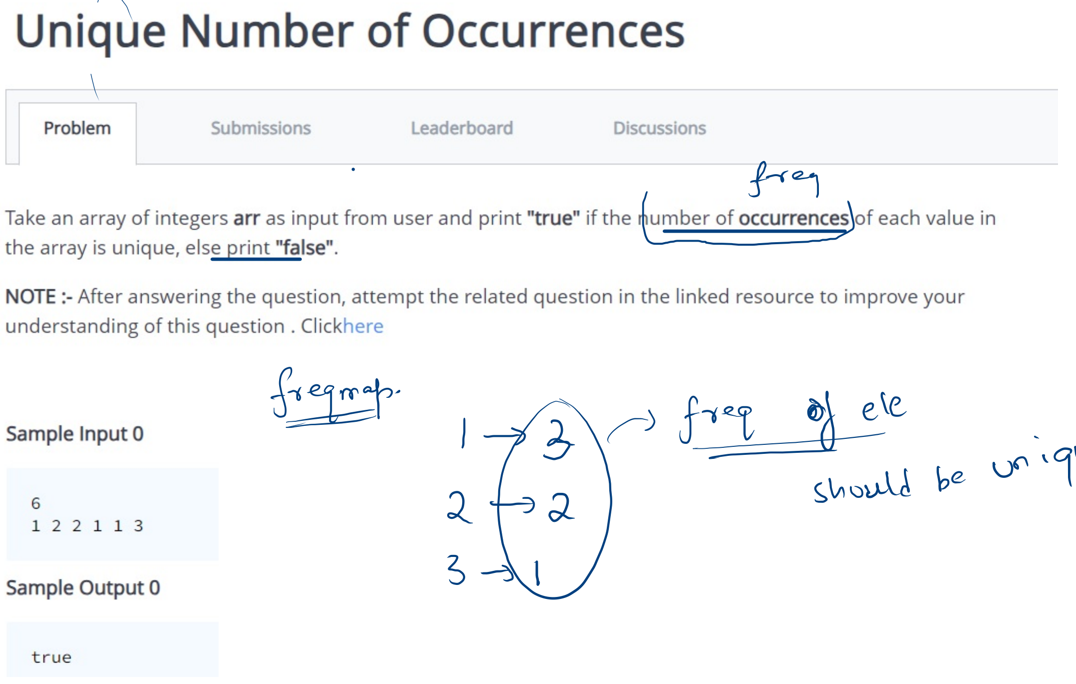


Unique Number of Occurrences

freq

Take an array of integers **arr** as input from user and print **"true"** if the number of **occurrences** of each value in the array is unique, else print **"false"**.

NOTE :- After answering the question, attempt the related question in the linked resource to improve your understanding of this question . Clickhere

Sample Input 0

```
6
1 2 2 1 1 3
```

Sample Output 0

```
true
```

freqmap.

$$\begin{tikzcd}
1 \arrow[r] & 3 \\
2 \arrow[r] & 2 \\
3 \arrow[r] & 1
\end{tikzcd}$$

freq of ele

should be unique

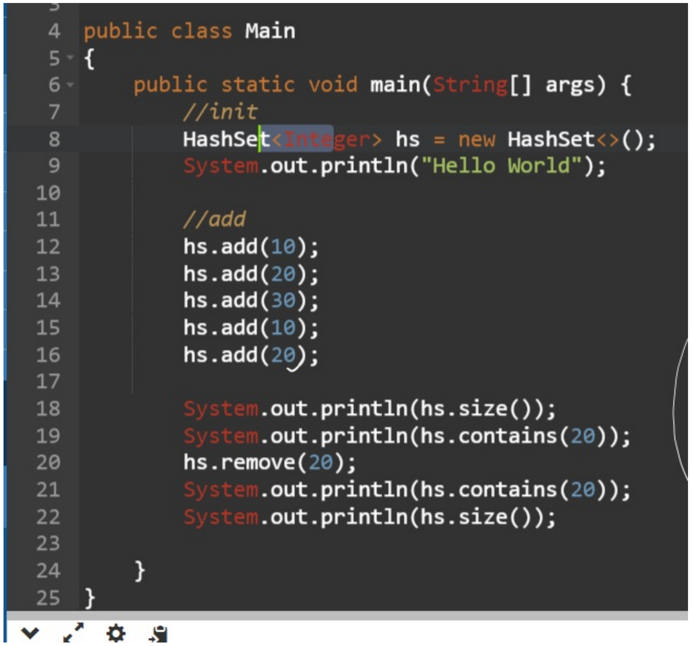
HashSet $\rightarrow$ holding keys only.

↳ every element will be unique.

unique.

↳ Hashset/ HashMap.

- add
- remove
- contains
- size



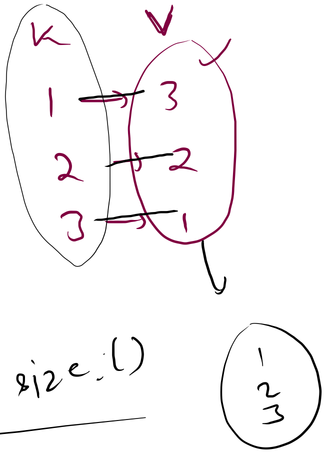
```java
public class Main
{
    public static void main(String[] args) {
        //init
        HashSet<Integer> hs = new HashSet<>();
        System.out.println("Hello World");

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

        System.out.println(hs.size());
        System.out.println(hs.contains(20));
        hs.remove(20);
        System.out.println(hs.contains(20));
        System.out.println(hs.size());

    }
}
```

n= 6

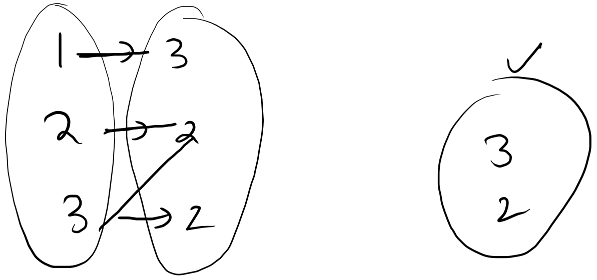
1 2 2 1 1 3

unique.

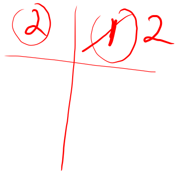
freqmap



hm.size()
== hs.size()

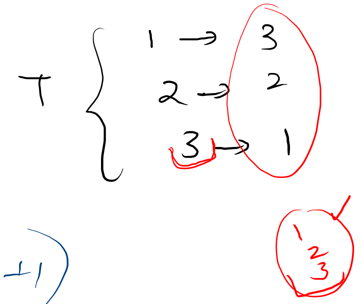


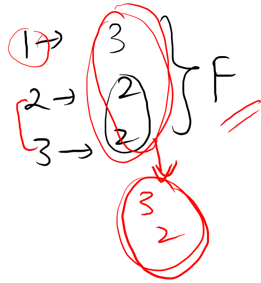
K V
3 — 2



1	2	2	1	3	1
0	1	2	3	4	5

i





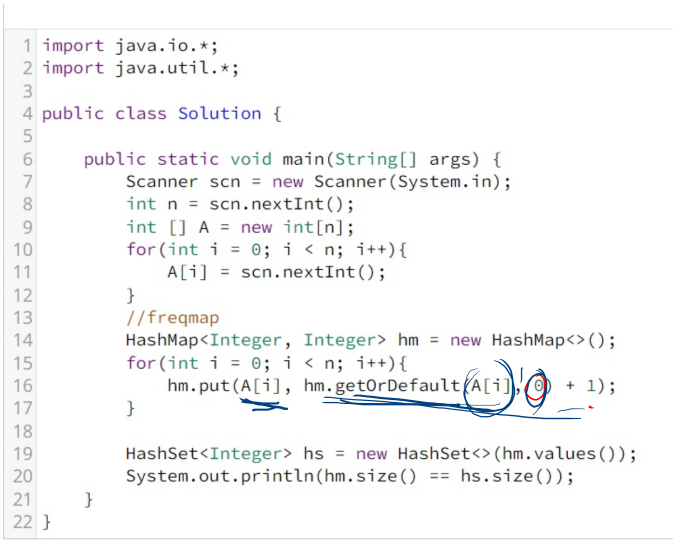
1, 2+1

1	~~1~~ ~~2~~ 3
2	~~1~~ 2
3	1

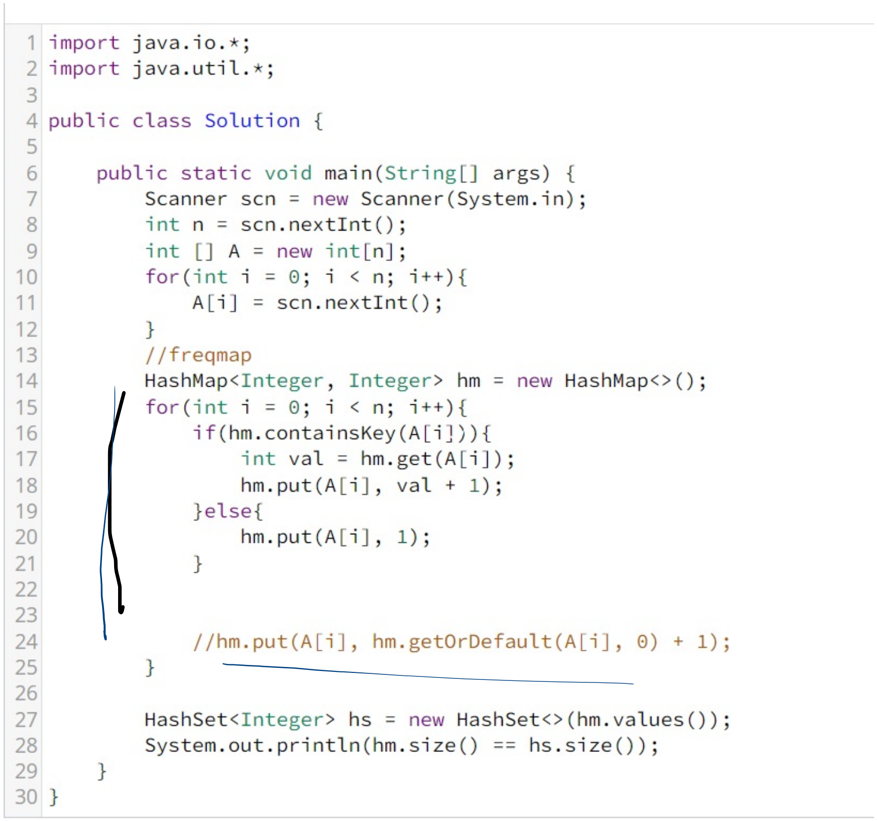
2,

3, 0+1

1, 2



```
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 [] A = new int[n];
        for(int i = 0; i < n; i++){
            A[i] = scn.nextInt();
        }
        //freqmap
        HashMap<Integer, Integer> hm = new HashMap<>();
        for(int i = 0; i < n; i++){
            hm.put(A[i], hm.getOrDefault(A[i], 0) + 1);
        }

        HashSet<Integer> hs = new HashSet<>(hm.values());
        System.out.println(hm.size() == hs.size());
    }
}
```



```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 [] A = new int[n];
        for(int i = 0; i < n; i++){
            A[i] = scn.nextInt();
        }
        //freqmap
        HashMap<Integer, Integer> hm = new HashMap<>();
        for(int i = 0; i < n; i++){
            if(hm.containsKey(A[i])){
                int val = hm.get(A[i]);
                hm.put(A[i], val + 1);
            }else{
                hm.put(A[i], 1);
            }


            //hm.put(A[i], hm.getOrDefault(A[i], 0) + 1);
        }

        HashSet<Integer> hs = new HashSet<>(hm.values());
        System.out.println(hm.size() == hs.size());
    }
}
```

Valid Anagram 5

Problem | Submissions | Leaderboard | Discussions

Given two strings s and t, return true if t is an anagram of s, and false otherwise. An Anagram is a word or phrase formed by rearranging the letters of a different word or phrase, typically using all the original letters exactly once.

Sample Input 0

```
anagram
nagaram
```

Sample Output 0

```
true
```

$s \rightarrow$ anagram
$t \rightarrow$ rnaagam

2 fm.

fms

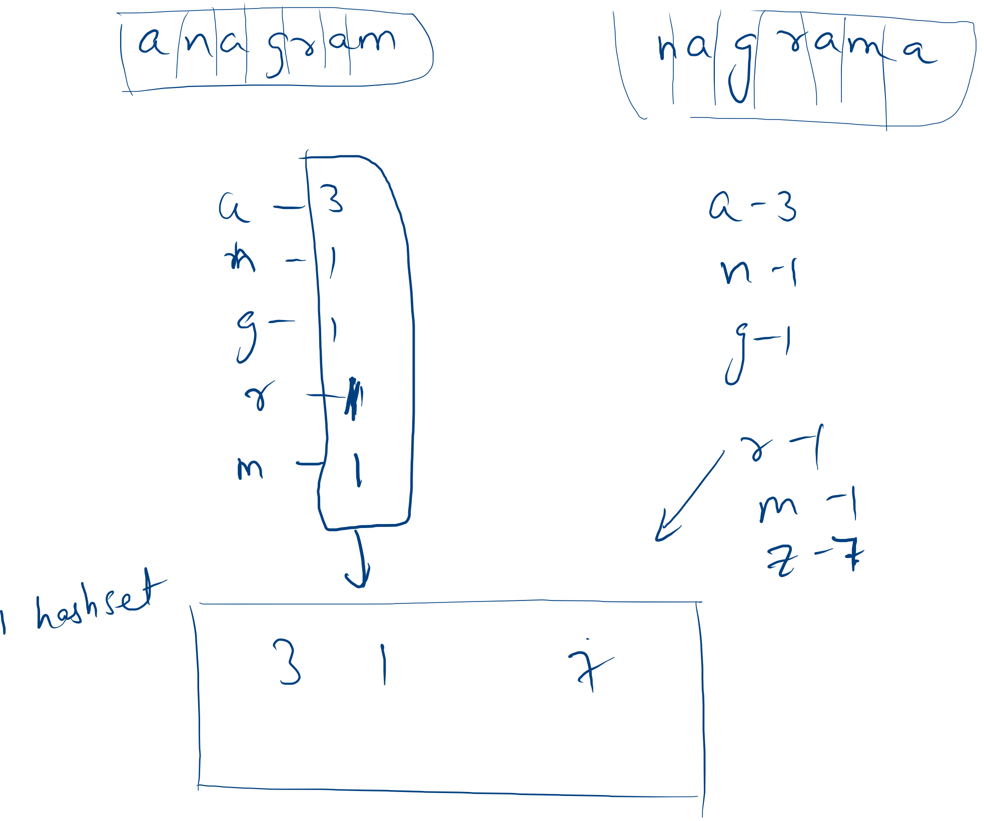
a - 3
n - 1
g - 1
r - 1
m - 1

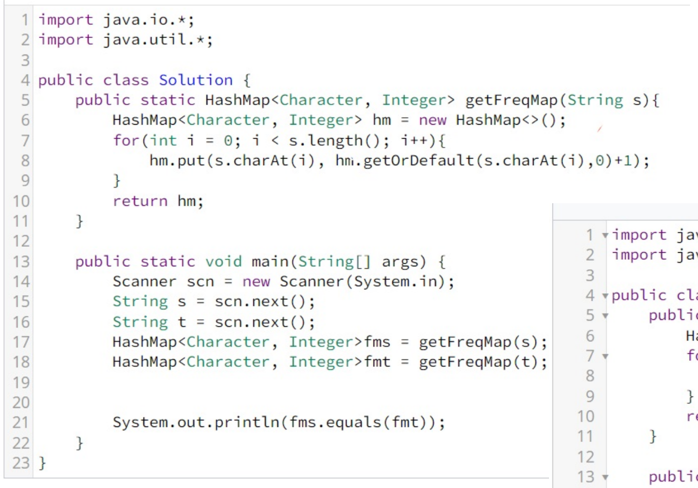
fmt

r - 1
n - 1
a - 3
g - 1
m - 1

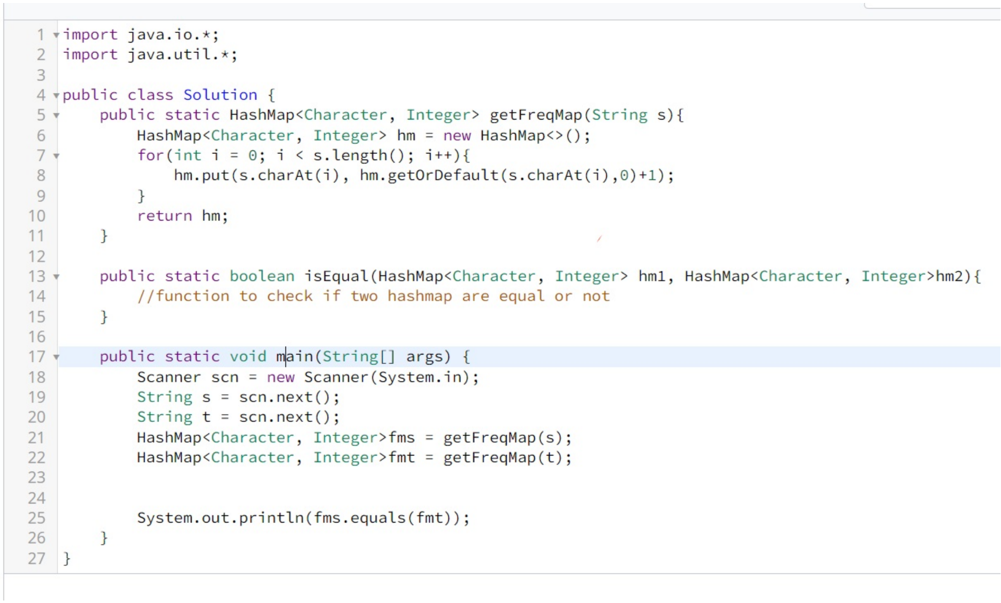
fms .equals (fmt)

false ↙ ↳ true


anagram
nagrama
a — 3
n — 1
g — 1
r — 1
m — 1
a - 3
n - 1
g - 1
r - 1
m - 1
z - 7
hashset
3 1 7



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

public class Solution {
    public static HashMap<Character, Integer> getFreqMap(String s){
        HashMap<Character, Integer> hm = new HashMap<>();
        for(int i = 0; i < s.length(); i++){
            hm.put(s.charAt(i), hm.getOrDefault(s.charAt(i),0)+1);
        }
        return hm;
    }

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        String s = scn.next();
        String t = scn.next();
        HashMap<Character, Integer>fms = getFreqMap(s);
        HashMap<Character, Integer>fmt = getFreqMap(t);


        System.out.println(fms.equals(fmt));
    }
}
```



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

public class Solution {
    public static HashMap<Character, Integer> getFreqMap(String s){
        HashMap<Character, Integer> hm = new HashMap<>();
        for(int i = 0; i < s.length(); i++){
            hm.put(s.charAt(i), hm.getOrDefault(s.charAt(i),0)+1);
        }
        return hm;
    }

    public static boolean isEqual(HashMap<Character, Integer> hm1, HashMap<Character, Integer>hm2){
        //function to check if two hashmap are equal or not
    }

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        String s = scn.next();
        String t = scn.next();
        HashMap<Character, Integer>fms = getFreqMap(s);
        HashMap<Character, Integer>fmt = getFreqMap(t);


        System.out.println(fms.equals(fmt));
    }
}
```

Longest Substring Without Repeating Characters 6

You are given a **string**, print the length of **Longest Substring** Without Repeating Characters.

Sample Input 0

```
abcabcbb
```

Sample Output 0

```
3
```

max (ans, hs.size())

a	b	c	a	b	c	b	b
0	1	2	3	4	5	6	7

j → acquire

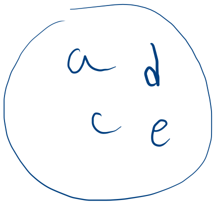
i → release

eg.

a	b	a	a	d	c	e	a
0	1	2	3	4	5	6	7

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

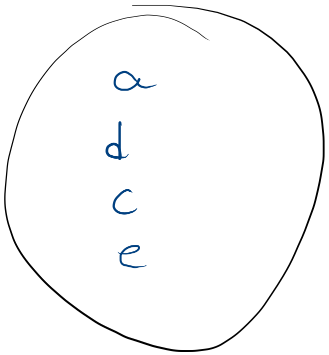
i   j

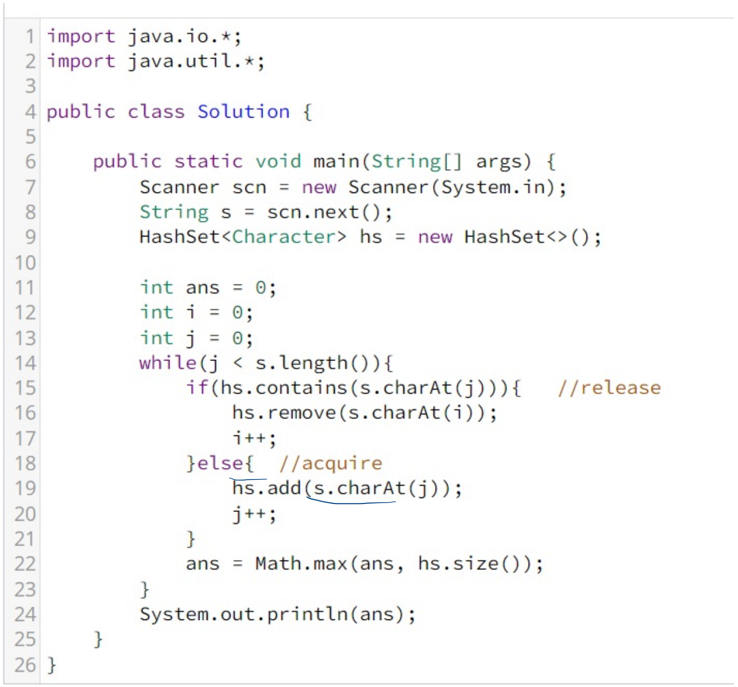


8 → a b $\overset{\checkmark}{a}$ a d c e a

0 1 2 3 4 5 6 7

i j

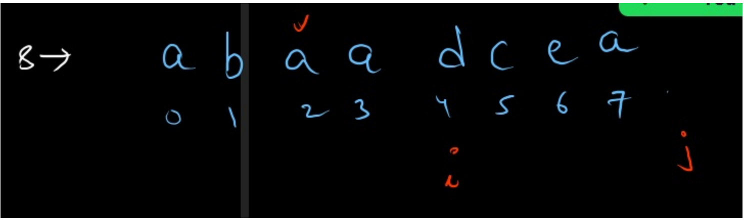




```java
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        String s = scn.next();
        HashSet<Character> hs = new HashSet<>();

        int ans = 0;
        int i = 0;
        int j = 0;
        while(j < s.length()){
            if(hs.contains(s.charAt(j))){    //release
                hs.remove(s.charAt(i));
                i++;
            }else{  //acquire
                hs.add(s.charAt(j));
                j++;
            }
            ans = Math.max(ans, hs.size());
        }
        System.out.println(ans);
    }
}
```

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

max(ans, hs.size())



a - 3

b - 1

d - 1

c - 1

e - 1

✓ abcde

cdabe

eabdc