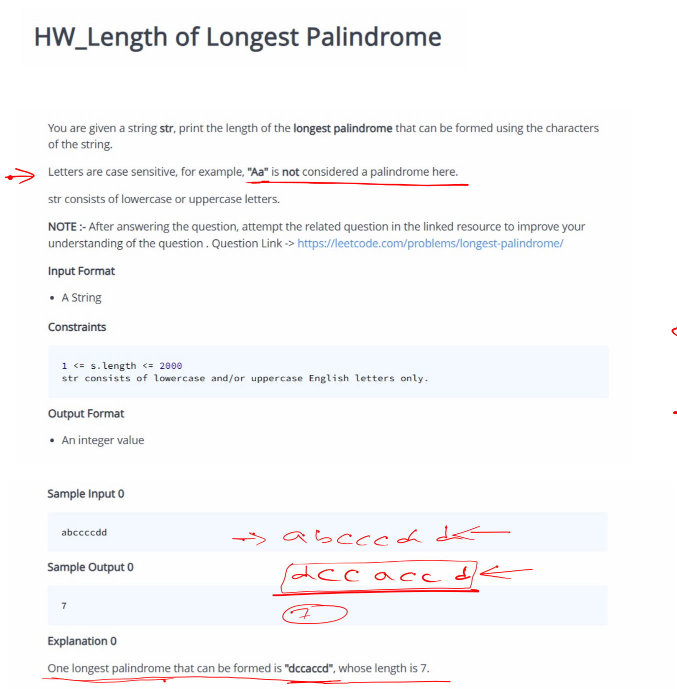


HW_Length of Longest Palindrome

You are given a string **str**, print the length of the **longest palindrome** that can be formed using the characters of the string.

Letters are case sensitive, for example, **"Aa"** is **not** considered a palindrome here.

str consists of lowercase or uppercase letters.

NOTE :- After answering the question, attempt the related question in the linked resource to improve your understanding of the question . Question Link -> https://leetcode.com/problems/longest-palindrome/

Input Format

- A String

Constraints

```
1 <= s.length <= 2000
str consists of lowercase and/or uppercase English letters only.
```

Output Format

- An integer value

Sample Input 0

```
abccccdd
```

abcccdd

Sample Output 0

dccaccd

```
7
```

7

Explanation 0

One longest palindrome that can be formed is **"dccaccd"**, whose length is 7.

DAD

DAD

AA

aa

A a

AA aa

abcccd d

d d c c c b a

5 b b b b b

1

a – 4

b – 3

a a a a – 4

b b b

2

1

a b c c c c d d

1 1 4 2

6

even

even odd

NITIN

N – 2 – 3

I – 2

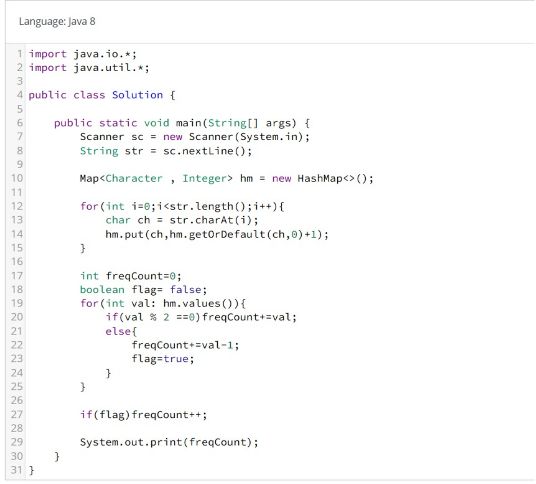
T – 1

2, 4, 6, 8, 12

2, 12

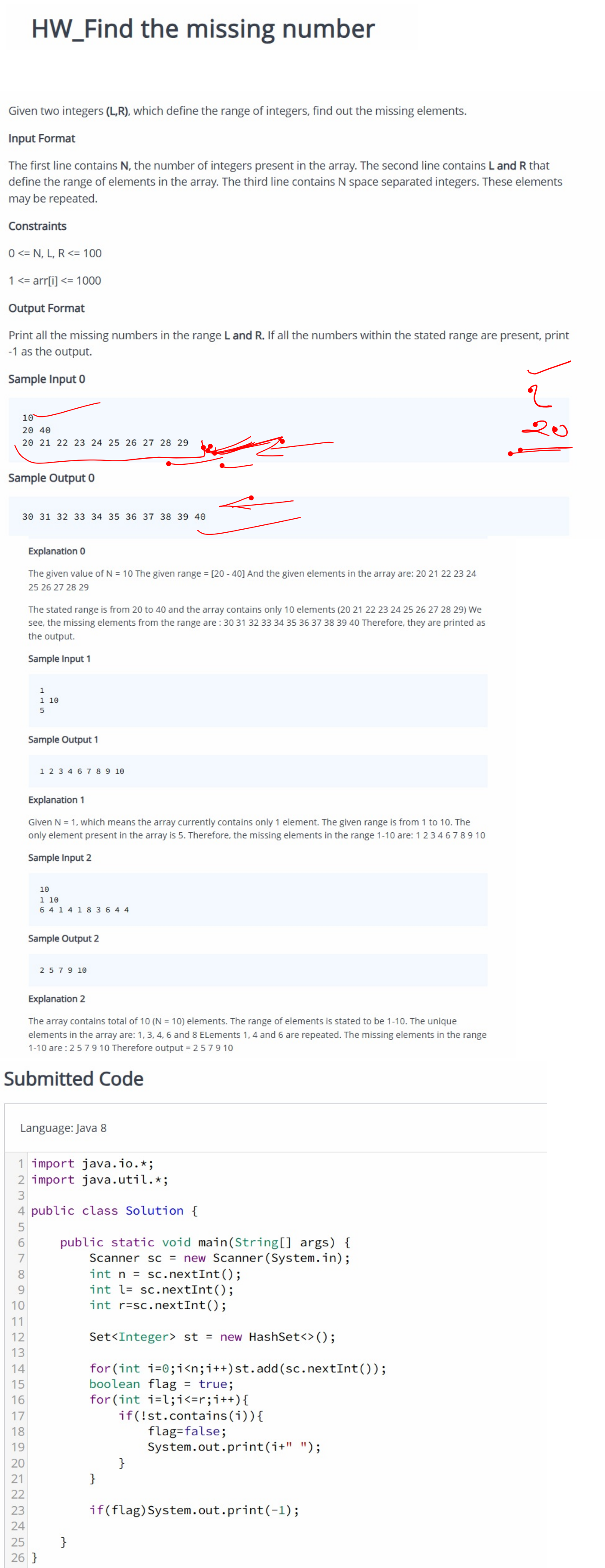
a, a

Submitted Code



Language: Java 8

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

public class Solution {

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        String str = sc.nextLine();

        Map<Character , Integer> hm = new HashMap<>();

        for(int i=0;i<str.length();i++){
            char ch = str.charAt(i);
            hm.put(ch,hm.getOrDefault(ch,0)+1);
        }

        int freqCount=0;
        boolean flag= false;
        for(int val: hm.values()){
            if(val % 2 ==0)freqCount+=val;
            else{
                freqCount+=val-1;
                flag=true;
            }
        }

        if(flag)freqCount++;

        System.out.print(freqCount);
    }
}
```



HW_Find the missing number

Given two integers **(L,R)**, which define the range of integers, find out the missing elements.

Input Format

The first line contains **N**, the number of integers present in the array. The second line contains **L and R** that define the range of elements in the array. The third line contains N space separated integers. These elements may be repeated.

Constraints

0 <= N, L, R <= 100

1 <= arr[i] <= 1000

Output Format

Print all the missing numbers in the range **L and R.** If all the numbers within the stated range are present, print -1 as the output.

Sample Input 0

```
10
20 40
20 21 22 23 24 25 26 27 28 29
```

L
20

R
40

Sample Output 0

```
30 31 32 33 34 35 36 37 38 39 40
```

Explanation 0

The given value of N = 10 The given range = [20 - 40] And the given elements in the array are: 20 21 22 23 24 25 26 27 28 29

The stated range is from 20 to 40 and the array contains only 10 elements (20 21 22 23 24 25 26 27 28 29) We see, the missing elements from the range are : 30 31 32 33 34 35 36 37 38 39 40 Therefore, they are printed as the output.

Sample Input 1

```
1
1 10
5
```

Sample Output 1

```
1 2 3 4 6 7 8 9 10
```

Explanation 1

Given N = 1, which means the array currently contains only 1 element. The given range is from 1 to 10. The only element present in the array is 5. Therefore, the missing elements in the range 1-10 are: 1 2 3 4 6 7 8 9 10

Sample Input 2

```
10
1 10
6 4 1 4 1 8 3 6 4 4
```

Sample Output 2

```
2 5 7 9 10
```

Explanation 2

The array contains total of 10 (N = 10) elements. The range of elements is stated to be 1-10. The unique elements in the array are: 1, 3, 4, 6 and 8 ELements 1, 4 and 6 are repeated. The missing elements in the range 1-10 are : 2 5 7 9 10 Therefore output = 2 5 7 9 10

10 : 5 5
11 : 0 5

Submitted Code

Language: Java 8

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

public class Solution {

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        int l= sc.nextInt();
        int r=sc.nextInt();

        Set<Integer> st = new HashSet<>();

        for(int i=0;i<n;i++)st.add(sc.nextInt());
        boolean flag = true;
        for(int i=l;i<=r;i++){
            if(!st.contains(i)){
                flag=false;
                System.out.print(i+" ");
            }
        }

        if(flag)System.out.print(-1);

    }
}
```