IRC_SKCT_Java2_SB_COD_Collection

Test Summary

- No. of Sections: 1
- No. of Questions: 5
- Total Duration: 120 min

Section 1 - Coding

Section Summary

- No. of Questions: 5
- Duration: 120 min

Additional Instructions:

None

Q1. Create a class **ArrayListMain** and in the main method get the names and store them in an ArrayList. After getting all the names, just display them in the same order.

Input Format

Number of names(N) in first line as integer
N names in separate lines

Output Format

Print the names

Sample Input

```
6
KL Rahul
Hetmyer
Pierre
```

Sample Output

```
KL Rahul
Hetmyer
Pierre
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q2. Input a positive integer N (N > 0), input N strings, and sort the strings in place in the order of increasing length. Print the sorted strings using ArrayList as an implementation of the List interface for storing the individual strings.

Input Format

Input number of elements
Input each string on a separate line

Output Format

Print the list of strings sorted by their length

Sample Input

```
3
aa
b
```

Sample Output

```
[b, aa, ccc]
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q3. Using Java Library ArrayList as a List Interface implementation, input N integers from standard input and add to the list only if they form an increasing sequence.

1. Take a number, N > 0 as input
2. Accept N integers as input
3. Add the number to the list only if it forms an increasing sequence else ignore
4. Print the list

Input Format

Input number of elements, N > 0
Enter each integer on the next N lines

Output Format

List of integers in increasing sequence ignoring out of order elements

Sample Input

```
7
3
5
```

Sample Output

```
[3, 5, 9, 11, 13]
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q4. **Frequency()**
While entering user names, We have to be very careful about the duplicate entries in the list.
To make a correct and perfect report, we have to remove the duplicate elements in the list. Write a program that obtains a set of names and a search element and prints its frequency.

Input Format

The first line of the input consists of the number of names.
The next input is the user names.
The last input is the user name to be searched.

Output Format

The output prints the frequency of the searched element.

Sample Input

```
5
alice
bob
```

Sample Output

```
2
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q5. **sort() a List of Objects**
Write a program to take hall objects as input in the list and sort them in the order of their costPerDay using the sort() method of the comparable interface. Then display them in tabular form.
Create a class **Hall** with the following attributes,

Attribute	**Data type**
name	String
contactNumber	String
costPerDay	Double
ownerName	String

Mark the attributes as private and add appropriate getter/setter, default, and parameterized constructor. Override toString() and print the details in a tabular format. And implement comparable interface in the class.
Create driver class Main and use the main method to get inputs, sort, and display.

Input Format

The first line has the number of halls n.
The next n lines have details of the hall

Output Format

The output displays the hall details
Refer sample output

Sample Input

```
3
SDH hall
12345
```

Sample Output

```
SDH hall 12345 12000.0 Jane
XUV hall 24680 15000.0 Jack
SRT hall 13579 20000.0 John
```

Sample Input

```
6
SDH hall
12345
```

Sample Output

```
DFG hall 24680 10000.0 Jack
SDH hall 12345 12000.0 Jane
XUV hall 24680 15000.0 Jack
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Answer Key & Solution

Section 1 - Coding

Q1

Test Case

Input

```
4
Hetmyer
Dube
```

Output

```
Hetmyer
Dube
Walsh
```

Weightage - 25

Input

```
12
V Kohli
Simmons
```

Output

```
V Kohli
Simmons
Williams
```

Weightage - 25

Input

```
5
V Kohli
Simmons
```

Output

```
V Kohli
Simmons
Williams
```

Weightage - 25

Input

```
7
King
Walsh
```

Output

```
King
Walsh
RA Jadeja
```

Weightage - 25

Sample Input

```
6
KL Rahul
Hetmyer
```

Sample Output

```
KL Rahul
Hetmyer
Pierre
```

Solution

```
 import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;
import java.util.Scanner;

class ArrayListMain {
    public static void main(String args[]) {
        List<String> names = new ArrayList<>();
        Scanner sc = new Scanner(System.in);
        int n =Integer.parseInt(sc.nextLine());
        for (int i=0;i<n;i++)
            names.add(sc.nextLine());

        Iterator it = names.iterator();
        while(it.hasNext()) {
            System.out.println(it.next());
    }

}
}
```

Q2

Test Case

Input

```
3
111
22
3333
```

Output

```
[22, 111, 3333]
```

Weightage - 80

Input

```
1
3a
```

Output

```
[3a]
```

Weightage - 10

Input

```
3
23
111
0000
```

Output

```
[23, 111, 0000]
```

Weightage - 10

Sample Input

```
3
aa
b
ccc
```

Sample Output

```
[b, aa, ccc]
```

Solution

```
import java.util.*;
import java.lang.*;
import java.io.*;

class Q01Simple_Sort
{
    public static void main (String[] args) throws java.lang.Exception
    {
        Scanner input = new Scanner(System. in);
        // Input number of elements
        int number_of_elements = input. nextInt();
        input. nextLine();
        if (number_of_elements <= 0) return;
        List<String> list = new ArrayList<>();
        for (int ctr = 0; ctr < number_of_elements; ctr++) {
            // Input next string
            String str = input. nextLine();
            list.add(str);
        }
        Collections.sort(list, new Comparator<String>() {
            public int compare(String o1, String o2) {
            return o1.length() - o2.length();
        }});
        System.out.println(list);
    }
}
```

Q3

Test Case

Input

```
3
5
11
```

Output

```
[5, 11]
```

Weightage - 25

Input

```
1
3
```

Output

```
[3]
```

Weightage - 25

Input

```
5
1
3
```

Output

```
[1, 3, 5]
```

Weightage - 25

Input

```
7
3
5
```

Output

```
[3, 5, 9, 11, 15]
```

Weightage - 25

Sample Input

```
7
3
5
```

Sample Output

```
[3, 5, 9, 11, 13]
```

Solution

```
 import java.util.*;
import java.lang.*;
import java.io.*;

class Q01Simple_List
{
    public static void main (String[] args) throws java.lang.Exception
    {
        Scanner input = new Scanner(System. in);
        // Inputnumber of elements
        int number_of_elements = input. nextInt();
        if (number_of_elements <= 0) return;
        ArrayList<Integer> numList = new ArrayList<Integer>();
        for (int ctr = 0; ctr < number_of_elements; ctr++) {
            // Input next element
            int num = input. nextInt();
            ListIterator<Integer> listIter = numList.listIterator(numList.size());
            if (listIter.hasPrevious()) {
                if( listIter.previous() < num)
                    numList.add(num);
            } else
                numList.add(num);
        }
        System.out.println(numList);
    }
}
```

Q4

Test Case

Input

```
5
alice
bob
```

Output

```
2
```

Weightage - 20

Input

```
6
alice
harry
```

Output

```
3
```

Weightage - 20

Input

```
7
harry
alice
```

Output

```
5
```

Weightage - 20

Input

```
8
harry
harry
```

Output

```
7
```

Weightage - 20

Input

```
10
ron
harry
```

Output

```
0
```

Weightage - 20

Sample Input

```
5
alice
bob
ankit
```

Sample Output

```
2
```

Solution

```
 import java.io.*;
import java.util.*;
class Main {
public static void main(String [] args) {
    int i,n;
    Scanner sc = new Scanner(System.in);
    n = Integer.parseInt(sc.nextLine());
    ArrayList<String> names = new ArrayList<String>(n);
    for(i=0;i<n;i++) {
        names.add(sc.nextLine());
    }
    String search = sc.nextLine();
    System.out.println(Collections.frequency(names, search));
}
}
```

Q5

Test Case

Input

```
3
SDH hall
12345
12000.0
```

Output

```
SDH hall 12345 12000.0 Jane
XUV hall 24680 15000.0 Jack
SRT hall 13579 20000.0 John
```

Weightage - 20

Input

```
4
SDH hall
12345
12000.0
```

Output

```
DFG hall 24680 10000.0 Jack
SDH hall 12345 12000.0 Jane
XUV hall 24680 15000.0 Jack
SRT hall 13579 20000.0 John
```

Weightage - 20

Input

```
5
SDH hall
12345
12000.0
```

Output

```
DFG hall 24680 10000.0 Jack
SDH hall 12345 12000.0 Jane
XUV hall 24680 15000.0 Jack
SRT hall 13579 20000.0 John
```

Weightage - 20

Input

```
6
SDH hall
12345
12000.0
```

Output

```
DFG hall 24680 10000.0 Jack
SDH hall 12345 12000.0 Jane
XUV hall 24680 15000.0 Jack
SRT hall 13579 20000.0 John
```

Weightage - 20

Input

```
7
SDH hall
12345
12000.0
```

Output

```
DFG hall 24680 10000.0 Jack
SDH hall 12345 12000.0 Jane
ABC hall 12345 13000.0 John
XUV hall 24680 15000.0 Jack
```

Weightage - 20

Sample Input

```
3
SDH hall
12345
12000.0
```

Sample Output

```
SDH hall 12345 12000.0 Jane
XUV hall 24680 15000.0 Jack
SRT hall 13579 20000.0 John
```

Sample Input

```
6
SDH hall
12345
12000.0
```

Sample Output

```
DFG hall 24680 10000.0 Jack
SDH hall 12345 12000.0 Jane
XUV hall 24680 15000.0 Jack
SRT hall 13579 20000.0 John
```

Solution

```java
import java.io.*;
import java.util.*;
class Hall implements Comparable<Hall> {
    public Hall(String name, String contactNumber, double costPerDay, String ownerName) {
        this.name = name;
        this.contactNumber = contactNumber;
        this.costPerDay = costPerDay;
        this.ownerName = ownerName;
    }
    public Hall() {
        this.name = null;
        this.contactNumber = null;
        this.costPerDay = 0;
        this.ownerName = null;
    }

    private String name;
    private String contactNumber;
    private double costPerDay;
    private String ownerName;


    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getContactNumber() {
        return contactNumber;
    }

    public void setContactNumber(String contactNumber) {
        this.contactNumber = contactNumber;
    }
    public double getCostPerDay() {
        return costPerDay;
    }

    public void setCostPerDay(double costPerDay) {
        this.costPerDay = costPerDay;
    }

    public String getOwnerName() {
        return ownerName;
    }

    public void setOwnerName(String ownerName) {
        this.ownerName = ownerName;
    }
    @Override
    public int compareTo(Hall h) {

        return Double.compare(this.costPerDay,h.costPerDay );
    }
    public String toString() {
        return name+" "+contactNumber+" "+costPerDay+" "+ownerName;
    }
}
class Main {
public static void main(String [] args) {
    int i,n;
    Scanner sc = new Scanner(System.in);
    n = Integer.parseInt(sc.nextLine());
    Hall [] h = new Hall[n];
    ArrayList<Hall> halls = new ArrayList<Hall>(n);
    for(i=0;i<n;i++) {
        h[i] = new Hall();
        h[i].setName(sc.nextLine());
        h[i].setContactNumber(sc.nextLine());
        h[i].setCostPerDay(Double.parseDouble(sc.nextLine()));
        h[i].setOwnerName(sc.nextLine());
        halls.add(h[i]);
    }
    Collections.sort(halls);
    for(i=0;i<n;i++) {
        System.out.println(halls.get(i));
    }
}
}
```