i)=='%')

b2=true;

if(b2==true)

if(s1.contains("#") || s1.contains("@") || s1.contains("%"))

b=true;

return b;

}

114.Password Validation

Write a code get a password as string input and validate using the rules specified below. Apply following validations:

1. Minimum length should be 8 characters

2. Must contain any one of these three special characters @ or \_ or #

3. May contain numbers or alphabets.

4. Should not start with special character or number

5. Should not end with special character

Include a class UserMainCode with a static method validatePassword which accepts password string as input and returns an integer. The method returns 1 if the password is valid. Else it returns -1.

Create a class Main which would get the input and call the static method validatePassword present in the UserMainCode.

Input and Output Format:

Input consists of a string.

Output is a string Valid or Invalid.

Refer sample output for formatting specifications.

Sample Input 1:

ashok\_23

Sample Output 1:

Valid

Sample Input 2:

1980\_200

Sample Output 2:

Invalid

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.Scanner;

public class Main

{

public static void main(String[] args) {

Scanner s=new Scanner(System.in);

String s1=s.next();

int z=UserMain.display(s1);

if(z==1)

System.out.println("invalid");

else

System.out.println("valid");

}

}

public class UserMain {

public static int display(String password){

if(password.matches(".\*[0-9]{1,}.\*") && password.matches(".\*[@#$]{1,}.\*") && password.length()>=6

&& password.length()<=20)

{

return 1;

}

else

{

return -1;

}

}

}

}

==========================================================================================

115.Pattern Matcher

Write a program to read a string and check if it complies to the pattern 'CPT-XXXXXX' where XXXXXX is a 6 digit number. If the pattern is followed, then print TRUE else print FALSE.

Include a class UserMainCode with a static method CheckID which accepts the string. The return type is a boolean value.

Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Output should print TRUE or FALSE .

Refer sample output for formatting specifications.

Sample Input 1:

CPT-302020

Sample Output 1:

TRUE

Sample Input 2:

CPT123412

Sample Output 2:

FALSE

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.Scanner;

public class Main

{

public static void main(String[] args) {

Scanner s=new Scanner(System.in);

String s1=s.next();

System.out.println(UserMain.matchCharacter(s1));

}

}

import java.util.\*;

public class UserMain

{

public static boolean matchCharacter(String s)

{

boolean b=false;

if(s.matches("(CPT)[-]{1}[0-9]{6}"))

{

b=true;

}

else

{

b=false;

}

return b;

}

}

=======================================================================================

116.Perfect Number

Write a program to that takes a positive integer and returns true if the number is perfect number.

A positive integer is called a perfect number if the sum of all its factors (excluding the number itself, i.e., proper divisor) is equal to its value.

For example, the number 6 is perfect because its proper divisors are 1, 2, and 3, and 6=1+2+3; but the number 10 is not perfect because its proper divisors are 1, 2, and 5, and 1+2+5 is not equal to 10

Include a class UserMainCode with a static method getPerfection which accepts the number. The return type is boolean (true / false).

Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a integer.

Output consists of TRUE / FALSE.

Refer sample output for formatting specifications.

Sample Input 1:

28

Sample Output 1:

TRUE

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public class Main {

public static void main(String[] args) {

int n=28;

System.out.println(perfectNumber(n));

}

public static boolean perfectNumber(int n) {

int n1=0;

boolean b=false;

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

if(n%i==0)

n1+=i;

System.out.println(n1);

if(n1==n)

b=true;

return b;

}

}

=========================================================================================

117.Phone Number Validator (“validatePhoneNumber”)

Given a phone number as a string input, write a program to verify whether the phone number is valid using the following business rules:

-It should contain only numbers or dashes (-)

- dashes may appear at any position

-Should have exactly 10 digits

Include a class UserMainCode with a static method ?validatePhoneNumber? that accepts a String input and returns a integer. The method returns 1 if the phone number is valid. Else it returns 2.

Create a class Main which would get a String as input and call the static method validatePhoneNumber present in the UserMainCode.

Input and Output Format:

Input consists of a string.

Output consists of a string that is either 'Valid' or 'Invalid'

Sample Input 1:

265-265-7777

Sample Output 1:

Valid

Sample Input 2:

265-65-7777

Sample Output 1:

Invalid

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 265-265-7777

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner s=new Scanner(System.in);

String s1=s.next();

int z=UserMain.display(s1);

if(z==1)

System.out.println("Valid");

else

System.out.println("nvalid");}

}

import java.util.StringTokenizer;

public class UserMain

{

public static int display(String s){

int sum=0,u=0;

StringTokenizer st=new StringTokenizer(s,"-");

while(st.hasMoreTokens())

{

String s1=st.nextToken();

sum=sum+s1.length();

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

{

if(!Character.isDigit(s1.charAt(i)))

u=10;

}

}

if(u==0 && sum==10)

return 1;

else

return 2;

}}

=========================================================================================

118.Playing with String - I

Given a string array and non negative integer (n) apply the following rules.

1. Pick nth character from each String element in the String array and form a new String.

2. If nth character not available in a particular String in the array consider $ as the character.

3. Return the newly formed string.

Include a class UserMainCode with a static method formString which accepts the string and integer. The return type is the string formed based on rules.

Create a Class Main which would be used to accept the string and integer and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a an integer which denotes the size of the array followed by the array of strings and an integer (n).

Output consists of a string .

Refer sample output for formatting specifications.

Sample Input 1:

4

ABC

XYZ

EFG

MN

3

Sample Output 1:

CZG$

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.Scanner;

public class Main

{

public static void main(String[] arg)

{

Scanner s=new Scanner(System.in);

int n=Integer.parseInt(s.nextLine());

String[] sc=new String[n];

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

{

sc[i]=s.nextLine();

}

int a=Integer.parseInt(s.nextLine());

System.out.println(UserMainCode.get(n,sc,a));

}

}

public class UserMainCode {

public static String get(int n,String[] input,int a)

{

StringBuffer sb=new StringBuffer();

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

{

if(input[i].length()>=a)

{

String a1=input[i];

sb.append(a1.substring(a1.length()-1));

}

else

{

sb.append('$');

}

}

return sb.toString();

}

}

===========================================================================================

119.Playing with String - II

Write a program to accept a string array as input, convert all the elements into lowercase and sort the string array. Display the sorted array.

Include a class UserMainCode with a static method sortArray which accepts the string array. The return type is the string array formed based on requirement.

Create a Class Main which would be used to accept the string array and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a an integer which denotes the size of the array followed by the array of strings,

Output consists of a string array.

Refer sample output for formatting specifications.

Sample Input 1:

5

AAA

BB

CCCC

A

ABCDE

Sample Output 1:

a

aaa

abcde

bb

cccc

???????????????????????????????????????????????????????????????????????????????????????

import java.util.\*;

public class Main {

/\*\*

\* @param args

\* @throws ParseException

\*/

public static void main(String[] args) {

// TODO Auto-generated method stub

String s1[]={"AAA","BB","CCCC","A","ABCDE"};

String s2[]=new String[s1.length];

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

{

s2[i]=s1[i].toLowerCase();

}

Arrays.sort(s2);

for (int i = 0; i < s2.length; i++) {

System.out.println(s2[i]);

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

C

======================================================================================

120.Prefix Finder

Given a string array (s) with each element in the array containing 0s and 1s. Write a program to get the number of strings in the array where one String is getting as prefixed in other String in that array .

Example 1: Input: {10,101010,10001,1111} Output =2 (Since 10 is a prefix of 101010 and 10001)

Example 2: Input: {010,1010,01,0111,10,10} Output =3(01 is a prefix of 010 and 0111. Also, 10 is a prefix of 1010) Note: 10 is NOT a prefix for 10.

Include a class UserMainCode with a static method findPrefix which accepts the string array. The return type is the integer formed based on rules.

Create a Class Main which would be used to accept the string and integer and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a an integer indicating the number of elements in the string array followed by the array.

Output consists of a integer .

Refer sample output for formatting specifications.

Sample Input 1:

4

0

1

11

110

Sample Output 1:

3

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.Scanner;

public class Main {

public static void main(String args[]){

Scanner sc = new Scanner(System.in);

int n=Integer.parseInt(sc.nextLine());

String s[]=new String[n];

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

s[i]=sc.nextLine();

int m=UserMain.reverseString(s);

System.out.println(m);}

}

import java.util.ArrayList;

import java.util.Iterator;

import java.util.LinkedHashSet;

public class UserMain {

public static int reverseString (String s[]) {

LinkedHashSet<String>l1=new LinkedHashSet<String>();

ArrayList<String>a1=new ArrayList<String>();

int c=0;

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

l1.add(s[i]);

Iterator<String> it=l1.iterator();

while(it.hasNext())

{

a1.add(it.next());

}

for(int i=0;i<a1.size();i++)

{

String s2=a1.get(i);

for(int j=0;j<a1.size();j++)

{

String s3=a1.get(j);

if(i!=j&&s3.length()>s2.length())

{

String s4=s3.substring(0,s2.length());

if(s2.equals(s4))

c++;

}

}

}

//System.out.println(c);

return c;

}

}

=========================================================================================

121.Price Calculator - II

Write a small price calculator application with the below mentioned flow:

1. Read a value n indicating the total count of devices. This would be followed by the name and price of the device. The datatype for name would be String and price would be float.

2. Build a hashmap containing the peripheral devices with name as key and price as value.

3. Read a value m indicating the number of devices for which the price has to be calculated. This would be followed by device names.

4. For each devices mentioned in the array calcuate the total price.

5. You decide to write a function costEstimator which takes the above hashmap and array as input and returns the total price (float) as output with two decimal points. Include this function in class UserMainCode.

Create a Class Main which would be used to read details in step 1 and build the hashmap. Call the static method present in UserMainCode.

Input and Output Format:

Input consists of device details. The first number indicates the size of the devices. The next two values indicate the name,price.

This would be followed by m indicating the size of the device array. The next m values would be the device names.

Output consists of the total price in float.

Refer sample output for formatting specifications.

Sample Input 1:

3

Monitor

1200.36

Mouse

100.42

Speakers

500.25

2

Speakers

Mouse

Sample Output 1:

600.67

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.\*;

public class Main {

public static void main(String[] args) {

HashMap<String, String> m1=new HashMap<String, String>();

m1.put("monitor", "1200.36");

m1.put("mouse","100.42");

m1.put("speaker", "500.25");

String[] s={"speaker","mouse"};

System.out.println(UserMain.getTheTotalCostOfPheripherals(m1,s));

}

}

import java.util.\*;

public class UserMain {

public static float getTheTotalCostOfPheripherals(HashMap<String,String> m1,String[] s) {

Float f=(float) 0;

Iterator<String> i=m1.keySet().iterator();

while(i.hasNext()){

String s1=(String) i.next();

Float f1=Float.parseFloat(m1.get(s1));

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

if(s[j].equals(s1))

f+=f1; }

return f;

}}

==============================================================================================

122.Programming Logic

Write a Program that accepts three integer values (a,b,c) and returns their sum. However, if one of the values is 13 then it does not count towards the sum and the next number also does not count. So for example, if b is 13, then both b and c do not count.

Include a class UserMainCode with a static method getLuckySum which accepts three integers. The return type is integer representing the sum.

Create a Class Main which would be used to accept the input integers and call the static method present in UserMainCode.

Input and Output Format:

Input consists of three integers.

Output consists of a single integer.

Refer sample output for formatting specifications.

Sample Input 1:

1

2

3

Sample Output 1:

6

Sample Input 2:

1

2

13

Sample Output 2:

3

Sample Input 3:

13

3

8

Sample Output 3:

8

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.ArrayList;

public class Main{

public static void main(String[] args) {

int ip1=13,ip2=3,ip3=8;

System.out.println(UserMain.thirteenLapse(ip1,ip2,ip3));

}

}

import java.util.ArrayList;

public class UserMain{

public static int thirteenLapse(int ip1, int ip2, int ip3) {

ArrayList<Integer> l=new ArrayList<Integer>();

l.add(ip1);

l.add(ip2);

l.add(ip3);

int s=0;

for(int i=0;i<l.size();i++){

if(l.get(i)!=13)

s+=l.get(i);

if(l.get(i)==13)

i=i+1;}

return s;

}}

=============================================================================================

123.Proper Case

Write a program to read a string and convert the intial letter of each word to uppercase.

Include a class UserMainCode with a static method changeCase which accepts the string. The return type is the modified string.

Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Output consists of a string.

Refer sample output for formatting specifications.

Sample Input 1:

This is cognizant academy

Sample Output 1:

This Is Cognizant Academy

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.StringTokenizer;

public class Main {

public static void main(String[] args){

String s1="This is cognizant academy";

System.out.println(UserMain.capsStart(s1));

}

}

import java.util.StringTokenizer;

public class UserMain{

public static String capsStart(String s1){

StringBuffer s5=new StringBuffer();

StringTokenizer t=new StringTokenizer(s1," ");

while(t.hasMoreTokens()){

String s2=t.nextToken();

String s3=s2.substring(0,1);

String s4=s2.substring(1, s2.length());

s5.append(s3.toUpperCase()).append(s4).append(" "); }

return s5.toString();

}

}

============================================================================================

124.Regular Expression - 1(validate)

Given a string (s) apply the following rules.

1. String should be only four characters long.

2. First character can be an alphabet or digit.

3. Second character must be uppercase 'R'.

4. Third character must be a number between 0-9.

If all the conditions are satisifed then print TRUE else print FALSE.

Include a class UserMainCode with a static method validate which accepts the string. The return type is the boolean formed based on rules.

Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Output consists of TRUE or FALSE .

Refer sample output for formatting specifications.

Sample Input 1:

vR4u

Sample Output 1:

TRUE

Sample Input 2:

vRau

Sample Output 2:

FALSE

Sample Input 3:

vrau

Sample Output 3:

FALSE

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.Scanner;

public class Main {

public static void main(String []args){

Scanner sc=new Scanner(System.in);

String n=sc.nextLine();

System.out.println(UserMain.display(n));

}

}

public class UserMain

{

public static String display(String s)

{

String w="FALSE";

if(s.length()==4 && (Character.isDigit(s.charAt(0))||Character.isAlphabetic(s.charAt(0)))&&s.charAt(1)=='R')

{

if(Character.isDigit(s.charAt(2)))

w="TRUE";

}

return w;

}

}

=========================================================================================

125.Regular Expression – 2 (Age Validator)-(ValidateAge)

Given a string (s) apply the following rules.

1. String should be only four characters long.

2. First character can be an alphabet or digit.

3. Second character must be uppercase 'R'.

4. Third character must be a number between 0-9.

If all the conditions are satisifed then print TRUE else print FALSE.

Include a class UserMainCode with a static method validate which accepts the string. The return type is the boolean formed based on rules.

Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Output consists of TRUE or FALSE .

Refer sample output for formatting specifications.

Sample Input 1:

vR4u

Sample Output 1:

TRUE

Sample Input 2:

vRau

Sample Output 2:

FALSE

Sample Input 3:

vrau

Sample Output 3:

FALSE

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.HashMap;

import java.util.Iterator;

import java.util.Scanner;

public class main {

public static void main(String []args){

Scanner sc=new Scanner(System.in);

String age=sc.nextLine();

System.out.println(Usermaincode.display(age));

}

}

public class Usermaincode

{public static boolean display(String s)

{int c=0;

boolean q=false;

int n=s.length();

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

{

char a=s.charAt(i);

if(Character.isDigit(a))

c++;

}

if(c==s.length())

{

int age=Integer.parseInt(s);

if((age>=21)&&(age<=45))

q=true;

}

return q;

}

}

126.Regular Expression – 3 (Phone Validator)

Regular Expressions - III

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.\*;

public class UserMainCode

{

public static boolean matchCharacter(String s)

{

boolean b=false;

if(s.matches("[0]{2}[0-9]{8}"))

{

b=false;

}

else if(s.matches("[0-9]{10}"))

{

b=true;

}

return b;

}

}

127.Regular Expression - II (validateString)

Given a string (s) apply the following rules.

1. String consists of three characters only.

2. The characters should be alphabets only.

If all the conditions are satisifed then print TRUE else print FALSE.

Include a class UserMainCode with a static method validateString which accepts the string. The return type is the boolean formed based on rules.

Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Output consists of TRUE or FALSE .

Refer sample output for formatting specifications.

Sample Input 1:

AcB

Sample Output 1:

TRUE

Sample Input 2:

A2B

Sample Output 2:

FALSE

Close

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* AcB/TRUE

import java.util.Scanner;

public class Main {

public static void main(String []args){

Scanner sc=new Scanner(System.in);

String n=sc.nextLine();

System.out.println(UserMain.display(n));

}

}

public class UserMain

{public static boolean display(String s)

{

boolean b=true;

if(s.length()==3)

{

char c[]=s.toCharArray();

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

{

if(!Character.isAlphabetic(c[i]))

{

b=false;

break;

}

}

}

return b;

}

}

??????????????????????????????????????????????????

import java.util.Scanner;

public class Main {

public static void main(String []args){

Scanner sc=new Scanner(System.in);

String n=sc.nextLine();

boolean p=UserMainCode.display(n);

if (p=false){

System.out.println("FALSE");

}

else{

System.out.println("TRUE");

}

}

}

public class UserMainCode

{public static boolean display(String s)

{

boolean b=true;

if(s.length()==3)

{

char c[]=s.toCharArray();

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

{

if(!Character.isLetter(c[i]))

{

b=false;

break;

}

}

}

return b;

}

}

==========================================================================================

128.Regular Expression - III (validateString)

Given a string (s) apply the following rules.

1. String should not begin with a number.

If the condition is satisifed then print TRUE else print FALSE.

Include a class UserMainCode with a static method validateString which accepts the string. The return type is the boolean formed based on rules.

Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Output consists of TRUE or FALSE .

Refer sample output for formatting specifications.

Sample Input 1:

ab2

Sample Output 1:

TRUE

Sample Input 2:

72CAB

Sample Output 2:

FALSE

Close

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ab2/TRUE

import java.util.Scanner;

public class Main {

public static void main(String []args){

Scanner sc=new Scanner(System.in);

String n=sc.nextLine();

System.out.println(UserMain.display(n));

}

}

public class User

{

public static boolean display(String s)

{

boolean a=false;

if(Character.isDigit(s.charAt(0)))

{

a=false;

}

else

{

a=true;

}

return a;

}

}

==========================================================================================

129.Regular Expressions - III

Write a program to read two strings S1 & S2, compute the number of times that S2 appears in S1.

Include a class UserMainCode with a static method searchString which accepts the two strings. The return type is the integer giving the count.

Note: The check is case-insensitive.

Create a Class Main which would be used to accept the two strings and call the static method present in UserMainCode.

Input and Output Format:

Input consists of two strings.

Output consists of count indicating the number of occurances.

Refer sample output for formatting specifications.

Sample Input 1:

Catcowcat

cat

Sample Output 1:

2

Sample Input 2:

Catcowcat

catp

Sample Output 2:

0

Close

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.\*;

public class Main {

/\*\*

\* @param args

\* @throws ParseException

\*/

public static void main(String[] args) {

// TODO Auto-generated method stub

String s1="catcowcat";

String s2="cat";

int m=UserMain.getvalue(s1,s2);

System.out.println(m);

}

}

public class UserMain

{

public static int getvalue(String s1,String s2){

int c=0;

for(int i=0;i<(s1.length()-(s2.length()-1));i++)

{

if(s2.equalsIgnoreCase(s1.substring(i,i+s2.length())))

c++;

}

return c;

}

}

===========================================================================================

130.Regular Expression - III(passwordValidation)

Given a string (s) apply the following rules.

I)At least 8 characters must be present

II)At least one capital letter must be present

III)At least one small letter must be present

Iv)At least one special symbol must be present

V)At least one numeric value must be present

If the condition is satisifed then print valid else print invalid.

Include a class UserMainCode with a static method passwordValidation which accepts the string. The return type is the string.

Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Output consists of string (valid / invalid) .

Refer sample output for formatting specifications.

Sample Input 1:

Technology$1213

Sample Output 1:

valid

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.Scanner;

public class Main {

public static void main(String []args){

Scanner sc=new Scanner(System.in);

String n=sc.nextLine();

System.out.println(UserMain.display(n));

}

}

public class UserMain

{

public static String display(String s)

{

String s1;

if(s.matches(".\*[0-9]{1,}.\*") && s.matches(".\*[@#$]{1,}.\*") && s.length()>=8 && s.matches(".\*[A-Z]{1,}.\*") &&

s.matches(".\*[a-z]{1,}.\*"))

s1="Valid";

else

s1="Invalid";

return s1;

}}

============================================================================================

131.REMOVE 10'S(removeTens)

Write a program to read an integer array and remove all 10s from the array, shift the other elements towards left and fill the trailing empty positions by 0 so that the modified array is of the same length of the given array.

Include a class UserMainCode with a static method removeTens which accepts the number of elements and an integer array. The return type (Integer array) should return the final array.

Create a Class Main which would be used to read the number of elements and the input array, and call the static method present in UserMainCode.

Input and Output Format:

Input consists of n+1 integers, where n corresponds to size of the array followed by n elements of the array.

Output consists of an integer array (the final array).

Refer sample output for formatting specifications.

Sample Input :

5

1

10

20

10

2

Sample Output :

1

20

2

o

o

import java.util.Scanner;

public class main

{

public static void main(String[]args)

{

Scanner sc=new Scanner(System.in);

int size=sc.nextInt();

int[]m=new int[size];

int[]n=new int[size];

int j=0;

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

{

n[i]=sc.nextInt();

}

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

{

if(n[i]!=10)

{

m[j]=n[i];

j++;

}}

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

{

System.out.println(m[i]);

}

}

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

????????????????????????????????????????????????????????????????????

import java.util.Scanner;

public class Main

{

public static void main(String[]args)

{

Scanner sc=new Scanner(System.in);

int size=sc.nextInt();

int[]m=new int[size];

int[]n=new int[size];

int j=0;

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

{

n[i]=sc.nextInt();

}

Usercode.removetens(size,m,n);

}

}

import java.util.Scanner;

public class Usercode

{

public static void removetens (int size,int[]m,int[]n)

{

int j = 0;

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

{

if(n[i]!=10)

{

m[j]=n[i];

j++;

}}

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

{

System.out.println(m[i]);

}

}

}

132.Remove 3 Multiples(Remove 3 Multiples)

Write a program that accepts an ArrayList of integers as input and removes every 3rd element and prints the final ArrayList.

Suppose the given arrayList contains 10 elements remove the 3rd, 6th and 9th elements.

Include a class UserMainCode with a static method ?removeMultiplesOfThree? that accepts an ArrayList as arguement and returns an ArrayList.

Create a class Main which would get the required input and call the static method removeMultiplesOfThree present in the UserMainCode.

Input and Output Format:

The first line of the input consists of an integer n, that corresponds to the number of elements to be added in the ArrayList.

The next n lines consist of integers that correspond to the elements in the ArrayList.

Output consists of an ArrayList of integers.

Sample Input:

6

3

1

11

19

17

19

Sample Output

3

1

19

17

import java.util.ArrayList;

import java.util.Iterator;

import java.util.Scanner;

public class Main

{

public static void main(String []args){

Scanner sc=new Scanner(System.in);

ArrayList<Integer> al=new ArrayList<Integer>();

ArrayList<Integer> al1=new ArrayList<Integer>();

int n=Integer.parseInt(sc.nextLine());

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

{

al.add(sc.nextInt());

}

al1=UserMain.findFruitName(al);

Iterator it=al1.iterator();

while(it.hasNext())

{

System.out.println(it.next());

}

}

}

import java.util.ArrayList;

import java.util.Iterator;

import java.util.StringTokenizer;

public class UserMain

{

public static ArrayList<Integer> findFruitName(ArrayList<Integer> al)

{

ArrayList<Integer> al2=new ArrayList<Integer>();

for(int i=0;i<al.size();i++)

{

if((i+1)%3!=0)

al2.add(al.get(i));

}

return al2;

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

133.Remove Elements (removeElements) 5/a/bb/b/ccc/ddd

Write a program to remove all the elements of the given length and return the size of the final array as output. If there is no element of the given length, return the size of the same array as output.

Include a class UserMainCode with a static method removeElements which accepts a string array, the number of elements in the array and an integer. The return type (integer) should return the size of the final array as output.

Create a Class Main which would be used to accept Input String array and a number and call the static method present in UserMainCode.

Assume maximum length of array is 20.

Input and Output Format:

Input consists of a integers that corresponds to n, followed by n strings and finally m which corresponds to the length value.

Output consists of a single Integer.

Refer sample output for formatting specifications.

Sample Input 1:

5

a

bb

b

ccc

ddd

2

Sample Output 1:

4

===========================================

import java.util.\*;

public class Main

{

public static void main(String[] args)

{

Scanner sc=new Scanner(System.in);

int n=Integer.parseInt(sc.nextLine());

String[] a=new String[n];

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

a[i]=sc.nextLine();

int m=Integer.parseInt(sc.nextLine());

System.out.println(UserMainCode.display(a,m));

}}

import java.util.\*;

public class UserMainCode

{

public static int display(String[] a,int m){

int u=a.length;

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

{

if(a[i].length()==m)

u--;

}

return u;

}}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

134.Removing elements from HashMap (afterDelete)4/339/RON/1010/JONS

Given a HashMap as input, write a program to perform the following operation : If the keys are divisible by 3 then remove that key and its values and print the number of remaining keys in the hashmap.

Include a class UserMainCode with a static method afterDelete which accepts a HashMap as input.

The return type of the output is an integer which represents the count of remaining elements in the hashmap.

Create a class Main which would get the input and call the static method afterDelete present in the UserMainCode.

Input and Output Format:

First input corresponds to the size of hashmap

Input consists a HashMap

Output is an integer which is the count of remaining elements in the hashmap.

Refer sample output for formatting specifications.

Sample Input 1:

4

339

RON

1010

JONS

3366

SMITH

2020

TIM

Sample Output 1:

2

Sample Input 2:

5

1010

C2WE

6252

XY4E

1212

M2ED

7070

S2M41ITH

8585

J410N

Sample Output 2:

3

===================================================================================

import java.util.HashMap;

import java.util.Iterator;

public class Main {

/\*\*

\* @param args

\*/

public static void main(String[] args) {

HashMap<Integer, String>hm=new HashMap<Integer, String>();

hm.put(339,"RON");

hm.put(1010, "jons");

hm.put(3366, "yoo");

hm.put(2020, "world");

int m= UserMain.getvalues(hm);

System.out.println(m);}

}

import java.util.HashMap;

import java.util.Iterator;

public class UserMain

{

public static int getvalues(HashMap<Integer, String> hm) {

int count=0;

//HashMap<Integer, String>hm1=new HashMap<Integer, String>();

Iterator<Integer>itr=hm.keySet().iterator();

while(itr.hasNext())

{

int n=itr.next();

if(n%3!=0)

{

count++;

}

}

return count;

}

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

135.Removing Keys from HashMap (sizeOfResultandHashMap) 3/2/hi/4/hello/12/helloworld

Given a method with a HashMap as input. Write code to remove all the entries having keys multiple of 4 and return the size of the final hashmap.

Include a class UserMainCode with a static method sizeOfResultandHashMap which accepts hashmap as input.

The return type of the output is an integer which is the size of the resultant hashmap.

Create a class Main which would get the input and call the static method sizeOfResultandHashMap present in the UserMainCode.

Input and Output Format:

First input corresponds to the size of the hashmap.

Input consists of a hashmap.

Output is an integer which is the size of the hashmap.

Refer sample output for formatting specifications.

Sample Input 1:

3

2

hi

4

hello

12

hello world

Sample Output 1:

1

Sample Input 2:

3

2

hi

4

sdfsdf

3

asdf

Sample Output 2:

2

import java.util.\*;

public class Main {

public static void main(String[] args) {

HashMap<Integer, String>hm=new HashMap<Integer, String>();

hm.put(2,"hi");

hm.put(8, "hello");

hm.put(15, "yoo");

hm.put(12, "world");

hm.put(45, "ya");

int m=UserMain.getvalues(hm);

System.out.println(m);

}}

import java.util.HashMap;

import java.util.Iterator;

public class UserMain{

public static int getvalues(HashMap<Integer, String> hm) {

int count=0;

HashMap<Integer, String>hm1=new HashMap<Integer, String>();

Iterator<Integer>itr=hm.keySet().iterator();

while(itr.hasNext())

{

int n=itr.next();

if(n%4!=0)

{

count++;

}

}

return count;

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

136.Removing vowels from String (removeEvenVowels) commitment/cmmitmnt

Given a method with string input. Write code to remove vowels from even position in the string.

Include a class UserMainCode with a static method removeEvenVowels which accepts a string as input.

The return type of the output is string after removing all the vowels.

Create a Main class which gets string as an input and call the static method removeEvenVowels present in the UserMainCode.

Input and Output Format:

Input is a string .

Output is a string .

Assume the first character is at position 1 in the given string.

Sample Input 1:

commitment

Sample Output 1:

cmmitmnt

Sample Input 2:

capacity

Sample Output 2:

cpcty

public class Main {

public static void main(String[] args) {

String s1="capacity";

System.out.println(UserMain.removeEvenElements(s1));

}

}

public class UserMain{

public static String removeEvenElements(String s1) {

StringBuffer sb1=new StringBuffer();

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

if((i%2)==0)

sb1.append(s1.charAt(i));

else if((i%2)!=0)

if(s1.charAt(i)!='a' && s1.charAt(i)!='e' && s1.charAt(i)!='i' && s1.charAt(i)!='o' && s1.charAt(i)!='u')

if(s1.charAt(i)!='A' && s1.charAt(i)!='E' && s1.charAt(i)!='I' &&

s1.charAt(i)!='O' && s1.charAt(i)!='U')

sb1.append(s1.charAt(i));

return sb1.toString();

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

137.Repeat Front(repeatFirstThreeCharacters)

Given a string (s) and non negative integer (n) apply the following rules.

Display the first three characters as front.

If the length of the string is less than 3, then consider the entire string as front and repeat it n times.

Include a class UserMainCode with a static method repeatFirstThreeCharacters which accepts the string and integer. The return type is the string formed based on rules.

Create a Class Main which would be used to accept the string and integer and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string and integer.

Output consists of a string .

Refer sample output for formatting specifications.

Sample Input 1:

Coward

2

Sample Output 1:

CowCow

Sample Input 2:

So

3

Sample Output 2:

SoSoSo

import java.util.HashMap;

import java.util.Iterator;

import java.util.Scanner;

public class Main {

public static void main(String []args){

Scanner sc=new Scanner(System.in);

String s=sc.nextLine();

int n=Integer.parseInt(sc.nextLine());

System.out.println(UserMain.display(s,n));

}

}

import java.util.StringTokenizer;

public class UserMain

{public static String display(String s,int n)

{

StringBuffer sb=new StringBuffer();

StringBuffer sb1=new StringBuffer();

if(s.length()>3)

{ sb.append(s.substring(0,3));

s=sb.toString();

}

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

sb1.append(s);

return sb1.toString();

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

138.Repeating set of characters in a string (getString)Cognizant/3/Cognizantantantant

Get a string and a positive integer n as input .The last n characters should repeat the number of times given as second input.Write code to repeat the set of character from the given string.

Include a class UserMainCode with a static method getString which accepts a string and an integer n as input.

The return type of the output is a string with repeated n characters.

Create a class Main which would get the input and call the static method getString present in the UserMainCode.

Input and Output Format:

Input consists a string and a positive integer n.

Output is a string with repeated characters.

Refer sample output for formatting specifications.

Sample Input 1:

Cognizant

3

Sample Output 1:

Cognizantantantant

Sample Input 2:

myacademy

2

Sample Output 2:

myacademymymy

import java.util.\*;

public class Main {

public static void main(String[] args) {

Scanner s=new Scanner(System.in);

System.out.println("enter the String:");

String s1=s.nextLine();

int n=s.nextInt();

System.out.println("the lengthiest string is:"+UserMain.lengthiestString(s1,n));

}

}

import java.util.\*;

public class UserMain {

public static String lengthiestString(String s1,int n){

StringBuffer sb=new StringBuffer();

sb.append(s1);

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

{

sb.append(s1.substring(s1.length()-n,s1.length()));

}

return sb.toString();

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

139.Retirement (retirementEmployeeList)

Given an input as HashMap which contains key as the ID and dob as value of employees, write a program to find out employees eligible for retirement. A person is eligible for retirement if his age is greater than or equal to 60.

Assume that the current date is 01/01/2014.

Include a class UserMainCode with a static method ?retirementEmployeeList? that accepts a HashMap as input and returns a ArrayList. In this method, add the Employee IDs of all the retirement eligible persons to list and return the sorted list.

(Assume date is in dd/MM/yyyy format).

Create a class Main which would get the HashMap as input and call the static method retirementEmployeeList present in the UserMainCode.

Input and Output Format:

The first line of the input consists of an integer n, that corresponds to the number of employees.

The next 2 lines of the input consists of strings that correspond to the id and dob of employee 1.

The next 2 lines of the input consists of strings that correspond to the id and dob of employee 2.

and so on...

Output consists of the list of employee ids eligible for retirement in sorted order.

Sample Input :

4

C1010

02/11/1987

C2020

15/02/1980

C3030

14/12/1952

T4040

20/02/1950

Sample Output :

[C3030, T4040]

====================================================================

import java.util.LinkedHashMap;

import java.util.LinkedList;

import java.util.Scanner;

public class Main

{

public static void main(String args[]){

Scanner sc=new Scanner(System.in);

int n=Integer.parseInt(sc.nextLine());

LinkedHashMap<String,String>a1=new LinkedHashMap<String,String>();

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

{

a1.put(sc.nextLine(),sc.nextLine());

}

UserMain.display(a1);

}

}

import java.text.SimpleDateFormat;

import java.util.ArrayList;

import java.util.Calendar;

import java.util.Date;

import java.util.HashMap;

import java.util.Iterator;

import java.util.LinkedHashSet;

public class UserMain

{

public static ArrayList<Integer> display(HashMap<String,String>a1)

{

LinkedHashSet<String>a=new LinkedHashSet<String>();

SimpleDateFormat sdf=new SimpleDateFormat("dd/MM/yyyy");

int y=0;

Iterator<String>it=a1.keySet().iterator();

while(it.hasNext())

{

String s=it.next();

String s1=a1.get(s);

try{

Date d1=sdf.parse(s1);

Calendar c=Calendar.getInstance();

c.setTime(d1);

int y1=c.get(Calendar.YEAR);

int m1=c.get(Calendar.MONTH);

int da1=c.get(Calendar.DAY\_OF\_MONTH);

Date d2=new Date();

c.setTime(d2);

int y2=c.get(Calendar.YEAR);

int m2=c.get(Calendar.MONTH);

int da2=c.get(Calendar.DAY\_OF\_MONTH);

y=Math.abs(y1-y2);

if(m1==m2)

{

if(da1>da2)

y--;

}

else if(m1>m2)

y--;

if(y>=60)

a.add(s);

}

catch(Exception e)

{

e.printStackTrace();

}

}

System.out.println(a);

return null;

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

140.Reverse Split(reshape)

Write a program to read a string and a character, and reverse the string and convert it in a format such that each character is separated by the given character. Print the final string.

Include a class UserMainCode with a static method reshape which accepts a string and a character. The return type (String) should return the final string.

Create a Class Main which would be used to accept a string and a character, and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string and a character.

Output consists of a string (the final string).

Refer sample output for formatting specifications.

Sample Input:

Rabbit

-

Sample Output:

t-i-b-b-a-R

import java.util.Scanner;

import java.util.StringTokenizer;

public class Main {

public static void main(String[]args){

Scanner sc=new Scanner(System.in);

String s=sc.nextLine();

String m=UserMain.display(s);

System.out.println(m);

}}

import java.util.Scanner;

import java.util.StringTokenizer;

public class UserMain {

public static String display(String s){

StringBuffer sb=new StringBuffer();

sb.append(s.charAt(s.length()-1));

for(int i=s.length()-2;i>=0;i--)

{

sb.append('-');

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

}

return sb.toString();

}}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*===========================================================================

141.Reverse SubString (“reverseSubstring”)

Given a string, startIndex and length, write a program to extract the substring from right to left. Assume the last character has index 0.

Include a class UserMainCode with a static method ?reverseSubstring? that accepts 3 arguments and returns a string. The 1st argument corresponds to the string, the second argument corresponds to the startIndex and the third argument corresponds to the length.

Create a class Main which would get a String and 2 integers as input and call the static method reverseSubstring present in the UserMainCode.

Input and Output Format:

The first line of the input consists of a string.

The second line of the input consists of an integer that corresponds to the startIndex.

The third line of the input consists of an integer that corresponds to the length of the substring.

Sample Input:

rajasthan

2

3

Sample Output:

hts

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Rajasthan/2/3

public class Main {

public static void main(String[] args) {

String input1="Rajasthan";

int input2=2, input3=3;

System.out.println(UserMain.retrieveString(input1,input2,input3));

}

}

public class UserMain {

public static String retrieveString(String input1, int input2, int input3) {

StringBuffer sb=new StringBuffer(input1);

sb.reverse();

String output=sb.substring(input2, input2+input3);

return output;

}

}

============================================================================================

142.Reversing a Number (reverseNumber)

Write a program to read a positive number as input and to get the reverse of the given number and return it as output.

Include a class UserMainCode with a static method reverseNumber which accepts a positive integer .

The return type is an integer value which is the reverse of the given number.

Create a Main class which gets the input as a integer and call the static method reverseNumber present in the UserMainCode

Input and Output Format:

Input consists of a positive integer.

Output is an integer .

Refer sample output for formatting specifications.

Sample Input 1:

543

Sample Output 1:

345

Sample Input 1:

1111

Sample Output 1:

1111

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 543/345

import java.util.Scanner;

public class Main

{

public static void main(String args[]){

Scanner sc=new Scanner(System.in);

int n=sc.nextInt();

int m=UserMain.display(n);

System.out.println(m);

}}

public class UserMain {

public static int display(int number)

{

int d=number,rev=0,rem=0;

while(d!=0)

{

rem=d%10;

rev=rev\*10+rem;

d=d/10;

}

return rev;

}

}

=================================================================================

143.Scores (checkScores)

Write a program to read a integer array of scores, if 100 appears at two consecutive locations return true else return false.

Include a class UserMainCode with a static method checkScores which accepts the integer array. The return type is boolean.

Create a Class Main which would be used to accept the integer array and call the static method present in UserMainCode.

Input and Output Format:

Input consists of an integer n which is the number of elements followed by n integer values.

Output consists of a string that is either 'TRUE' or 'FALSE'.

Refer sample output for formatting specifications.

Sample Input 1:

3

1

100

100

Sample Output 1:

TRUE

Sample Input 2:

3

100

1

100

Sample Output 2:

FALSE

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 3/1/100/100

import java.util.Scanner;

public class Main

{

public static void main (String[] args)

{

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

int[] arr = new int[n];

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

arr[i] = sc.nextInt();

}

System.out.println(UserMain.consecutiveNum(arr, n));

}

}

public class UserMain{

public static boolean consecutiveNum(int arr[], int n){

boolean b = false;

for(int i=0;i<n-1;i++){

if(arr[i] == 100){

if(arr[i+1] == 100){

b = true;

break;

}

}

}

return b;

}}

===========================================================================================

144.Sequence in Array

Write a program to accept an int array as input, and check if [1,2,3] appears somewhere in the same sequence.

Include a class UserMainCode with a static method searchSequence which accepts the int array. The return type is a boolean which returns true or false.

Create a Class Main which would be used to accept the integer array and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a an integer which denotes the size of the array followed by the array of integers.

Output should print true or false.

Refer sample output for formatting specifications.

Sample Input 1:

9

11

-2

5

1

2

3

4

5

6

Sample Output 1:

TRUE

Sample Input 2:

6

-2

5

1

3

2

6

Sample Output 2:

FALSE

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public class Main

{

public static void main(String[] args)

{

//int[] a={2,1,4,1,2,3,6,1,2,3};

int[] a={1,2,1,3,4,5,8};

System.out.println(UserMain.sequenceInArray(a));

}

}public class UserMain

{

public static boolean sequenceInArray(int[] a)

{

boolean b = false;

for(int i = 0 ; i< a.length-3; i++)

{

if(a[i]==1 && a[i+1]==2 && a[i+2]==3)

b = true;

}

return b;

} }

}

=========================================================================================

145.Sequence Sum

Write a program to read a non-negative integer n, and find sum of fibonanci series for n number..

Include a class UserMainCode with a static method getFibonacciSum which accepts the integer value. The return type is integer.

The fibonacci seqence is a famous bit of mathematics, and it happens to have a recursive definition.

The first two values in the sequnce are 0 and 1.

Each subsequent value is the sum of the previous two values, so the whole seqence is 0,1,1,2,3,5 and so on.

You will have to find the sum of the numbers of the Fibonaaci series for a given int n.

Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a integer.

Output consists of integer.

Refer sample output for formatting specifications.

Sample Input 1:

5

Sample Output 1:

7

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.ArrayList;

import java.util.Scanner;

public class Main {

public static int sumOfFibonacci(int n){

int a=0,b=1,c=0,d=1;

for(int i=3;i<=n;i++){

c=a+b;

a=b; b=c;

d=d+c;

}

return d;

}

public static void main(String[] args) {

Scanner s=new Scanner(System.in);

int n=s.nextInt();

System.out.println(sumOfFibonacci(n));

}

}

146.Shift Left (shiftLeft)

Write a program to read a integer array of scores, and return a version of the given array where all the 5's have been removed. The remaining elements should shift left towards the start of the array as needed,

and the empty spaces at the end of the array should be filled with 0.

So {1, 5, 5, 2} yields {1, 2, 0, 0}.

Include a class UserMainCode with a static method shiftLeft which accepts the integer array. The return type is modified array.

Create a Class Main which would be used to accept the integer array and call the static method present in UserMainCode.

Input and Output Format:

Input consists of an integer n which is the number of elements followed by n integer values.

Output consists of modified array.

Refer sample output for formatting specifications.

Sample Input 1:

7

1

5

2

4

5

3

5

Sample Output 1:

1

2

4

3

0

0

0

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.Scanner;

public class usercc

{

public static void main(String[]args)

{

Scanner sc=new Scanner(System.in);

int size=sc.nextInt();

int[]m=new int[size];

int[]n=new int[size];

int j=0;

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

{

n[i]=sc.nextInt();

}

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

{

if(n[i]!=5)

{

m[j]=n[i];

j++;

}}

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

{

System.out.println(m[i]);

}

}

}

??????????????????????????????????????????????????????????????

import java.util.Scanner;

public class Main

{

public static void main(String[]args)

{

Scanner scan=new Scanner(System.in);

int size=scan.nextInt();

int[]m=new int[size];

int []out=new int[size];

//int[]n=new int[size];

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

{m[i]=scan.nextInt();

}

out= Usercode.ShiftRight(m,size);

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

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

}}

import java.util.Scanner;

public class Usercode{

public static int[] ShiftRight(int[]m,int size)

{

int[]n=new int[size];

int j=0;

//int size=0;

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

{

if(m[i]!=5)

{

n[j]=m[i];

j++;

}}

/\*

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

{

System.out.println(n[i]);

}

\*/

return n;}

}

147.Simple String Manipulation - II

Write a program to read a string and return an integer based on the following rules.

If the first word and the last word in the String match, then return the number of characters in the word else return sum of the characters in both words. Assume the Strings to be case - sensitive.

Include a class UserMainCode with a static method calculateWordSum which accepts a string. The return type (integer) should be based on the above rules.

Create a Class Main which would be used to accept Input string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string with maximum size of 100 characters.

Output consists of a string.

Refer sample output for formatting specifications.

Sample Input 1:

COGNIZANT TECHNOLOGY SOLUTIONS COGNIZANT

Sample Output 1:

9

Sample Input 2:

HOW ARE YOU

Sample Output 2:

6

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.util.ArrayList;

import java.util.Scanner;

public class Main {

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

Scanner sc=new Scanner(System.in);

String inpList=sc.nextLine();

System.out.println(UserMainCode.calculateWordSum(inpList));

}

}

public class UserMainCode {

public static int calculateWordSum(String inp) {

int count=0;

String st[]=inp.split(" ");

String s1=st[0];

String slst=st[st.length-1];

if(s1.equals(slst))

{

count=s1.length();

}

else

{

count=s1.length()+slst.length();

}

return count;

}

}

148.Simple String Manipulation (getString)

Write a program to read a string and return a modified string based on the following rules.

Return the String without the first 2 chars except when

keep the first char if it is 'j'

keep the second char if it is 'b'.

Include a class UserMainCode with a static method getString which accepts a string. The return type (string) should be the modified string based on the above rules. Consider all letters in the input to be small case.

Create a Class Main which would be used to accept Input string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string with maximum size of 100 characters.

Output consists of a string.

Refer sample output for formatting specifications.

Sample Input 1:

hello

Sample Output 1:

llo

Sample Input 2:

java

Sample Output 2:

jva

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Hello/llo

import java.util.Scanner;

public class useer

{

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

String s=sc.nextLine();

System.out.println(useerm.display(s));

}}

public class useerm

{

public static String display(String s)

{

StringBuffer sb=new StringBuffer();

char a=s.charAt(0);

char b=s.charAt(1);

if(a!='j'&& b!='b')

sb.append(s.substring(2));

else if(a=='j' && b!='b')

sb.append("j").append(s.substring(2));

else if(a!='j' && b=='b')

sb.append(s.substring(1));

else

sb.append(s.substring(0));

return sb.toString();

}

}

149.Sorted Array(orderElements)

Write a program to read a string array, remove duplicate elements and sort the array.

Note:

The check for duplicate elements must be case-sensitive. (AA and aa are NOT duplicates)

While sorting, words starting with upper case letters takes precedence.

Include a class UserMainCode with a static method orderElements which accepts the string array. The return type is the sorted array.

Create a Class Main which would be used to accept the string arrayand integer and call the static method present in UserMainCode.

Input and Output Format:

Input consists of an integer n which is the number of elements followed by n string values.

Output consists of the elements of string array.

Refer sample output for formatting specifications.

Sample Input 1:

6

AAA

BBB

AAA

AAA

CCC

CCC

Sample Output 1:

AAA

BBB

CCC

Sample Input 2:

7

AAA

BBB

aaa

AAA

Abc

A

b

Sample Output 2:

A

AAA

Abc

BBB

aaa

b

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.\*;

public class Main

{

public static void main(String[] args)

{

int n;

Scanner sin = new Scanner(System.in);

n = sin.nextInt();

String[] a1 = new String[n];

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

{

a1[i] = sin.next();

}

a1 = UserMainCode.orderElements(a1);

for(int i=0;i<a1.length;i++)

System.out.println(""+a1[i]);

}

}

import java.util.\*;

public class UserMainCode

{

public static String[] orderElements(String[] arr)

{

HashSet<String> al=new HashSet<String>();

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

{

al.add(arr[i]);

}

Iterator<String> itr=al.iterator();

arr = new String[al.size()];

int i =0 ;

while(itr.hasNext()){

arr[i++] = itr.next();

}

Arrays.sort(arr);

return arr;

}

}

150.Start Case

public class Main {

public static void main(String[] args){

String s1="Now is the time to act!";

System.out.println(UserMainCode.capsStart(s1));

}

}

import java.util.StringTokenizer;

public class UserMainCode {

public static String capsStart(String s1){

StringBuffer s5=new StringBuffer();

StringTokenizer t=new StringTokenizer(s1," ");

while(t.hasMoreTokens()){

String s2=t.nextToken();

String s3=s2.substring(0,1);

String s4=s2.substring(1, s2.length());

s5.append(s3.toUpperCase()).append(s4).append(" "); }

return s5.toString();

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

151.State ID generator

import java.util.HashMap;

import java.util.Map;

public class Main {

public static void main(String[] args)

{

String[] s1={"goa","kerala","gujarat"};

HashMap<String,String> x= new HashMap<String,String>();

x= UserMainCode.getStateID(s1);

for(Map.Entry<String, String> ans: x.entrySet())

{

System.out.println(ans.getKey()+":"+ans.getValue());

}

}

}

import java.util.\*;

public class UserMainCode {

public static HashMap<String, String> getStateID(String[] s1)

{

HashMap<String, String> hm = new HashMap<String, String>();

ArrayList<String> lst1 = new ArrayList<String>();

ArrayList<String> lst2 = new ArrayList<String>();

for(String s : s1)

lst1.add(s.toUpperCase().substring(0,3));

for(String s : s1)

lst2.add(s);

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

{

hm.put(lst1.get(i),lst2.get(i));

}

//System.out.println(map);

return hm;

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

152.States and Capitals(getCapital)

Write a program that construts a hashmap with ?state? as key and ?capital? as its value. If the next input is a state, then it should return capital$state in lowercase.

Include a class UserMainCode with a static method getCapital which accepts a hashmap. The return type is the string as given in the above statement

Create a Class Main which would be used to accept Input string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of 2n+2 values. The first value corresponds to size of the hashmap. The next n pair of numbers contains the state and capital. The last value consists of the ?state? input.

Output consists of a string as mentioned in the problem statement.

Refer sample output for formatting specifications.

Sample Input 1:

3

Karnataka

Bangaluru

Punjab

Chandigarh

Gujarat

Gandhinagar

Punjab

Sample Output 1:

chandigarh$punjab

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.HashMap;

import java.util.Map;

public class UserMainCode

{

public static String display(HashMap<String,String> hm,String sa)

{

for(Map.Entry m1:hm.entrySet())

{

String ss=(String)m1.getKey();

if(ss.equalsIgnoreCase(sa))

{

return (m1.getValue()+"$"+m1.getKey());

}

}

return sa;

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

153.String Concatenation (concatstring)

Write code to get two strings as input and If strings are of same length simply append them together and return the final string. If given strings are of different length, remove starting characters from the longer string so that both strings are of same length then append them together and return the final string.

Include a class UserMainCode with a static method concatstring which accepts two string input.

The return type of the output is a string which is the concatenated string.

Create a class Main which would get the input and call the static method concatstring present in the UserMainCode.

Input and Output Format:

Input consists of two strings.

Output is a string.

Refer sample output for formatting specifications.

Sample Input 1:

Hello

hi

Sample Output 1:

lohi

Sample Input 2:

Hello

Delhi

Sample Output 2:

HelloDelhi

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public static void getvalues(String s1, String s2) {

StringBuffer sb=new StringBuffer();

int l1=s1.length();

int l2=s2.length();

if(l1==l2)

{

sb.append(s1).append(s2);

}

else if(l1>l2)

{

sb.append(s1.substring(s1.length()-s2.length(),s1.length())).append(s2);

}

else if(l1<l2)

{

sb.append(s1).append(s2.substring(s2.length()-s1.length(),s2.length()));

}

System.out.println(sb);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

154.String Encryption (encrypt)

Given an input as string and write code to encrypt the given string using following rules and return the encrypted string:

1. Replace the characters at odd positions by next character in alphabet.

2. Leave the characters at even positions unchanged.

Note:

- If an odd position charater is 'z' replace it by 'a'.

- Assume the first character in the string is at position 1.

Include a class UserMainCode with a static method encrypt which accepts a string.

The return type of the output is the encrypted string.

Create a Main class which gets string as an input and call the static method encrypt present in the UserMainCode.

Input and Output Format:

Input is a string .

Output is a string.

Sample Input 1:

curiosity

Sample Output 1:

dusipsjtz

Sample Input 2:

zzzz

Sample Output 2:

azaz

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public static String stringFormatting(String s1) {

StringBuffer sb=new StringBuffer();

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

char c=s1.charAt(i);

if(i%2==0){

if(c==122)

c=(char) (c-25);

else{

c=(char) (c+1);}

sb.append(c);}

else

sb.append(c);}

return sb.toString();

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

155.String Finder (“stringFinder”)

Given three strings say Searchstring, Str1 and Str2 as input, write a program to find out if Str2 comes after Str1 in the Searchstring.

Include a class UserMainCode with a static method ?stringFinder? that accepts 3 String arguments and returns an integer. The 3 arguments correspond to SearchString, Str1 and Str2. The function returns 1 if Str2 appears after Str1 in the Searchtring. Else it returns 2.

Create a class Main which would get 3 Strings as input and call the static method stringFinder present in the UserMainCode.

Input and Output Format:

Input consists of 3 strings.

The first input corresponds to the SearchString.

The second input corresponds to Str1.

The third input corresponds to Str2.

Output consists of a string that is either ?yes? or ?no?

Sample Input 1:

geniousRajKumarDev

Raj

Dev

Sample Output 1:

yes

Sample Input 2:

geniousRajKumarDev

Dev

Raj

Sample Output 2:

no

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public static void getvalues(String s1, String s2, String s3) {

if(s1.contains(s2)&& s1.contains(s3))

{

if(s1.indexOf(s2)<s1.indexOf(s3))

{

System.out.println(1);

}

else

System.out.println(2);

}

}}

??????????????????????????????????????????????????????????????????

import java.util.\*;

public class Main

{

public static void main(String[] args)

{

Scanner scan= new Scanner(System.in);

String s1=scan.nextLine();

String s2=scan.nextLine();

String s3=scan.nextLine();

String z=UserMainCode.getvalues(s1,s2,s3);

}

}

public class UserMainCode{

public static String getvalues(String s1, String s2, String s3) {

if(s1.contains(s2)&& s1.contains(s3))

{

if(s1.indexOf(s2)<s1.indexOf(s3))

{

System.out.println("yes");

}

else

System.out.println("no");

}

return s3;

}}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

156.String Occurances - II

Obtain two strings S1,S2 from user as input. Your program should count the number of times S2 appears in S1.

Return the count as output. Note - Consider case.

Include a class UserMainCode with a static method getSubstring which accepts two string variables. The return type is the count.

Create a Class Main which would be used to accept two Input strings and call the static method present in UserMainCode.

Input and Output Format:

Input consists of two strings with maximum size of 100 characters.

Output consists of an integer.

Refer sample output for formatting specifications.

Sample Input 1:

catcowcat

cat

Sample Output 1:

2

Sample Input 2:

catcowcat

CAT

Sample Output 2:

0

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.\*;

public class UserMainCode {

public static void main(String[]args){

Scanner sc=new Scanner(System.in);

String s=sc.nextLine();

String s1=sc.nextLine();

char []c=s.toCharArray();

char[] c1=s1.toCharArray();

int count=0,result=0;

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

{

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

{

if(c[i]==c1[j])

{

i++;

count++;

if(i>=s.length())

break;

}

}

if(count==s1.length())

{

result++;

i=i-1;

}

count=0;

}

System.out.print(result);

}}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

157.String Occurences

Obtain two strings from user as input. Your program should count the number of occurences of second word of second sentence in the first sentence.

Return the count as output. Note - Consider case.

Include a class UserMainCode with a static method countNoOfWords which accepts two string variables. The return type is the modified string.

Create a Class Main which would be used to accept two Input strings and call the static method present in UserMainCode.

Input and Output Format:

Input consists of two strings with maximum size of 100 characters.

Output consists of a single string.

Refer sample output for formatting specifications.

Sample Input 1:

abc bcd abc bcd abc abc

av abc

Sample Output 1:

4

Sample Input 2:

ABC xyz AAA

w abc

Sample Output 2:

0

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public static void getvalues(String s1, String s2) {

int count=0;

StringTokenizer st=new StringTokenizer(s2," ");

String s3=st.nextToken();

String s4=st.nextToken();

//System.out.println(s4);

StringTokenizer st1=new StringTokenizer(s1," ");

while(st1.hasMoreTokens())

{

String s5=st1.nextToken();

if(s4.equals(s5))

{

count++;

}

}

System.out.println(count);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

158.String Processing - III (moveX)

Write a program to read a string where all the lowercase 'x' chars have been moved to the end of the string.

Include a class UserMainCode with a static method moveX which accepts the string. The return type is the modified string.

Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Output consists of a string.

Refer sample output for formatting specifications.

Sample Input 1:

xxhixx

Sample Output 1:

hixxxx

Sample Input 2:

XXxxtest

Sample Output 2:

XXtestxx

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public static String removalOfx(String input) {

StringBuffer sb=new StringBuffer(input);

int j=0;

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

if(sb.charAt(i)=='x')

{

sb.deleteCharAt(i);

j++;

}

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

sb.append('x');

return sb.toString();

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

159.String Processing - IV (getStringUsingNthCharacter)

Write a program to read a string and also a number N. Form a new string starting with 1st character and with every Nth character of the given string. Ex - if N is 3, use chars 1, 3, 6, ... and so on to form the new String. Assume N>=1.

Include a class UserMainCode with a static method getStringUsingNthCharacter which accepts the string and the number n. The return type is the string as per the problem statement.

Create a Class Main which would be used to accept the string and integer and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string and integer.

Output consists of a string.

Refer sample output for formatting specifications.

Sample Input 1:

HelloWorld

2

Sample Output 1:

HelWrd

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public static String deletingtheCharOccuringTwice(String input1, int input2)

{

StringBuffer sb=new StringBuffer(input1);

int c=1;

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

{

c=1;

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

{

if(sb.charAt(i)==sb.charAt(j))

c++;

}

if(c>input2)

{

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

{

if(sb.charAt(i)==sb.charAt(j))

{

sb.deleteCharAt(j);

j--;

}

}

}

if(c==input2)

{

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

{

if(sb.charAt(i)==sb.charAt(j))

sb.deleteCharAt(j);

}

sb.deleteCharAt(i);

i--;

}

}

return sb.toString();

}

}

160.String processing – Long + Short + Long

Obtain two strings S1,S2 from user as input. Your program should form a string of ?long+short+long?, with the shorter string inside of the longer String.

Include a class UserMainCode with a static method getCombo which accepts two string variables. The return type is the string.

Create a Class Main which would be used to accept two Input strings and call the static method present in UserMainCode.

Input and Output Format:

Input consists of two strings with maximum size of 100 characters.

Output consists of an string.

Refer sample output for formatting specifications.

Sample Input 1:

Hello

Hi

Sample Output 1:

HelloHiHello

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.StringTokenizer;

public class User {

public static void main(String[] args){

String s1="Hi";

String s2="Hello";

System.out.println(capsStart(s1,s2));

}

public static String capsStart(String s1,String s2){

StringBuffer s5=new StringBuffer();

int q=s1.length();

int w=s2.length();

if(q>w)

{

s5.append(s1).append(s2).append(s1);

}

else

{

s5.append(s2).append(s1).append(s2);

}

return s5.toString();

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

161.String Processing - MixMania (checkPattern)

Write a program to read a string and check if it starts with '\_ix' where '\_' is any one char(a-z, A-Z, 0-9).

If specified pattern is found return true else false.

Include a class UserMainCode with a static method checkPattern which accepts the string. The return type is TRUE / FALSE.

Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Output consists of TRUE / FALSE.

Refer sample output for formatting specifications.

Sample Input 1:

Mix Mania

Sample Output 1:

TRUE

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.StringTokenizer;

public class UserMainCode{

public static boolean checkPattern(String s1){

StringBuffer sb=new StringBuffer();

String s2=s1.substring(0,3);

int a=0,b=0,c=0;

boolean bul=false;

char c21=s2.charAt(0);

if (Character.isDigit(c21)||Character.isLetter(c21))

{

a=1;

}

if(a==1)

{

char c1=s2.charAt(1);

char c2=s2.charAt(2);

if(c1=='i'){

b=1;

}

if(c2=='x')

{c=1;

}

}

if(a==1&&b==1&&c==1)

{

bul= true;

}else

{

bul=false;

}

return bul;

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

162.String Processing - TrimCat (getAlternateChars)

Write a program to read a string and return a new string which is made of every alternate characters starting with the first character. For example NewYork will generate Nwok, and Samurai will generate Smri.

Include a class UserMainCode with a static method getAlternateChars which accepts the string. The return type is the modified string.

Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Output consists of string.

Refer sample output for formatting specifications.

Sample Input 1:

Hello

Sample Output 1:

Hlo

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.StringTokenizer;

public class UserMainCode{

public static String getAlternateChars(String s)

{

StringBuffer sbf = new StringBuffer();

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

{

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

i=i+1;

}

String str = sbf.toString();

return str;

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

163.String Processing - Username (fetchUserName)

Write a program to read a valid email id and extract the username.

Note - user name is the string appearing before @ symbol.

Include a class UserMainCode with a static method fetchUserName which accepts the string. The return type is the modified string.

Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Output consists of string.

Refer sample output for formatting specifications.

Sample Input 1:

admin@xyz.com

Sample Output 1:

admin

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.StringTokenizer;

public class UserMainCode{

public static String fetchUserName(String s1) {

StringTokenizer st=new StringTokenizer(s1,"@");

String s2=st.nextToken();

return s2;

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

164.String Processing - V (concatString)

Write a program to read a string array, concatenate the array elements one by one separated by comma and return the final string as output.

Include a class UserMainCode with a static method concatString which accepts the string array. The return type is the string.

Create a Class Main which would be used to accept the string array and call the static method present in UserMainCode.

Input and Output Format:

Input consists of an integer n which is the number of elements followed by n string values.

Output consists of the string.

Refer sample output for formatting specifications.

Sample Input 1:

3

AAA

BBB

CCC

Sample Output 1:

AAA,BBB,CCC

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public class UserMainCode{

public static String concatString(String s1[]){

StringBuffer sb=new StringBuffer();

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

{

sb.append(s1[i]).append(",");

}

sb.deleteCharAt(sb.length()-1);

return sb.toString();

} }

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

165.String Processing - V (returnLastRepeatedCharacters)

public class UserMainCode{

public static String formattingOfString(String s1, int n1)

{

StringBuffer sb = new StringBuffer();

for(int i = 0 ; i < n1 ; i++)

sb.append(s1.substring(s1.length()-n1, s1.length()));

return sb.toString();

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

166.String Processing - VII (isEqual)

Write a program to read a two strings and one int value(N). check if Nth character of first String from start and Nth character of second String from end are same or not. If both are same return true else return false.

Check need not be Case sensitive

Include a class UserMainCode with a static method isEqual which accepts the two strings and a integer n. The return type is the TRUE / FALSE.

Create a Class Main which would be used to read the strings and integer and call the static method present in UserMainCode.

Input and Output Format:

Input consists of two strings and an integer.

Output consists of TRUE / FALSE .

Refer sample output for formatting specifications.

Sample Input 1:

AAAA

abab

2

Sample Output 1:

TRUE

Sample Input 2:

MNOP

QRST

3

Sample Output 2:

FALSE

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public class UserMainCode{

public static boolean isEqual(String s1,String s2,int n){

boolean a=false;

char c=s1.charAt(n);

char d=s2.charAt(s2.length()-n);

String s3=Character.toString(c);

//System.out.println(s3);

String s4=Character.toString(d);

//System.out.println(s4);

if(s3.equalsIgnoreCase(s4))

{

a=true;

}else

{

a=false;

}

return a;

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

167.String Processing - ZigZag

Write a program to read a string containing date in DD-MM-YYYY format. find the number of days in the given month.

Note - In leap year February has got 29 days.

Include a class UserMainCode with a static method getLastDayOfMonth which accepts the string. The return type is the integer having number of days.

Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Output consists of integer.

Refer sample output for formatting specifications.

Sample Input 1:

12-06-2012

Sample Output 1:

30

Sample Input 2:

10-02-2012

Sample Output 2:

29

import java.io.IOException;

import com.sun.org.apache.xerces.internal.impl.xpath.regex.ParseException;

public class Main {

static String s1="10-02-2012";

public static void main(String[] args) throws IOException, ParseException, java.text.ParseException {

System.out.println(UserMainCode.getLastDayOfMonth(s1));

}

}

import java.util.Date;

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Calendar;

public class UserMainCode{

public static int getLastDayOfMonth(String s1) throws ParseException{

SimpleDateFormat sdf=new SimpleDateFormat("dd-MM-yyyy");

Calendar cal=Calendar.getInstance();

Date d1=sdf.parse(s1);

cal.setTime(d1);

int n=cal.getActualMaximum(Calendar.DAY\_OF\_MONTH);

return n;

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

168.String Processing (exchangeCharacters)

class UserMainCode{

public static String exchange(string s1){

StringBuffer sb=new StringBuffer();

char c=s1.charAt(0);

String s2=s1.substring(1, s1.length()-1);

char c1=s1.charAt(s1.length()-1);

sb.append(c1).append(s2).append(c);

return sb.toString();

}

}

????????????????????????????????????????????????????????????????

public class Main {

/\*\*

\* @param args

\*/

public static void main(String[] args) {

String s1="HelloWorld";

StringBuffer sb=new StringBuffer();

char c=s1.charAt(0);

String s2=s1.substring(1, s1.length()-1);

char c1=s1.charAt(s1.length()-1);

sb.append(c1).append(s2).append(c);

System.out.println(sb);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

169.String Repetition (repeatString)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* COGNIZANT/4/COG COG

import java.util.\*;

public class useer {

public static String lengthiestString(String s1,int n){

StringBuffer sb=new StringBuffer();

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

{if(n>=3)

sb.append(s1.substring(0,3)).append(" ");

else if(n==2)

sb.append(s1.substring(0,2)).append(" ");

else if(n==1)

sb.append(s1.substring(0,1));

}

return sb.toString();

}

public static void main(String[] args) {

Scanner s=new Scanner(System.in);

System.out.println("enter the String:");

String s1=s.nextLine();

int n=s.nextInt();

System.out.println("the lengthiest string is:"+lengthiestString(s1,n));

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

170.String Splitter

Write a program which would accept a string and a character as a delimiter. Apply the below rules

1. Using the delimiter, split the string and store these elements in array.

2. Reverse each element of the string and convert it into lowercase.

Include a class UserMainCode with a static method manipulateLiteral which accepts the string and character. The return type is the string array formed.

Create a Class Main which would be used to accept the string and characterand call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string and character.

Output consists of a string array.

Refer sample output for formatting specifications.

Sample Input 1:

AAA/bba/ccc/DDD

/

Sample Output 1:

aaa

abb

ccc

ddd

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public class Main {

public static void main(String[] args)

{

String ip1="AAA/bba/ccc/DDD";

char ip2='/';

String op[]=UserMainCode.manipulateLiteral(ip1,ip2);

for(String s:op)

System.out.println(s);

}

}

import java.util.ArrayList;

import java.util.StringTokenizer;

class UserMainCode{

public static String[] manipulateLiteral(String ip1, char ip2)

{

StringTokenizer t1 = new StringTokenizer(ip1,"/");

ArrayList<String> lst = new ArrayList<String>();

while(t1.hasMoreTokens())

{

StringBuffer sb = new StringBuffer();

sb.append(t1.nextToken().toLowerCase());

lst.add(sb.reverse().toString());

}

String[] op = new String[lst.size()];

for(int i = 0;i<lst.size();i++)

{

op[i] = (String) lst.get(i);

}

return op;

}

}

171.Strings – Unique & Existing Characters (replaceplus)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.Scanner;

public class Main {

public static void main(String []args){

Scanner sc=new Scanner(System.in);

String n=sc.nextLine();

String n1=sc.nextLine();

System.out.println(UserMainCode.display(n,n1));

}

}

public class UserMainCode

{

public static String display(String s,String s1)

{

String s2=s.toLowerCase();

String s3=s1.toLowerCase();

StringBuffer sb=new StringBuffer();

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

{

char c=s2.charAt(i);

if(s3.indexOf(c)==-1)

sb.append("+");

else

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

} return sb.toString();

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

172.Strings Processing - Replication (repeatString)

Write a program to read a string and also a number N. Return the replica of original string for n given time.

Include a class UserMainCode with a static method repeatString which accepts the the string and the number n. The return type is the string based on the problem statement.

Create a Class Main which would be used to accept the string and integer and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string and integer.

Output consists of a string.

Refer sample output for formatting specifications.

Sample Input 1:

Lily

2

Sample Output 1:

LilyLily

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Lily/2/LilyLily

import java.util.\*;

public class UserMainCode {

public static String lengthiestString(String s1,int n){

StringBuffer sb=new StringBuffer();

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

{

sb.append(s1);

}

return sb.toString();

}

import java.util.Scanner;

public class Main {

public static void main(String [] args )

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

String s1=sc.nextLine();

int n=sc.nextInt();

System.out.println(Usercode.lengthiestString(s1, n));

}

}

173.Strings Processing(findFruitName)

Write a program to read a string that contains comma separated fruit names and also a number N. Pick the nth fruit and return it. If the total number of elements are less than the number specified in N, then return the last element.

Include a class UserMainCode with a static method findFruitName which accepts the the string and the number n. The return type is the string which has the fruit name.

Create a Class Main which would be used to accept the string and integer and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string and integer.

Output consists of a string.

Refer sample output for formatting specifications.

Sample Input 1:

Apple,Banana,Orange

2

Sample Output 1:

Banana

Sample Input 2:

Apple,Banana,Orange

4

Sample Output 2:

Orange

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.Scanner;

public class Main

{

public static void main(String args[])

{

String str=new String();

Scanner sc=new Scanner(System.in);

str=sc.nextLine();

int n=sc.nextInt();

String k=UserMainCode.findFruitName(str, n);

System.out.println(k);

}

import java.util.StringTokenizer;

public class UserMainCode

{

public static String findFruitName(String m,int n)

{

int i=0;

String h=null;

StringTokenizer st=new StringTokenizer(m,",");

int max=st.countTokens();

String[] ss=new String[max];

while(st.hasMoreElements())

{

ss[i++]=st.nextToken();

}

if(n>max)

h=ss[i-1];

else

h=ss[n-1];

return h;

}

}

174.Sum Non Prime Numbers

Write a program to calculate the sum of all the non prime positive numbers less than or equal to the given number.

Note: prime is a natural number greater than 1 that has no positive divisors other than 1 and itself

Example:

input = 9

Prime numbers = 2,3,5 and 7

output = 1+4+6+8+9=28

Include a class UserMainCode with a static method ?addNumbers? that accepts an integer arguement and returns an integer.

Create a class Main which would get an integer as input and call the static method validateNumber present in the UserMainCode.

Input and Output Format:

Input consists of an integer.

Output consists of an integer.

Sample Input:

9

Sample Output:

28

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.text.DecimalFormat;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.Iterator;

import java.util.Map;

import java.util.StringTokenizer;

public class UserMainCode {

public static int func(int n) {

int sum=0;int k=0;int sum1=0;

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

{ k=0;

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

{

if(i%j==0)

k++;

}

if(k==0)

{

sum=sum+i;

}

}

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

sum1=sum1+r;

return sum1-sum;

}

}

import java.util.Scanner;

public class Main {

public static void main(String [] args )

{

Scanner sc=new Scanner(System.in);

int n=sc.nextInt();

int k=Usercode.func(n);

System.out.println(k);

}

}

175.Sum of Common Elements (getSumOfIntersection)

Write a program to read two integer arrays and find the sum of common elements in both the arrays. If there are no common elements return -1 as output

Include a class UserMainCode with a static method sumCommonElements which accepts two single integer array. The return type (integer) should be the sum of common elements.

Create a Class Main which would be used to accept Input array and call the static method present in UserMainCode.

Assume that all the elements will be distinct.

Input and Output Format:

Input consists of 2n+1 integers. The first integer corresponds to n, the number of elements in the array. The next 'n' integers correspond to the elements in the array, The last n elements correspond to the elements of the second array.

Output consists of a single Integer value.

Refer sample output for formatting specifications.

Assume that the maximum number of elements in the array is 20.

Sample Input 1:

4

1

2

3

4

2

3

6

7

Sample Output 1:

5

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.Scanner;

public class Main {

public static void main(String[] args)

{

Scanner sc=new Scanner(System.in);

int n=sc.nextInt();

int m=sc.nextInt();

int[] a=new int[n];

int[] b=new int[m];

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

a[i]=sc.nextInt();

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

b[i]=sc.nextInt();

int u=UserMainCode.display(a,b);

if(u==-1)

System.out.println("No common elements");

else

System.out.println(u);}}

public class UserMainCode {

public static int display(int a[],int b[])

{

int sum=0;

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

{

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

{if(a[i]==b[j])

sum=sum+a[i];

}}

if(sum==0)

return -1;

else

return sum;

}}

176.Sum of cubes and squares of elements in an array (addEvenOdd)

Write a program to get an int array as input and identify even and odd numbers. If number is odd get cube of it, if number is even get square of it. Finally add all cubes and squares together and return it as output.

Include a class UserMainCode with a static method addEvenOdd which accepts integer array as input.

The return type of the output is an integer which is the sum of cubes and squares of elements in the array.

Create a class Main which would get the input and call the static method addEvenOdd present in the UserMainCode.

Input and Output Format:

Input consists of integer array.

Output is an integer sum.

Refer sample output for formatting specifications.

Sample Input 1:

5

2

6

3

4

5

Sample Output 1:

208

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 5/2/6/3/4/5/208

public class Main {

public static void main(String[] args) {

int a[]={2,4,3,5,6};

System.out.println(summationPattern(a));

}

public static int summationPattern(int[] a) {

int n1=0,n2=0;

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

if(a[i]%2==0)

n1+=(a[i]\*a[i]);

else

n2+=(a[i]\*a[i]\*a[i]);

return n1+n2;

}

}

177.Sum of Digits

Write a Program that accepts a word as a parameter, extracts the digits wi0thin the string and returns its sum.

Include a class UserMainCode with a static method getdigits which accepts a string. The return type is integer representing the sum.

Create a Class Main which would be used to accept the input string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string with maximum size of 100 characters.

Output consists of a single string.

Refer sample output for formatting specifications.

Sample Input 1:

abc12de4

Sample Output 1:

7

Close

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.Scanner;

public class Main {

/\*\*

\* @param args

\*/

public static void main(String[] args) {

Scanner sc=new Scanner(System.in);

String s1=sc.nextLine();

int z=Usercode.getvalues(s1);

System.out.println(z);}

}

public class Usercode

{

public static int getvalues(String s1) {

int sum=0;

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

{

char a=s1.charAt(i);

if(Character.isDigit(a))

{

int b=Integer.parseInt(String.valueOf(a));

sum=sum+b;

}

}

if(sum==0)

{

//System.out.println(-1);

return -1;

}

else

//System.out.println(sum);

return sum;

}

}

===========================================================================

178.Sum of Digits in a String (sumOfDigits)

Write code to get the sum of all the digits present in the given string.

Include a class UserMainCode with a static method sumOfDigits which accepts string input.

Return the sum as output. If there is no digit in the given string return -1 as output.

Create a class Main which would get the input and call the static method sumOfDigits present in the UserMainCode.

Input and Output Format:

Input consists of a string.

Output is a single integer which is the sum of digits in a given string.

Refer sample output for formatting specifications.

Sample Input 1:

good23bad4

Sample Output 1:

9

Sample Input 2:

good

Sample Output 2:

-1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* good23bad4

import java.util.Scanner;

public class Main {

/\*\*

\* @param args

\*/

public static void main(String[] args) {

Scanner sc=new Scanner(System.in);

String s1=sc.nextLine();

int z=Usercode.getvalues(s1);

System.out.println(z);}

}

public class Usercode

{

public static int getvalues(String s1) {

int sum=0;

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

{

char a=s1.charAt(i);

if(Character.isDigit(a))

{

int b=Integer.parseInt(String.valueOf(a));

sum=sum+b;

}

}

if(sum==0)

{

//System.out.println(-1);

return -1;

}

else

//System.out.println(sum);

return sum; }

}

179.Sum of Lowest marks (getLowest)

Given input as HashMap, value consists of marks and rollno as key.Find the sum of the lowest three subject marks from the HashMap.

Include a class UserMainCode with a static method getLowest which accepts a Hashmap with marks and rollno.

The return type of the output is the sum of lowest three subject marks.

Create a class Main which would get the input and call the static method getLowest present in the UserMainCode.

Input and Output Format:

First line of the input corresponds to the HashMap size.

Input consists a HashMap with marks and rollno.

Output is an integer which is the sum of lowest three subject marks.

Refer sample output for formatting specifications.

Sample Input 1:

5

1

54

2

85

3

74

4

59

5

57

Sample Output 1:

170

Sample Input 2:

4

10

56

20

58

30

87

40

54

Sample Output 2:

168

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 5/1/54/2/85/3/74/4/59/5/57/170

import java.util.ArrayList;

import java.util.HashMap;

import java.util.Scanner;

public class Main {

public static void main(String args[]){

Scanner sc = new Scanner(System.in);

int n=Integer.parseInt(sc.nextLine());

HashMap<Integer,Integer>h1=new HashMap<Integer,Integer>();

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

{

h1.put(sc.nextInt(),sc.nextInt());

}

System.out.println(UserMainCode.getvalues(h1));

}

}

import java.util.ArrayList;

import java.util.Collections;

import java.util.HashMap;

import java.util.Iterator;

public class UserMainCode {

public static int getvalues(HashMap<Integer,Integer>h1)

{

ArrayList<Integer>a1=new ArrayList<Integer>();

int m=0;

Iterator<Integer> it=h1.keySet().iterator();

while(it.hasNext())

{

int x=it.next();

a1.add(h1.get(x));

}

Collections.sort(a1);

m=a1.get(0)+a1.get(1)+a1.get(2);

return m;

}}

180.Sum of Max & Min (getSumMaxMin)

Write a Program that accepts three integers, and returns the sum of maximum and minimum numbers.

Include a class UserMainCode with a static method getSumMaxMin which accepts three integers. The return type is integer representing the formula.

Create a Class Main which would be used to accept the input integers and call the static method present in UserMainCode.

Input and Output Format:

Input consists of three integers.

Output consists of a single integer.

Refer sample output for formatting specifications.

Sample Input 1:

12

17

19

Sample Output 1:

31

Close

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 12/17/19/31

public class Usercode

{

public static int display(int a,int b,int c)

{

int d=0;

if(a<b&&b<c)

{

d=a+c;

}

else if(a<b&&b>c)

{

d=b+c;

}

else if(a>b&&b<c)

{

d=a+b;

}

return d;

}}

import java.util.Scanner;

public class Main {

public static void main(String args[]){

Scanner sc = new Scanner(System.in);

int n1=Integer.parseInt(sc.nextLine());

int n2=Integer.parseInt(sc.nextLine());

int n3=Integer.parseInt(sc.nextLine());

System.out.println(Usercode.display(n1, n2, n3));

}

}

181.Sum of Powers of elements in an array (getSumOfPower)

Given a method with an int array. Write code to find the power of each individual element accoding to its position index, add them up and return as output.

Include a class UserMainCode with a static method getSumOfPower which accepts an integer array as input.

The return type of the output is an integer which is the sum powers of each element in the array.

Create a Main class which gets integer array as an input and call the static method getSumOfPower present in the UserMainCode.

Input and Output Format:

Input is an integer array.First element corresponds to the number(n) of elements in an array.The next inputs corresponds to each element in an array.

Output is an integer .

Sample Input 1:

4

3

6

2

1

Sample Output 1:

12

Sample Input 2:

4

5

3

7

2

Sample Output 2:

61

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 4/3/6/2/1

public class Usercode{

public static int display(int n,int[]a)

{

{

int sum=0;

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

sum=(int)(sum+Math.pow(a[i], i));

return sum;

}}}

import java.util.Scanner;

public class Main {

public static void main(String args[]){

Scanner sc = new Scanner(System.in);

int s=sc.nextInt();

int a[]=new int[s];

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

{

a[i]=sc.nextInt();

}

System.out.println(Usercode.display(s, a));

}

}

182.Sum of Squares of Even Digits (sumOfSquaresOfEvenDigits)

Write a program to read a number , calculate the sum of squares of even digits (values) present in the given number.

Include a class UserMainCode with a static method sumOfSquaresOfEvenDigits which accepts a positive integer . The return type (integer) should be the sum of squares of the even digits.

Create a class Main which would get the input as a positive integer and call the static method sumOfSquaresOfEvenDigits present in the UserMainCode.

Input and Output Format:

Input consists of a positive integer n.

Output is a single integer .

Refer sample output for formatting specifications.

Sample Input 1:

56895

Sample Output 1:

100

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 56895/100

public class Usercode

{

public static int display(int number){

int n1=0,n2=0;

while(number!=0)

{

n1=number%10;

if((n1%2)==0)

n2+=n1\*n1;

number/=10;

}

return n2;

}

}

import java.util.Scanner;

public class Main {

public static void main(String args[]){

Scanner sc = new Scanner(System.in);

int s=sc.nextInt();

System.out.println(Usercode.display(s));

}

}

183.Sum Squares of Digits

Write a program that accepts a positive number as input and calculates the sum of squares of individual digits of the given number.

Include a class UserMainCode with a static method ?getSumOfSquaresOfDigits? that accepts an integer argument and returns an integer.

Create a class Main which would get an integer as input and call the static method getSumOfSquaresOfDigits present in the UserMainCode.

Input and Output Format:

Input consists of an integer.

Output consists of an integer.

Sample Input:

321

Sample Output:

14

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc=new Scanner(System.in);

int n=sc.nextInt();

System.out.println(Usercode.getvalues(n));

} }

public class Usercode{

public static int getvalues(int n) {

int a=n;

int rem=0;

int sum=0;

while(a!=0)

{

rem=a%10;

sum=sum+(rem\*rem);

a=a/10;

}

return sum;

}

}

184.SumOdd (addOddNumbers)

Write a program to read an integer and find the sum of all odd numbers from 1 to the given number. [inclusive of the given number]

if N = 9 [ 1,3,5,7,9]. Sum = 25

Include a class UserMainCode with a static method addOddNumbers which accepts the number n. The return type is the integer based on the problem statement.

Create a Class Main which would be used to accept the integer and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a integer.

Output consists of a integer.

Refer sample output for formatting specifications.

Sample Input 1:

6

Sample Output 1:

9

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 6/9

public class Usercode {

public static int sumOf(int n){

int a=0;

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

{

if(i%2!=0)

{

a=a+i;

}

}

return a;

}

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc=new Scanner(System.in);

int n=sc.nextInt();

System.out.println(Usercode.sumOf(n));

} }

185.Swap Characters (swapCharacter)

Write a program to input a String and swap the every 2 characters in the string. If size is an odd number then keep the last letter as it is. Print the final swapped string.

Include a class UserMainCode with a static method swapCharacter which accepts a string. The return type (String) should return the character swapped string.

Create a Class Main which would be used to accept Input String and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Output consists of a string.

Refer sample output for formatting specifications.

Sample Input 1:

TRAINER

Sample Output 1:

RTIAENR

Sample Input 2:

TOM ANDJERRY

Sample output 2:

OT MNAJDREYR

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc=new Scanner(System.in);

String s1=sc.nextLine();

System.out.println(Usercode.getvalues(s1));

} }

public class Usercode

{

public static String getvalues(String s1){

StringBuffer sb=new StringBuffer();

int l=s1.length();

if(l%2==0)

{

for(int i=0;i<s1.length()-1;i=i+2)

{

char a=s1.charAt(i);

char b=s1.charAt(i+1);

sb.append(b).append(a);

}

return sb.toString();

}

else

{

for(int i = 0;i<s1.length()-1;i=i+2)

{

char a=s1.charAt(i);

char b=s1.charAt(i+1);

sb.append(b).append(a);

}

sb.append(s1.charAt(l-1));

return sb.toString();

}

}

}

186.Symmetric Difference (getSymmetricDifference)

Write a program to read two integer array and calculate the symmetric difference of the two arrays. Finally Sort the array.

Symmetric difference is the difference of A Union B and A Intersection B ie. [ (A U B) - (A ^ B)]

Union operation merges the two arrays and makes sure that common elements appear only once. Intersection operation includes common elements from both the arrays.

Ex - A={12,24,7,36,14} and B={11,26,7,14}.

A U B ={ 7,11,12,14,24,26,36} and

A ^ B = {7,14}

Symmetric difference of A and B after sorting= [A U B] - [ A ^ B] = {11,12,24,26,36}.

Include a class UserMainCode with a static method getSymmetricDifference which accepts the integer array. The return type is an integer array.

Create a Class Main which would be used to accept the two integer arrays and call the static method present in UserMainCode.

Input and Output Format:

Input consists of an integer n which is the number of elements followed by n integer values. The same sequnce is followed for the next array.

Output consists of sorted symmetric difference array.

Refer sample output for formatting specifications.

Sample Input 1:

5

11

5

14

26

3

3

5

3

1

Sample Output 1:

1

11

14

26

import java.util.\*;

public class Main

{

public static void main(String[] args)

{

int n,m;

Scanner sin = new Scanner(System.in);

n = sin.nextInt();

int[] a1 = new int[n];

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

{

a1[i] = sin.nextInt();

}

m = sin.nextInt();

int[] a2 = new int[m];

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

{

a2[i] = sin.nextInt();

}

int[] result = UserMainCode.getSymmetricDifference (a1,a2);

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

System.out.println(result[i]);

}

}

import java.util.\*;

public class Usercode

{

public static int[] getSymmetricDifference (int[] a1,int[] a2)

{

//int[] a1 = new int[]{11,5,14,26,3};

//int[] a2 = new int[]{5,3,1};

int[] union,inter,result;

int count=0;

int max = a1.length+a2.length;

ArrayList<Integer> temp = new ArrayList<Integer>(max);

/\*union\*/

for(int i=0;i<a1.length;i++)

{

if(!temp.contains(a1[i]))

{

++count;

temp.add(a1[i]);

}

}

for(int i=0;i<a2.length;i++)

{

if(!temp.contains(a2[i]))

{

++count;

temp.add(a2[i]);

}

}

union = new int[count];

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

{

union[i] = (int)temp.get(i);

}

Arrays.sort(union);

/\*intersection\*/

temp = new ArrayList<Integer>(max);

count =0;

Arrays.sort(a2);

for(int i=0;i<a1.length;i++)

{

if(Arrays.binarySearch(a2,a1[i]) >= 0)

{

++count;

temp.add(a1[i]);

}

}

inter = new int[count];

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

{

inter[i] = (int)temp.get(i);

}

Arrays.sort(inter);

/\*difference \*/

temp = new ArrayList<Integer>(max);

count =0;

Arrays.sort(inter);

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

{

if(Arrays.binarySearch(inter,union[i]) < 0)

{

++count;

temp.add(union[i]);

}

}

result = new int[count];

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

{

result[i] = (int)temp.get(i);

}

Arrays.sort(result);

//System.out.println("resultant array : \n "+Arrays.toString(result));

return result;

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

187.Test Vowels

Write a program to read a string and check if given string contains exactly five vowels in any order. Print ?Yes? if the condition satisfies, else print ?No?.

Assume there is no repetition of any vowel in the given string and all characters are lowercase.

Include a class UserMainCode with a static method testVowels which accepts a string. The return type (Integer) should return 1 if all vowels are present, else return 2.

Create a Class Main which would be used to accept a string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Output consists of a string (?Yes? or ?No?).

Refer sample output for formatting specifications.

Sample Input 1:

acbisouzze

Sample Output 1:

Yes

Sample Input 2:

cbisouzze

Sample Output 2:

No

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc=new Scanner(System.in);

String s1=sc.nextLine();

System.out.println(Usercode.vowelsCheck(s1));}}

public static boolean vowelsCheck(String s1) {

boolean b=false;

int n1=0,n2=0,n3=0,n4=0,n5=0;

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

char c=s1.charAt(i);

if(c=='a')

n1++;

if(c=='e')

n2++;

if(c=='i')

n3++;

if(c=='o')

n4++;

if(c=='u')

n5++;}

if(n1==1 && n2==1 && n3==1 && n4==1 && n5==1)

b=true;

return b;

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

188.Three Digits (validatestrings)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* CTS-215

public class Main {

public static void main(String[] args) {

String s1="CTS-2j4";

getvalues(s1);

}

public static void getvalues(String s1) {

if(s1.matches("(CTS)[-]{1}[0-9]{3}"))

{

System.out.println(1);

}

else

System.out.println(-1);

}

}

189.Transfer from Hashmap to Arraylist(getName)

Write a program that constructs a hashmap with ?employee id? as key and ?name? as its value. Based on the rules below, on being satisfied, the name must be added to the arraylist.

i)First character should be small and the last character should be Capital.

ii)In name at least one digit should be there.

Include a class UserMainCode with a static method getName which accepts a hashmap. The return type is an arraylist as expected in the above statement

Create a Class Main which would be used to accept Input string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of 2n+1 values. The first value corresponds to size of the hashmap. The next n pair of numbers contains the employee id and name.

Output consists of arraylist of strings as mentioned in the problem statement.

Refer sample output for formatting specifications.

Sample Input 1:

4

1

ravi5raJ

2

sita8gitA

3

ram8sitA

4

rahul

Sample Output 1:

ravi5raJ

sita8gitA

ram8sitA

import java.util.ArrayList;

import java.util.HashMap;

import java.util.Iterator;

import java.util.LinkedHashMap;

import java.util.Scanner;

public class Main {

public static void main(String[] args)

{

LinkedHashMap<Integer,String> hm1=new LinkedHashMap<Integer,String>();

int n;

Scanner sc=new Scanner(System.in);

n=Integer.parseInt(sc.nextLine());

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

{

hm1.put(Integer.parseInt(sc.nextLine()),sc.nextLine());

}

ArrayList<String> al1=new ArrayList<String>();

al1=UserMain.getName(hm1);

Iterator<String> it=al1.iterator();

while(it.hasNext())

{

System.out.println(it.next());

}

}

}

import java.util.ArrayList;

import java.util.HashMap;

import java.util.Iterator;

import java.util.LinkedHashMap;

public class UserMain {

public static ArrayList<String> getName(HashMap<Integer,String> hm1)

{

ArrayList<String> al2=new ArrayList<String>();

LinkedHashMap<Integer,String> hm2=new LinkedHashMap<Integer,String>(hm1);

Iterator<Integer> it =hm2.keySet().iterator();

int flag1=0,flag2=0;

while(it.hasNext())

{

int id=it.next();

String name=hm2.get(id);

char[] check=name.toCharArray();

if(check[0]>='a'&&check[0]<='z')

{

if(check[name.length()-1]>='A'&&check[name.length()-1]<='Z')

{

flag1=1;

}

}

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

{

if(check[i]>='0'&&check[i]<='9')

{

flag2=1;

}

}

if(flag1==1&&flag2==1)

{

al2.add(name);

}

flag1=0;

flag2=0;

}

return al2;

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

190.7) Triplets(checkTripplets)

Given an integer array, Write a program to find if the array has any triplets. A triplet is a value if it appears 3 consecutive times in the array.

Include a class UserMainCode with a static method checkTripplets which accepts an integer array. The return type is boolean stating whether its a triplet or not.

Create a Class Main which would be used to accept the input arrayand call the static method present in UserMainCode.

Input and Output Format:

Input consists of n+1 integers. The first integer would represent the size of array and the next n integers would have the values.

Output consists of a string stating TRUE or FALSE.

Refer sample output for formatting specifications.

Sample Input 1:

7

3

3

5

5

5

2

3

Sample Output 1:

TRUE

Sample Input 2:

7

5

3

5

1

5

2

3

Sample Output 2:

FALSE

import java.util.Scanner;

public class Main {

public static void main(String[] args)

{

int n;

int[] a=new int[10];

Scanner sc=new Scanner(System.in);

n=sc.nextInt();

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

{

a[i]=sc.nextInt();

}

boolean s=Usercode.checkTripplets(a);

if(s)

System.out.println("TRUE");

else

System.out.println("FALSE");

}

}

public class Usercode {

public static boolean checkTripplets(int[] a)

{

boolean b=false;

for(int i=0;i<a.length-3;i++)

{

if((a[i]==a[i+1])&&(a[i+1]==a[i+2]))

{

b=true;

}

}

return b;

}

}

191.Unique Characters (checkUnique)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.ArrayList;

import java.util.List;

public class kape

{

public static void main(String[] args)

{

String s1="HOWAREYOU";

kape1.getvalues(s1);

}}

public class kape1

{

public static void getvalues(String s1) {

String s2=s1.toLowerCase();

StringBuffer sb=new StringBuffer(s2);

int l=sb.length();

int count=0;

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

{ count=0;

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

{

if(sb.charAt(i)==sb.charAt(j))

{

sb.deleteCharAt(j);

count++;

j--;

l--;

}

}

if(count>0)

{

sb.deleteCharAt(i);

i--;

l--;

}

}

if(sb.length()==0)

{

System.out.println(-1);

}

else

System.out.println(sb.length());

}

}

192.Unique Characters in a string (uniqueCounter)

Write a program that takes a string and returns the number of unique characters in the string. If the given string doest not contain any unique characters return -1

Include a class UserMainCode with a static method uniqueCounter which accepts a string as input.

The return type of the output is the count of all unique characters in the strings.

Create a class Main which would get the input and call the static method uniqueCounter present in the UserMainCode.

Input and Output Format:

Input consists a string.

Output is an integer.

Refer sample output for formatting specifications.

Sample Input 1:

HelloWorld

Sample Output 1:

5

Sample Input 2:

coco

Sample Output 2:

-1

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* HelloWorld/5

import java.util.Scanner;

public class Main {

/\*\*

\* @param args

\*/

public static void main(String[] args) { Scanner sc= new Scanner(System.in);

String s1=sc.nextLine();

System.out.println(Usercode.getvalues(s1));

}

}

public class Usercode{

public static int getvalues(String s1) {

String s2=s1.toLowerCase();

StringBuffer sb=new StringBuffer(s2);

int l=sb.length();

int count=0;

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

{ count=0;

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

{

if(sb.charAt(i)==sb.charAt(j))

{

sb.deleteCharAt(j);

count++;

j--;

l--;

j=i;

}

}

if(count>0)

{

sb.deleteCharAt(i);

i--;

l--;

}

}

if(sb.length()==0)

{

return -1;

}

else

{int abc= sb.length();

return abc; }}

}

193.Unique Even Sum (addUniqueEven)

Write a program to read an array, eliminiate duplicate elements and calculate the sum of even numbers (values) present in the array.

Include a class UserMainCode with a static method addUniqueEven which accepts a single integer array. The return type (integer) should be the sum of the even numbers. In case there is no even number it should return -1.

Create a Class Main which would be used to accept Input array and call the static method present in UserMainCode.

Input and Output Format:

Input consists of n+1 integers. The first integer corresponds to n, the number of elements in the array. The next 'n' integers correspond to the elements in the array.

In case there is no even integer in the input array, print no even numbers as output. Else print the sum.

Refer sample output for formatting specifications.

Assume that the maximum number of elements in the array is 20.

Sample Input 1:

4

2

5

1

4

Sample Output 1:

6

Sample Input 2:

3

1

1

1

Sample Output 2:

no even numbers

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.ArrayList;

import java.util.List;

import java.util.Scanner;

public class Main

{

public static void main(String[] args)

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

int n=sc.nextInt();

int a[]=new int[n];

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

{

a[i]= sc.nextInt();}

int res=Usercode.sumOfEvenNos(a);

if(res==-1)

System.out.println("No even Numbers");

else

System.out.println(res);

}}

import java.util.ArrayList;

public class Usercode

{

public static int sumOfEvenNos(int[] a)

{

int sum = 0,count=0;

ArrayList<Integer> lst = new ArrayList<Integer>();

lst.add(a[0]);

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

{

if(!lst.contains(a[i]))

{

lst.add(a[i]);

}

}

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

{

if(lst.get(j)%2==0)

{

sum+=lst.get(j);

}

else

count++;

}

if(count == lst.size())

return -1;

else

return sum;

}

}

194.Unique Number (getUnique)

Write a program that accepts an Integer as input and finds whether the number is Unique or not. Print Unique if the number is ?Unique?, else print ?Not Unique?.

Note: A Unique number is a positive integer (without leading zeros) with no duplicate digits.For example 7, 135, 214 are all unique numbers whereas 33, 3121, 300 are not.

Include a class UserMainCode with a static method getUnique which accepts an integer. The return type (Integer) should return 1 if the number is unique else return -1.

Create a Class Main which would be used to accept Input Integer and call the static method present in UserMainCode.

Input and Output Format:

Input consists of an integer .

Output consists of a String (?Unique? or ?Not Unique?).

Refer sample output for formatting specifications.

Sample Input 1:

123

Sample Output 1:

Unique

Sample Input 2:

33

Sample Output 2:

Not Unique

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 123/unique

import java.util.Scanner;

public class Main{

public static void main(String[]args)

{

Scanner sc=new Scanner(System.in);

int n=sc.nextInt();

int m=Usercode.cou(n);

if(m>0)

{

System.out.println("Invalid");

}

else

{

System.out.println("valid");

}

}}

public class Usercode{

public static int cou(int n)

{

int []a=new int[100];

int i=0,count=0;

while(n!=0)

{

int num=n%10;

a[i]=num;

i++;

n=n/10;

}

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

{

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

{

if(a[j]==a[k]){

count++;

}

}}

return count;}}

195.Unique Number (calculateUnique)

GIven three integers (a,b,c) , Write a program that returns the number of unique integers among the three.

Include a class UserMainCode with a static method calculateUnique which accepts three integers and returns the count as integer.

Create a Class Main which would be used to accept three integers and call the static method present in UserMainCode.

Input and Output Format:

Input consists of three integers.

Output consists of a integer.

Refer sample output for formatting specifications.

Sample Input 1:

12

4

3

Sample Output 1:

3

Sample Input 2:

4

-4

4

Sample Output 2:

2

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 12/4/3/3

import java.util.Scanner;

public class Main{

public static void main(String[]args)

{

Scanner sc=new Scanner(System.in);

int n1=sc.nextInt();

int n2=sc.nextInt();

int n3=sc.nextInt();

System.out.println(Usercode.display(n1, n2, n3));

}}

public class Usercode

{

public static int display(int a,int b,int c)

{

int d=0;

if(a!=b&&a!=c)

{

d=3;

}

else if(a==b&&a==c)

{

d=1;

}

else if(a!=b&&a==c)

{

d=2;

}

else if(a==b&&a!=c)

{

d=2;

}

return d;

}}

196.Valid Date

Obtain a date string in the format dd/mm/yyyy. Write code to validate the given date against the given format.

Include a class UserMainCode with a static method validateDate which accepts a string .

The return type of the validateDate method is 1 if the given date format matches the specified format , If the validation fails return the output as -1.

Create a Main class which gets date string as an input and call the static method validateDate present in the UserMainCode.

Input and Output Format:

Input is a string .

Refer sample output for formatting specifications

Sample Input 1:

12/06/1987

Sample Output 1:

Valid date format

Sample Input 2:

03/1/1987

Sample Output 2:

Invalid date format

import java.util.Date;

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc=new Scanner(System.in);

String s=sc.nextLine();

int res=Usercode.getvalues(s);

if(res==-1)

System.out.println("InValid date format");

else

System.out.println("Valid date format");

} }

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Date;

public class Usercode{

public static int getvalues(String s) {

int res=0;

if(s.matches("[0-9]{2}[.]{1}[0-9]{2}[.]{1}[0-9]{4}"))

{

SimpleDateFormat sdf=new SimpleDateFormat("dd.MM.yyyy");

sdf.setLenient(false);

try

{

Date d1=(Date) sdf.parse(s);

System.out.println(1);

} catch (ParseException e) {

res=-1;

}

}

else if(s.matches("[0-9]{2}[/]{1}[0-9]{2}[/][0-9]{4}"))

{

SimpleDateFormat sdf=new SimpleDateFormat("dd/MM/yyyy");

sdf.setLenient(false);

try

{

Date d1=(Date) sdf.parse(s);

res= 1;

} catch (ParseException e) {

res= -1;

}

}

else if(s.matches("[0-9]{2}[-]{1}[0-9]{2}[-][0-9]{4}"))

{

SimpleDateFormat sdf=new SimpleDateFormat("dd-MM-yyyy");

sdf.setLenient(false);

try

{

Date d1=(Date) sdf.parse(s);

res=1;

} catch (ParseException e) {

res=-1;

}

}

else

res= -1;

return res;

}}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

197.Validate Number (validateNumber) -94923

Given a negative number as string input, write a program to validate the number and to print the corresponding positive number.

Include a class UserMainCode with a static method ?validateNumber? that accepts a string argument and returns a string. If the argument string contains a valid negative number, the method returns the corresponding positive number as a string. Else the method returns -1.

Create a class Main which would get a String as input and call the static method validateNumber present in the UserMainCode.

Input and Output Format:

Input consists of a String.

Output consists of a String.

Sample Input 1:

-94923

Sample Output 1:

94923

Sample Input 2:

-6t

Sample Output 2:

-1

import java.util.Date;

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc=new Scanner(System.in);

String s=sc.nextLine();

System.out.println(Usercode.display(s));

} }

public class Usercode

{

public static int display(String s)

{

int count=0,n;

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

{

char c=s.charAt(i);

if(Character.isDigit(c))

count++;

}if(count==s.length()-1)

{n=Integer.parseInt(s);

n=(-1)\*n;

}

else

n=-1;

return n;

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

198.Validate Time (validateTime)

Obtain a time string as input in the following format 'hh:mm am' or 'hh:mm pm'. Write code to validate it using the following rules:

- It should be a valid time in 12 hrs format

- It should have case insensitive AM or PM

Include a class UserMainCode with a static method validateTime which accepts a string.

If the given time is as per the given rules then return 1 else return -1.If the value returned is 1 then print as valid time else print as Invalid time.

Create a Main class which gets time(string value) as an input and call the static method validateTime present in the UserMainCode.

Input and Output Format:

Input is a string .

Output is a string .

Sample Input 1:

09:59 pm

Sample Output 1:

Valid time

Sample Input 2:

10:70 AM

Sample Output 2:

Invalid time

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

import java.text.SimpleDateFormat;

import java.util.Date;

import java.util.Scanner;

import java.util.StringTokenizer;

public class Main{

public static void main(String []args){

Scanner sc=new Scanner(System.in);

String str=sc.nextLine();

int sum=Usercode.display(str);

if(sum==1)

{System.out.println("Valid ");}

else{

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

}}

import java.text.SimpleDateFormat;

import java.util.Date;

import java.util.StringTokenizer;

public class Usercode

{

public static int display(String s)

{int res=0;

StringTokenizer st=new StringTokenizer(s,":");

if(st.countTokens()==3)

{

SimpleDateFormat sdf1 = new SimpleDateFormat("h:mm:ss a");

sdf1.setLenient(false);

try

{

Date d2=sdf1.parse(s);

res=1;

}

catch(Exception e)

{

res=-1;

}

}

else

{

SimpleDateFormat sdf = new SimpleDateFormat("h:mm a");

sdf.setLenient(false);

try

{

Date d1=sdf.parse(s);

res=1;

}

catch(Exception e){

res=-1;

}

}

return res;

}}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

199.Validating Date Format (validateDate)

Obtain a date string in the format dd/mm/yyyy. Write code to validate the given date against the given format.

Include a class UserMainCode with a static method validateDate which accepts a string .

The return type of the validateDate method is 1 if the given date format matches the specified format , If the validation fails return the output as -1.

Create a Main class which gets date string as an input and call the static method validateDate present in the UserMainCode.

Input and Output Format:

Input is a string .

Refer sample output for formatting specifications

Sample Input 1:

12/06/1987

Sample Output 1:

Valid date format

Sample Input 2:

03/1/1987

Sample Output 2:

Invalid date format

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Date;

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc=new Scanner(System.in);

String s1=sc.nextLine();

int s=Usercode.getvalues(s1);

if(s==1)

System.out.println("Valid date format");

else

System.out.println("InValid date format");}

}

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Date;

import java.util.StringTokenizer;

public class Usercode

{

public static int getvalues(String s1) {

int res=0;

if(s1.matches("[0-9]{2}[/]{1}[0-9]{2}[/]{1}[0-9]{4}"))

{

SimpleDateFormat sdf=new SimpleDateFormat("dd/MM/yyyy");

sdf.setLenient(false);

try {

Date d1=sdf.parse(s1);

res= 1;

} catch (ParseException e) {

res= -1;

}

}

else

res= -1;

return res;

}}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 12/06/1987

200.Validating Input Password (validatePassword)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ashok\_23

import java.util.\*;

public class Main {

public static void main(String[] args) {

Scanner s=new Scanner(System.in);

String s1=s.next();

boolean b=passwordValidation(s1);

if(b==true)

System.out.println("valid password");

else

System.out.println("not a valid password");

}

public static boolean passwordValidation(String s1) {

boolean b=false,b1=false,b2=false;

if(s1.length()>=8)

if(!Character.isDigit(s1.charAt(0)))

if(s1.charAt(0)!='@' && s1.charAt(0)!='\_' && s1.charAt(0)!='#')

if(s1.charAt(s1.length()-1)!='@' && s1.charAt(s1.length()-1)!='\_' && s1.charAt(s1.length()-1)!='#')

b1=true;

if(b1==true)

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

if(Character.isAlphabetic(s1.charAt(i)) || Character.isDigit(s1.charAt(i)) || s1.charAt(i)=='#' || s1.charAt(i)=='@' || s1.charAt(i)=='\_')

b2=true;

if(b2==true)

if(s1.contains("#") || s1.contains("@") || s1.contains("\_"))

b=true;

return b;

}

}

201.Vowel Check (getVowels)

Write a program to read a String and check if that String contains all the vowels. Print ?yes? if the string contains all vowels else print ?no?.

Include a class UserMainCode with a static method getVowels which accepts a string. The return type (integer) should return 1 if the String contains all vowels else return -1.

Create a Class Main which would be used to accept Input String and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Output consists of a string(?yes? or ?no?).

Refer sample output for formatting specifications.

Sample Input 1:

abceiduosp

Sample Output 1:

yes

Sample Input 2:

bceiduosp

Sample Output 2:

no

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.\*;

public class Main {

public static void main(String[] args) {

String name="abceiduosp";

int n1=0,n2=0,n3=0,n4=0,n5=0;

System.out.println(UserMainCode.getVowels(name));

}

}

class UserMainCode {

public static int getVowels(String name){

String s1=name;

int n1=0,n2=0,n3=0,n4=0,n5=0;

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

char c=s1.charAt(i);

if(c=='a' || c=='A')

n1++;

if(c=='e' || c=='E')

n2++;

if(c=='i' || c=='I')

n3++;

if(c=='o' || c=='O')

n4++;

if(c=='u' || c=='U')

n5++;}

if(n1==1 && n2==1 && n3==1 && n4==1 && n5==1)

return 1;

else

return -1 ;

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

202.Vowel Count

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.Scanner;

class UserMainCode {

public static int countVowels(String name){

Scanner sc=new Scanner(System.in);

String s=sc.nextLine();

int max=0;

int count=0;

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

{

char c=s.charAt(i);

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

{

count++;

}

}

if(count>max)

{

max=count;

return 1;}

else

return -1;

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

203.Vowels (storeMaxVowelWord)

Given a String input, write a program to find the word which has the the maximum number of vowels. If two or more words have the maximum number of vowels, print the first word.

Include a class UserMainCode with a static method ?storeMaxVowelWord? that accepts a string argument and returns the word containing the maximum number of vowels.

Create a class Main which would get the a String as input and call the static method storeMaxVowelWord present in the UserMainCode.

Input and Output Format:

Input consists of a string. The string may contain both lower case and upper case letters.

Output consists of a string.

Sample Input :

What is your name?

Sample Output :

your

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.\*;

public class Main {

public static void main(String[] args) {

String s1 = "What is your name?";

String s2=UserMainCode.storeMaxVowelWord(s1);

System.out.println(s2);

}

}

import java.util.Scanner;

import java.util.StringTokenizer;

class UserMainCode {

public static String storeMaxVowelWord(String s1) {

int i = 0;

StringTokenizer st = new StringTokenizer(s1," ");

int len = 0;

int count = 0;

int count2 = 0;

String s6 = null;

while (st.hasMoreTokens()) {

String s5 = st.nextToken();

len = s5.length();

count=0;

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

if (s5.charAt(i) == 'a' || s5.charAt(i) == 'e'|| s5.charAt(i) == 'i' || s5.charAt(i) == 'o'|| s5.charAt(i) == 'u'

||s5.charAt(i) == 'A' ||s5.charAt(i) == 'E' ||s5.charAt(i) == 'I' ||s5.charAt(i) == 'O' ||s5.charAt(i) == 'U')

count++;

}

if (count > count2)

{

count2 = count;

s6 = s5;

}

}

return s6;

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

204.Vowels, Arrays & ArrayLists(matchCharacter)

Write a program to read an array of strings and return an arraylist which consists of words whose both first and last characters are vowels. Assume all inputs are in lowecase.

Include a class UserMainCode with a static method matchCharacter which accepts a string array. The return type shoud be an arraylist which should contain elements as mentioned above.

Create a Class Main which would be used to accept Input array and call the static method present in UserMainCode.

Input and Output Format:

Input consists of n+1 integers. The first integer corresponds to n, the number of elements in the array. The next 'n' string correspond to the elements in the array.

Output consists of strings which are elements of arraylist

Refer sample output for formatting specifications.

Sample Input 1:

4

abcde

pqrs

abci

orto

Sample Output 1:

abcde

abci

orto

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.ArrayList;

import java.util.Scanner;

import javax.swing.text.html.HTMLDocument.Iterator;

public class Main {

public static void main(String[] args)

{

int n;

Scanner sc=new Scanner(System.in);

n=Integer.parseInt(sc.nextLine());

String[] str=new String[n];

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

{

str[i]=sc.nextLine();

}

ArrayList<String> arr=new ArrayList<String>();

arr=UserMainCode.getName(str);

Iterator<String> it=arr.iterator();

while(it.hasNext())

{

System.out.println(it.next());

}

}

}

import java.util.ArrayList;

import java.util.Scanner;

import java.util.StringTokenizer;

class UserMainCode {

public static ArrayList<String> getName(String[] ss)

{

ArrayList<String> as=new ArrayList<String>();

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

{

String sp=ss[i];

char[] mp=sp.toLowerCase().toCharArray();

if((mp[0]=='a'||mp[0]=='e'||mp[0]=='i'||mp[0]=='o'||mp[0]=='u')&&(mp[sp.length()-1]=='a'||mp[sp.length()-1]=='e'||mp[sp.length()-1]=='i'||mp[sp.length()-1]=='o'||mp[sp.length()-1]=='u'))

{

as.add(sp);

}

}

return as;

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

204.Vowels, Arrays & ArrayLists(matchCharacter)

Write a program to read an array of strings and return an arraylist which consists of words whose both first and last characters are vowels. Assume all inputs are in lowecase.

Include a class UserMainCode with a static method matchCharacter which accepts a string array. The return type shoud be an arraylist which should contain elements as mentioned above.

Create a Class Main which would be used to accept Input array and call the static method present in UserMainCode.

Input and Output Format:

Input consists of n+1 integers. The first integer corresponds to n, the number of elements in the array. The next 'n' string correspond to the elements in the array.

Output consists of strings which are elements of arraylist

Refer sample output for formatting specifications.

Sample Input 1:

4

abcde

pqrs

abci

orto

Sample Output 1:

abcde

abci

orto

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.util.\*;

public class Main

{

public static void main(String[] args)

{

int n;

Scanner sc=new Scanner(System.in);

n=Integer.parseInt(sc.nextLine());

String[] str=new String[n];

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

{

str[i]=sc.nextLine();

}

ArrayList<String> arr=new ArrayList<String>();

arr=UserMainCode.getName(str);

Iterator<String> it=arr.iterator();

while(it.hasNext())

{

System.out.println(it.next());

}

}

}

import java.util.\*;

public class UserMainCode

{

public static ArrayList<String> getName(String[] ss)

{

ArrayList<String> as=new ArrayList<String>();

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

{

String sp=ss[i];

char[] mp=sp.toLowerCase().toCharArray();

if((mp[0]=='a'||mp[0]=='e'||mp[0]=='i'||mp[0]=='o'||mp[0]=='u')&&(mp[sp.length()-1]=='a'||mp[sp.length()-1]=='e'||mp[sp.length()-1]=='i'||mp[sp.length()-1]=='o'||mp[sp.length()-1]=='u'))

{

as.add(sp);

}

}

return as;

}

}

205.Word Count - II (countWord)

Write a program to read a string and count the number of words present in it.

Include a class UserMainCode with a static method countWord which accepts the string. The return type is the integer giving out the count of words.

Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Output consists of integer.

Refer sample output for formatting specifications.

Sample Input 1:

Today is Sunday

Sample Output 1:

3

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Today is Sunday/3

import java.util.Scanner;

import java.util.StringTokenizer;

public class Main {

public static void main(String[] args) {

Scanner sc=new Scanner(System.in);

String s1=sc.nextLine();

StringTokenizer st=new StringTokenizer(s1," ");

int n=st.countTokens();

System.out.println(n);

}

}

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Date;

import java.util.StringTokenizer;

public class Usercode

{

public static int getvalues(String s1) {

StringTokenizer st=new StringTokenizer(s1," ");

int n=st.countTokens();

return n;

}}

206.Word Count (countWord)

Given a string array (s) and non negative integer (n) and return the number of elements in the array which have same number of characters as the givent int N.

Include a class UserMainCode with a static method countWord which accepts the string array and integer. The return type is the string formed based on rules.

Create a Class Main which would be used to accept the string and integer and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a an integer indicating the number of elements in the string array followed the elements and ended by the non-negative integer (N).

Output consists of a integer .

Refer sample output for formatting specifications.

Sample Input 1:

4

a

bb

b

ccc

1

Sample Output 1:

2

Sample Input 2:

5

dog

cat

monkey

bear

fox

3

Sample Output 2:

3

public class UserMainCode

{public static int display(int n,String str[],int c)

{

int count=0;

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

{

if(str[i].length()==c)

{

count++;

}

}

return count;

}

}

??????????????????????????????????????????????????????????????????

import java.util.Scanner;

public class Main {

public static void main(String []args){

Scanner sc=new Scanner(System.in);

int a=sc.nextInt();

String []s=new String[a];

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

s[i]=sc.next();

}

int b=sc.nextInt();

System.out.println(UserMainCode.display(a,s,b));

}}

public class UserMainCode

{public static int display(int n,String str[],int c)

{

int count=0;

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

{

if(str[i].length()==c)

{

count++;

}

}

return count;

}

}

207.Word Count (sumOfDigits)

Given a string array (s) and non negative integer (n) and return the number of elements in the array which have same number of characters as the givent int N.

Include a class UserMainCode with a static method countWord which accepts the string array and integer. The return type is the string formed based on rules.

Create a Class Main which would be used to accept the string and integer and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a an integer indicating the number of elements in the string array followed the elements and ended by the non-negative integer (N).

Output consists of a integer.

Refer sample output for formatting specifications.

Sample Input 1:

4

a

bb

b

ccc

1

Sample Output 1:

2

Sample Input 2:

5

dog

cat

monkey

bear

fox

3

Sample Output 2:

3

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public class B

{

public static void main(String[] args)

{

String[] s1={"2AA","12","A2C","C5a"};

getSum(s1);

}

public static void getSum(String[] s1)

{

int sum = 0;

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

{

String s = s1[i];

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

{

Character c = s.charAt(j);

if(Character.isDigit(c))

{

sum+=Integer.parseInt(s.valueOf(c));

}

}

}

System.out.println(sum);

}

}