Contents

Java – First Problem Statement

Java – Second Problem Statement

Java – First Problem Statement

Question

Write main method in **Solution** class.

In the main method, read a String value and print the count of lower case characters present in the input String value. If no lower case characters are present in the String value then it should print "No lower case characters present" as a String.

Sample input1:

XpLore

Output:

4

Sample input2:

XPLORE

Output:

No lower case characters present

Sample code snippet for reference:

Please use below code to build your MyClass.

```
public class MyClass
{

        public static void main(String[] args)
```

```
    {
        //code to read values
        //code to display the result
    }

}
```

Java – First Problem Statement Solution

```
public class Solution {
    public static void main(String args[]){
        int lower=0;
        Scanner sc=new Scanner(System.in);
        String s1 = sc.nextLine();
        for(int i=0; i<s1.length(); i++){
            char ch=s1.charAt(i);
            if(ch>='a' && ch<='z') {
                lower++;
            }
        }
        if(lower>0)
            System.out.println(lower);
        else
            System.out.println("No lower case characters present");
    }
}
```

Test Cases- First Problem Statement

Test Case1

XpLore

Output:

4

Test Case2:

XPLORE

Output:

No lower case characters present

Test Case3:

Morning

Output:

6

Test Case4:

MORNINg

Output:

1

Java- Second Problem Statement

Question

Create a class HeadSets with below attributes:

headsetName - String

brand-String

price - int

available- boolean

The above attributes should be private, write getters, setters and parameterized constructor as required.

Create class Solution with main method.

Implement two static methods - findTotalPriceForGivenBrand and findAvailableHeadsetWithSecondMinPrice in Solution class.

findTotalPriceForGivenBrand method:
The method will return the total price of Headsets from array of Headset objects for the given brand(String parameter passed).
If no Headsets with the given brand is present in the array of Headsets objects, then the method should return 0.

findAvailableHeadsetWithSecondMinPrice method:
This method will take array of Headset objects as an input parameter and returns the available Headset object with the second lowest/minimum price from the given array of objects.
If no Headsets with the above condition is present in the array of Headsets objects, then the method should return null.

Note : No two Headsets will have the same price.

All the searches should be case insensitive.

The above mentioned static methods should be called from the main method.

For findTotalPriceForGivenBrand method - The main method should print the returned total price as it is if the returned value is greater than 0 or it should print "No Headsets available with the given brand".

Eg: 4500
where 4500 is the total Headsets price

For findAvailableHeadsetWithSecondMinPrice method - The main method should print the headsetsName and price from the returned Headsets object if the returned value is not null.
If the returned value is null then it should print "No Headsets available".

Eg:Logitech Wired
1500
where Logitech Wired is the headsetName and 1500 is the price

Before calling these static methods in main, use Scanner object to read the values of four Headset objects referring attributes in the above mentioned attribute sequence.
Next, read the value of String parameter for capturing brand.

Consider below sample input and output:

Input1:

boAt BassHeads

boAt

1220

```
true
Over Ear Wired
boAt
549
true
In Ear with Mic
JBL
450
true
Buds 2 Neo
RealMe
500
true
boAt
```

Output:

```
1769
Buds 2 Neo
500
```

Input2:

```
boAt BassHeads
boAt
1220
false
Over Ear Wired
boAt
1549
false
```

```
In Ear with Mic
JBL
450
false
Buds 2 Neo
RealMe
500
false
boAt
```

Output:

```
2769
No Headsets available
```

Sample code snippet for reference:

Please use below code to build your solution.

```
import java.util.Scanner;
public class MyClass
{

 public static void main(String[] args)
 {
  //code to read values
  //code to call required method
  //code to display the result
 }

public static int findTotalPriceForGivenBrand (/* required parameters to be added */)
  {
   //method logic
  }
public static HeadSets findAvailableHeadsetWithSecondMinPrice (/* required parameters to be added */)
   //method logic
  }
}
class HeadSets
{
  //code to build the class
}
```

Note on using Scanner object:

Sometimes scanner does not read the new line character while invoking methods like nextInt(), nextDouble() etc.

Usually, this is not an issue, but this may be visible while calling nextLine() immediately after those methods.

Consider below input values:

1001

Savings

Referring below code:

```
Scanner sc = new Scanner(System.in);
int x = sc.nextInt();
String str = sc.nextLine(); -> here we expect str to have value Savings.Instead it may be "".
```

If above issue is observed, then it is suggested to add one more explicit call to nextLine() after reading numeric value.

Java – Second Problem Statement Solution

```
import java.util.Scanner;

public class Solution{

    public static void main(String[] args) {

        Headsets[] hs = new Headsets[4];

        Scanner sc = new Scanner(System.in);
        for (int i = 0; i < hs.length; i++) {
            String headsetName = sc.nextLine();
            String brand = sc.nextLine();
            int price = sc.nextInt();
            boolean available = sc.nextBoolean();
   sc.nextLine();
   hs[i] = new Headsets(headsetName, brand, price, available);
        }
        String new_brand = sc.nextLine();
        int sum = findTotalPriceForGivenBrand (hs,new_brand);
        if (sum>0) {
            System.out.println(sum);
        } else {
            System.out.println("No Headsets available with the given brand");
        }
        Headsets res1 = findAvailableHeadsetWithSecondMinPrice(hs);
     if (res1 != null) {
            System.out.println(res1.getHeadsetName());
            System.out.println(res1.getPrice());
        } else{
```

```
        System.out.println("No Headsets available");
    }





}

public static int findTotalPriceForGivenBrand (Headsets[] hs, String brand) {

    int sum=0;
    for(int i=0;i<hs.length;i++) {
            if( hs[i].getBrand().equalsIgnoreCase(brand)) {
                sum = sum + hs[i].getPrice();
            }
        }
    return sum;
}

public static Headsets findAvailableHeadsetWithSecondMinPrice (Headsets[] hs) {
    int count = 0;
    Headsets temp = null;

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

        if (hs[i].isAvailable()) {
            count++;
        }
    }
    Headsets[] newheadset = new Headsets[count];
    int k=0;
```

```
                for (int i = 0; i < hs.length; i++) {
                        if (hs[i].isAvailable()) {
                                newheadset[k] = hs[i];
                                k++;
                        }
                }
                for (int i = 0; i < newheadset.length; i++) {
                        for (int j = i + 1; j < newheadset.length; j++) {
                                if (newheadset[i].getPrice()>newheadset[j].getPrice()) {
                                        temp = newheadset[i];
                                        newheadset[i] = newheadset[j];
                                        newheadset[j] = temp;
                                }

                        }
                }
                        if(count>0){
                                return newheadset[1];
                        }
                        else{
                                return null;
                        }
        }


}

class Headsets{

        private String headsetName;
        private String brand;
        private int price;
        private boolean available;
        public String getHeadsetName() {
                return headsetName;
        }
        public void setHeadsetName(String headsetName) {
                this.headsetName = headsetName;
        }
        public String getBrand() {
                return brand;
        }
        public void setBrand(String brand) {
                this.brand = brand;
        }
        public int getPrice() {
                return price;
        }
        public void setPrice(int price) {
                this.price = price;
        }
        public boolean isAvailable() {
                return available;
        }
        public void setAvailable(boolean available) {
                this.available = available;
        }
        public Headsets(String headsetName, String brand, int price,
                        boolean available) {
                super();
                this.headsetName = headsetName;
                this.brand = brand;
```

```
                    this.price = price;
                    this.available = available;
          }

}
```

Test Cases – Second Problem Statement

Test Case1:

```
boAt BassHeads
boAt
1220
true
Over Ear Wired
boAt
549
true
In Ear with Mic
JBL
450
true
Buds 2 Neo
RealMe
500
true
boAt
```

Output:

```
1769
Buds 2 Neo
500
```

Test Case2:

```
boAt BassHeads
boAt
1220
false
Over Ear Wired
boAt
1549
false
In Ear with Mic
JBL
450
false
Buds 2 Neo
RealMe
500
false
boAt
```

Output:

```
2769
No Headsets available
```

Testcase 3:

```
Bluetooth Wireless
Sony
2690
true
Wired InEar with Mic
Philips
```

```
455
false
Wireless In Ear
Sony
2799
true
Live Over Ear with Mic
JBL
5999
true
sony
```

Output:

```
5489
Wireless In Ear
2799
```

Testcase 4:

```
Bluetooth Wireless
Sony
2890
true
Wired InEar with Mic
Philips
455
false
Wireless In Ear
Sony
2799
true
Live Over Ear with Mic
JBL
```

```
5999
true
apple
```

Output:

```
No Headsets available with the given brand
Bluetooth Wireless
2890
```