

Q) Validating Date Format

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

Main:

=====

```
import java.util.*;
public class Main {
    public static void main (String [] args)
    {
    }
```

UserMainCode:

=====

```
import java.util.*;
import java.text.*;
public class UserMainCode{
    public static int validateDate(String s1){

    }
}
```

Q) Validate Time

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

Main:

=====

```
import java.util.*;
public class Main {
    public static void main (String [] args)
    {

    }
}
```

UserMainCode:

=====

```
import java.util.*;
import java.text.*;
public class UserMainCode{
    public static int validateDate(String str){
```

```
}}
```

15. Difference between largest and smallest elements in an array

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
```

Main:

```
=====
```

```
import java.util.*;
public class Main {
    public static void main (String [] args)
```

```
{  
}
```

UserMainCode:

=====

```
import java.util.*;  
import java.text.*;  
public class UserMainCode{  
public static int getBigDiff(int[] a, int n){  
  
}}
```

16. Find the element position in a reversed string array

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 position 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

Main:

=====

```
import java.util.*;  
public class Main {  
    public static void main (String [] args)  
    {  
  
    }  
}
```

UserMainCode:

=====

```
import java.util.*;  
import java.text.*;  
public class UserMainCode{  
    public static void getElementPosition(String[] a, String b) {  
    }  
}}
```


18. Calculate Electricity Bill

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

Main:

=====

```
import java.util.*;
public class Main {
public static void main (String [] args)
{
}
}
```

UserMainCode:

=====

```
import java.util.*;
import java.text.*;

public class UserMainCode{
public static int calculateElectricityBill(String input1,String input2,int input3) {

}}
}
```

Q)Removing Keys from HashMap

Given a method with a `HashMap<Integer,string>` 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<integer,string>`.

Output is an integer which is the size of the hashmap. Refer sample output for formatting specifications.

SampleInput1:

```
3
2
hi
4
Hell

12

Helloworld
```

SampleOutput1:

```
1
```


SampleInput2:

3
2
hi
4
sdfsdf
3
asdf

SampleOutput2:

2

Main

====

```
import java.util.*;  
public class Main {  
    public static void main (String [] args)  
    {  
  
    }
```

UserMainCode:

=====

```
import java.util.*;  
import java.text.*;
```

```
public class UserMainCode{  
    public static int sizeOfResultandHashMap(HashMap<Integer,String> hm){  
    }  
}
```

Q) Sum of Lowest marks

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
```

Main

=====

```
import java.util.*;
public class Main{
public static void main(String[] args){

}
```

UserMainCode

=====

```
import java.util.*;
public class UserMainCode {
public static int getLowest(HashMap<Integer, Integer> h1){

}}
```

Q)ArrayList Sorting and Merging

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

MAIN:

=====

```
Import java.util.*;  
Public class Main {
```

```
Public static void main (String [] args)
{

}}
USERMAINCODE:
=====
public class UserMainCode {
public static ArrayList<Integer> sortMergedArraylist (ArrayList<Integer>
list1, ArrayList<Integer>list2) {

}}
```