51. 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.*;
public class Main {
public static void main(String[] args) {
double d=845.69;
System.out.println(noOfDigits(d));
}
public static String noOfDigits(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.StringTokenizer;
publicclass User{
publicstatic String noOfDigits(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(n2-1)=='0')
n2=s2.length()-1;
//String s3=String.valueOf(n1)+":"+String.valueOf(n2);
StringBuffer sb=new StringBuffer();
sb.append(n1).append(":").append(n2);
return sb.toString();
}
}
```

52. Employees & Designations

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

```
class Main
{
	public static void main(String[] arg)
	{

	Scanner sc=new Scanner(System.in);
	int n=sc.nextInt();
		sc.nextLine();
		HashMap<String,String> hm=new HashMap<String,String>();

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

```
        {
                hm.put(sc.nextLine(),sc.nextLine());
        }
                String b=sc.nextLine();

                HashMap<String,String> op=new HashMap<String,String>();
                op=MainClass.obtainDesig(hm,b);
                Iterator<String> itr=op.keySet().iterator();
                while(itr.hasNext())
                {
                        String key=itr.next();
                                System.out.println(key);
                                String value=hm.get(key);
                                System.out.println(value);

                }
    }}
```

```
import java.util.HashMap;
```

```
import java.util.Iterator;
import java.util.LinkedHashMap;


public class MainClass {
	public static  HashMap<String,String> obtainDesig(HashMap<String,String> hm,String s)
	{
		LinkedHashMap<String,String> op=new LinkedHashMap<String,String>();
		Iterator<String> itr=hm.keySet().iterator();

		while(itr.hasNext())
		{
			String key=itr.next();
			String value=hm.get(key);
			if(s.equals(value))
			{
				op.put(key,value);
			}


		}
		return op;
			}
}
```

53.Grade Calculator

A School wants to give 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 name, mark in the given order. The datatype for name is string, mark is float.

You decide to build a hashmap. The hashmap contains name as key and mark as value.

BUSINESS RULE:

1. If Mark is less than 60, then grade is FAIL.
2. If Mark is greater than or equal to 60, then grade is PASS.

Note: FAIL/PASS should be in uppercase.

Store the result in a new Hashmap with name as Key and grade as value.

4. You decide to write a function **calculateGrade** which takes the above hashmap as input and returns the hashmap as output. Include this function in class UserMainCode.

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

Input and Output Format:

Input consists of student details. The first number indicates the size of the students. The next two values indicate the name, mark.

Output consists of a name and corresponding grade for each student.

Refer sample output for formatting specifications.

Sample Input 1:

```
3
Avi
76.36
Sunil
68.42
Raja
36.25
```

Sample Output 1:

Avi
PASS
Sunil
PASS
Raja
FAIL

```
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;

}

}

(Or)

```
import java.util.*;
publicclass Main {
publicstaticvoid main(String[] args) {
      Scanner sc=new Scanner(System.in);

int n=sc.nextInt();
LinkedHashMap<String,Float> ip=new LinkedHashMap<String,Float>();
for(int i=0;i<n;i++)
{
      ip.put(sc.next(),sc.nextFloat());
}

LinkedHashMap<String,String> op=new LinkedHashMap<String,String>();
op=User.noOfDigits(ip);
Iterator<String> itr= op.keySet().iterator();
while(itr.hasNext())
{
      String key=itr.next();
      System.out.println(key);
      String value=op.get(key);
      System.out.println(value);

}
}}
import java.util.HashMap;
import java.util.Iterator;
```

```
import java.util.LinkedHashMap;
publicclass User{
publicstatic LinkedHashMap<String,String>  noOfDigits(HashMap<String,Float>
hm) {

	LinkedHashMap<String,String> op=new LinkedHashMap<String,String>();
	Iterator<String> itr=hm.keySet().iterator();
	String res=null;
		for(int i=0;i<hm.size();i++)
		{
	while(itr.hasNext())
	{
		String key=itr.next();

		float value=hm.get(key);
		if(value>=60)
			res="pass";
		else
			res="fail";
		op.put(key,res);


	}
		}

	return op;
}
}
```

```
import java.util.HashMap;
import java.util.Iterator;
import java.util.LinkedHashMap;
publicclass User{
publicstatic LinkedHashMap<String,String>  noOfDigits(HashMap<String,Float>
hm) {

	LinkedHashMap<String,String> op=new LinkedHashMap<String,String>();
	Iterator<String> itr=hm.keySet().iterator();
	String res=null;
			while(itr.hasNext())
	{
		String key=itr.next();

		float value=hm.get(key);
		if(value>=60)
			res="pass";
		else
			res="fail";
		op.put(key,res);

	}

	return op;
}
```

}

54. DOB - Validation

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");
}
}
}
```

55. Experience Validator

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

```
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.getvalues(s,s1));
    }
}
import java.util.Calendar;

import java.util.Date;



public class UserMainCode {
            public static boolean getvalues(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;


}}
```

56. 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

```
import java.util.ArrayList;

import java.util.Scanner;


class Main

{

	public static void main(String[] arg)

	{

		Scanner sc=new Scanner(System.in);

		int n=sc.nextInt();

		sc.nextLine();


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


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

```
		{
			aa.add(sc.nextLine());
		}

		String a[]=MainClass.convertToString( aa);
		for(int i=0;i<a.length;i++)
		{
			System.out.println(a[i]);
		}


	}}

import java.util.ArrayList;
import java.util.Collections;


public class MainClass {


	public static  String[] convertToString(ArrayList<String> a1)
	{
		Collections.sort(a1);// uses to sort arraylist string
		String a[]=new String[a1.size()];
```

```
        a1.toArray(a);
        return a;

}
}
```

57. State ID generator

Write a program to generate the state ID.

1)Read n Strings as input(as State Name).
2)Create a String Array to Store the above Input.
3)Write a function **getStateId** which accepts String Array as input.
4)Create a HashMap<String,String> which stores state name as key and state Id as Value.
5)The function getStateId returns the HashMap to the Main Class.

Include UserMainCode Class With static method **getStateId** which accepts String array and return a hashmap.

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 n denotes the size of the string array.
Output consists of an HashMap displayed in the string array order.

Sample Input 1:
3
Kerala
Gujarat
Goa

Sample Output 1:
KER:Kerala

GUJ:Gujarat

GOA:Goa

```
import java.util.*;
publicclass Main
{
	publicstaticvoid main(String[] args)
	{

	Scanner sc=new Scanner(System.in);

		int n=sc.nextInt();
		String s1[]=new String[n];
		for(int i=0;i<n;i++)
		{
	 s1[i]=sc.next();
		}


		LinkedHashMap<String,String>ip=new
LinkedHashMap<String,String>();
		ip=User.Method(s1);
		Iterator<String> itr=ip.keySet().iterator();

		//while(itr.hasNext())
		for(int i=0;i<ip.size();i++)
		{
			String key=itr.next();
			String value=ip.get(key);
			System.out.println(value+""+key);
		}

	}
}
```

```
import java.util.LinkedHashMap;
publicclass User
{
publicstatic LinkedHashMap<String,String>  Method(String[] s1)
{
	LinkedHashMap<String,String> op=new LinkedHashMap<String,String>();

	for(int i=0;i<s1.length;i++)
	{
		StringBuffer sb=new StringBuffer();
	StringBuffer key=sb.append(s1[i].substring(0,3)).append(":");
	op.put(s1[i],key.toString().toUpperCase());

	}
	return op;
}
```

```
}
```

(or)

STATE id

```
  import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.util.HashMap;



public class Main {

      public static void main(String[] args) {
            BufferedReader br=new BufferedReader(new
InputStreamReader(System.in));
            try
            {
                  int n=Integer.parseInt(br.readLine());
                  String[] input=new String[n];
                  for(int i=0;i<n;i++)
                  {
                        input[i]=br.readLine();
                  }
                  HashMap<String,String>hm=UserMainCode.costEst
              imator(input);
                  for(int i=0;i<n;i++)
                  {
                        String s=input[i];
                        String key=hm.get(s);
                        System.out.println(key+":"+s);
                  }
            }


catch(Exception e)
            {

            }

      }

}
```

```
  import java.util.HashMap;

public class UserMainCode {
	public static HashMap<String,String>
costEstimator(String[] name)
	{
		int n=name.length;
		HashMap<String, String> hm=new HashMap<String,
String>();
		for(int i=0;i<n;i++)
		{
			String sub=name[i].substring(0, 3);
			hm.put(name[i],sub.toUpperCase());
		}
		return hm;
	}

}
```

58.ArrayList to String Array

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.*;
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;
}
```

```
import java.util.*;
publicclass Main {
publicstaticvoid main(String[] args) {
      List<String> l1=new ArrayList<String>();
      Scanner sc=new Scanner(System.in);
      int n=sc.nextInt();
      for(int i=0;i<n;i++)
      {
            l1.add(sc.next());
      }
      List<String> l2=new ArrayList<String>();

      int n1=sc.nextInt();
      for(int i=0;i<n1;i++)
      {
            l2.add(sc.next());
      }

      String[] s2=User.fruitsList(l1,l2);
      // for(int i=0;i<s2.length;i++)
```

```
            // System.out.println(s2[i].toString());
      for(String s3:s2)
            System.out.println(s3);

}
}


publicclass User
{
publicstatic String[] fruitsList(List<String> l1, List<String> l2){
      ArrayList<String> l3=new ArrayList<String>();
      for(int i=0;i<l1.size();i++)
      {
            String s1=l1.get(i);
            intlen=s1.length();
      if(s1.charAt(len-1)!='a'&& s1.charAt(len-1)!='A'
            && s1.charAt(len-1)!='e'&& s1.charAt(len-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;
}

}



]
   /'
```

59. Elements in ArrayList

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.*;
publicclass Main
{
	publicstaticvoid main(String[] args)
	{
		int n,m;
		Scanner sc = new Scanner(System.in);
		n = sc.nextInt();
		ArrayList<Integer> a1 = new ArrayList<Integer>();
		for(int i=0;i<n;i++)
		{

			a1.add(sc.nextInt());
		}
		m = sc.nextInt();
		ArrayList<Integer> a2 = new ArrayList<Integer>();
		for(int i=0;i<m;i++)
		{

			a2.add(sc.nextInt());
		}
		int[] result = User.arrayListSubtractor(a1, a2);
		Arrays.sort(result);
		for(int i=0;i<result.length;i++)
			System.out.println(result[i]);
}
}
```

```
import java.util.*;

publicclass User
{


	publicstaticint[] arrayListSubtractor(ArrayList<Integer>
	arrlist1,ArrayList<Integer> arrlist2)
	{
	TreeSet<Integer> ts1=new TreeSet<Integer>();
	TreeSet<Integer> ts2=new TreeSet<Integer>();
	TreeSet<Integer> ts3=new TreeSet<Integer>();
	ArrayList<Integer> aa=new ArrayList<Integer>();
	for(int i=0;i<arrlist1.size();i++)
	ts1.add(arrlist1.get(i));

	for(int i=0;i<arrlist2.size();i++)
		ts2.add(arrlist2.get(i));

	ts1.addAll(ts2);

	for(int i=0;i<arrlist1.size();i++)
	{
	for(int j=0;j<arrlist2.size();j++)
	{
	if(arrlist1.get(i)==arrlist2.get(j))
	ts3.add(arrlist1.get(i));
	}
	}
	ts1.removeAll(ts3);
	aa.addAll(ts1);
	int res[]=newint[aa.size()];
	for(int i=0;i<res.length;i++)
	res[i]=aa.get(i);
	return res;


	}
}
```

60. 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 UserMainCode {

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(getTheTotalCostOfPheripherals(m1,s));

}

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;

}}
```