Hashmap

K, V

K	V
Delhi	120
Mumbai	240
Goa	360

- put → insert
  → update
- remove(key)
- size()
- get(key) → value
- getOrDefault(key, 0)
- containsKey(key) → T / F

Character and it's Frequency

Sample Input 0

```
6
a b a d b c
```

Sample Output 0

```
a 2
b 2
c 1
d 1
```

Sample Input 1

```
7
* / - + * . +
```

Sample Output 1

```
* 2
+ 2
- 1
. 1
/ 1
```

freq map.

a - 2

b - 2

d - 1

c - 1

K V

1. freq map.
2. print in sorted manner (key).

* — 2

/ — 1

\- — 1

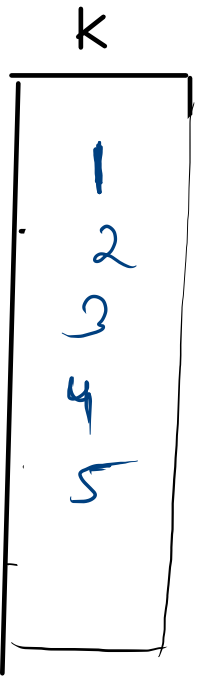
\+ — 2

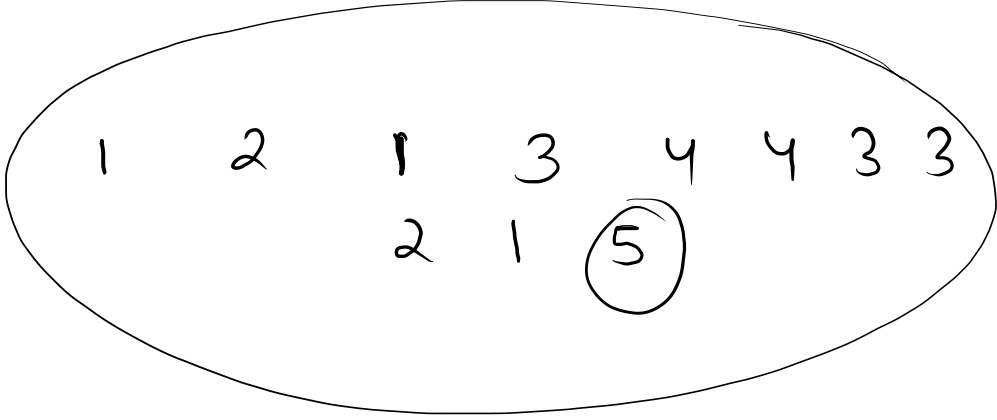
. — 1

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

public class Solution {

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

        int min = Integer.MAX_VALUE;
        int max = Integer.MIN_VALUE;

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

        for(int i = 0; i < n; i++){
            char ch = scn.next().charAt(0);
            min = Math.min(min, ch);
            max = Math.max(max, ch);


            hm.put(ch, hm.getOrDefault(ch, 0)+ 1);

        }

        //print
        while(min <= max){
            char key = (char)(min);

            if(hm.containsKey(key)){
                System.out.println(key + " " + hm.get(key));
            }
            min++;
        }
    }
}
```

Hash Set.

↳ unique values





Que. Arr → unique no.

↓

HashSet.

5

Hash Set.

```java
import java.util.HashSet;
public class Main
{
    public static void main(String[] args) {
        HashSet<Integer> hs = new HashSet<>();

        hs.add(10);
        hs.add(20);
        hs.add(30);
        hs.add(10);

        hs.remove(30);
        System.out.println(hs.size());

        if(hs.contains(30)){
            System.out.println("Present");
        }
    }
}

```

Two Sum 14

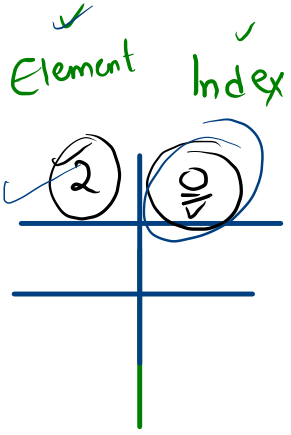
Sample Input 0

```
4 9
2 7 11 15
```

Sample Output 0

```
0 1
```

0 1 stop.



n = 4

tar = 9

2 7 11 15

0 1 2 3

rem = tar - ek

= 9 - 7

= 2

```java
public class Solution {

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

        //ele vs index
        HashMap<Integer, Integer> hm = new HashMap<>();

        for(int i = 0; i < n; i++){
            int key = scn.nextInt();

            int rem = tar-key;

            if(hm.containsKey(rem)){
                System.out.println(hm.get(rem) + " " + i);
                break;
            }

            hm.put(key, i);

        }


    }
}
```

tar = 9

n = 4

2 2 11 7

0 1 2 3

1.
15

B

i = 3

k = 7

rem = 9 - 7 = 2

0 2

1 3

LC - 1

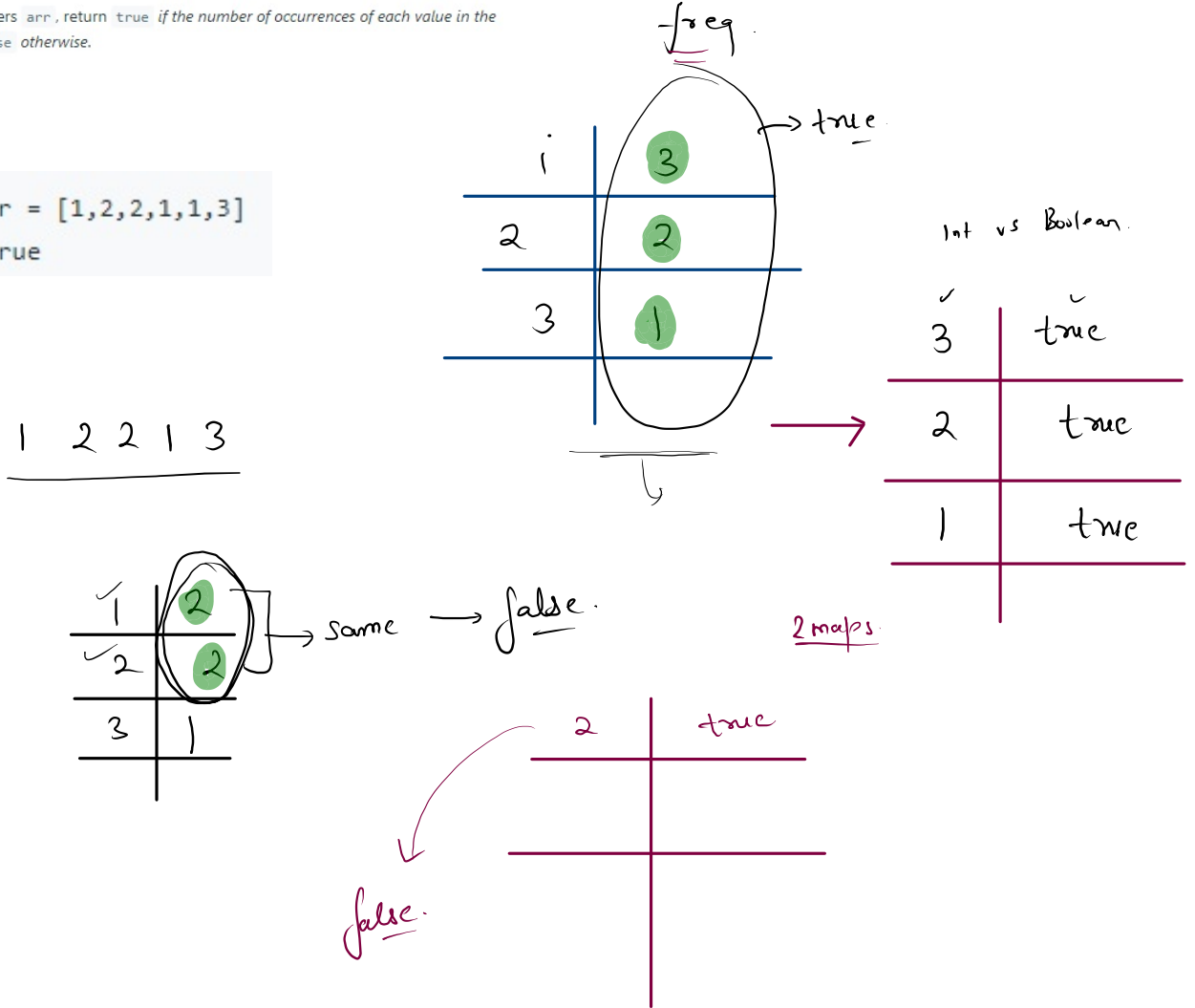
2	0
11	1
15	2

1207. Unique Number of Occurrences

Easy 3810 88 Add to List Share

Given an array of integers `arr`, return `true` *if the number of occurrences of each value in the array is* ***unique*** *or* `false` *otherwise.*

```
Input: arr = [1,2,2,1,1,3]
Output: true
```



Steps:

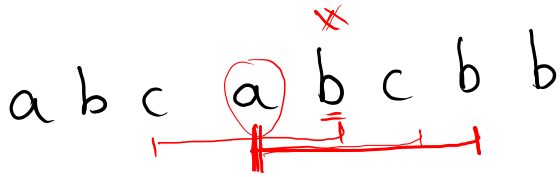
1. → create freq map.

2. → create freq v/s boolean
↳ and check.

```java
class Solution {
    public boolean uniqueOccurrences(int[] arr) {
        //step 1
       HashMap<Integer, Integer> freq = new HashMap<>();
        for(int key : arr){
            freq.put(key, freq.getOrDefault(key, 0) + 1);
        }

        //freq val vs true /false
        HashMap<Integer, Boolean> hm = new HashMap<>();

        for(int key : freq.keySet()){
            int val = freq.get(key);
            if(hm.containsKey(val)){
                return false;
            }
            hm.put(val, true);
        }

        return true;
    }
}
```

Longest Substring Without Repeating Characters 6

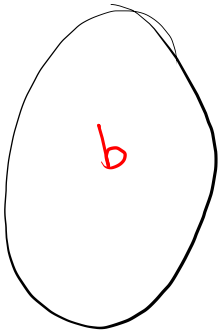
```
Input: s = "abcabcbb"
Output: 3
```



ans = 1 ≠ 3

i

hs.



j

"a

---

g will acquire with i & release with j

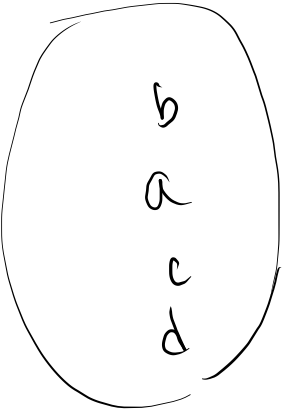
ans = 4.

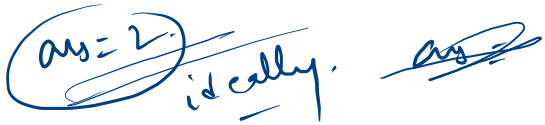
b a c a b c d

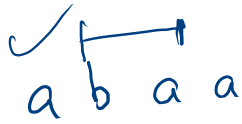
i

j

ans = ~~0~~ ~~1~~ ~~2~~ ~~3~~ 4








```java
class Solution {
    public int lengthOfLongestSubstring(String s) {
        HashSet<Character> hs = new HashSet<>();
        int ans = 0;
        int i = 0, j = 0;
        //i to add
        //j to remove
        while(i < s.length()){
            char ch = s.charAt(i);
            if(hs.contains(ch)){
                //remove
                hs.remove(s.charAt(j));
                j++;
            }
            else{
                //add
                hs.add(ch);
                i++;
            }

            ans = Math.max(ans, hs.size());
        }
        return ans;
    }
}
```