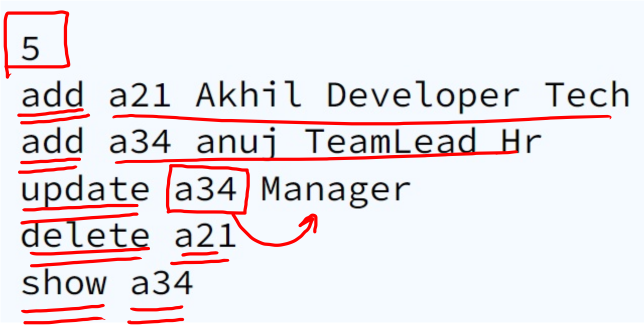
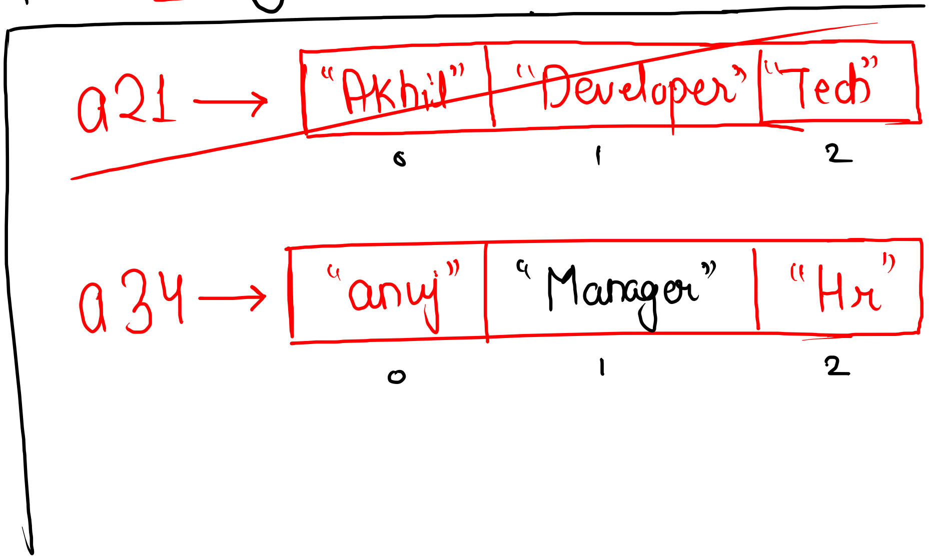
employee management



```
5
add a21 Akhil Developer Tech
add a34 anuj TeamLead Hr
update a34 Manager
delete a21
show a34
```

0 → name
1 → job title
2 → department

map String vs Arraylist<String>

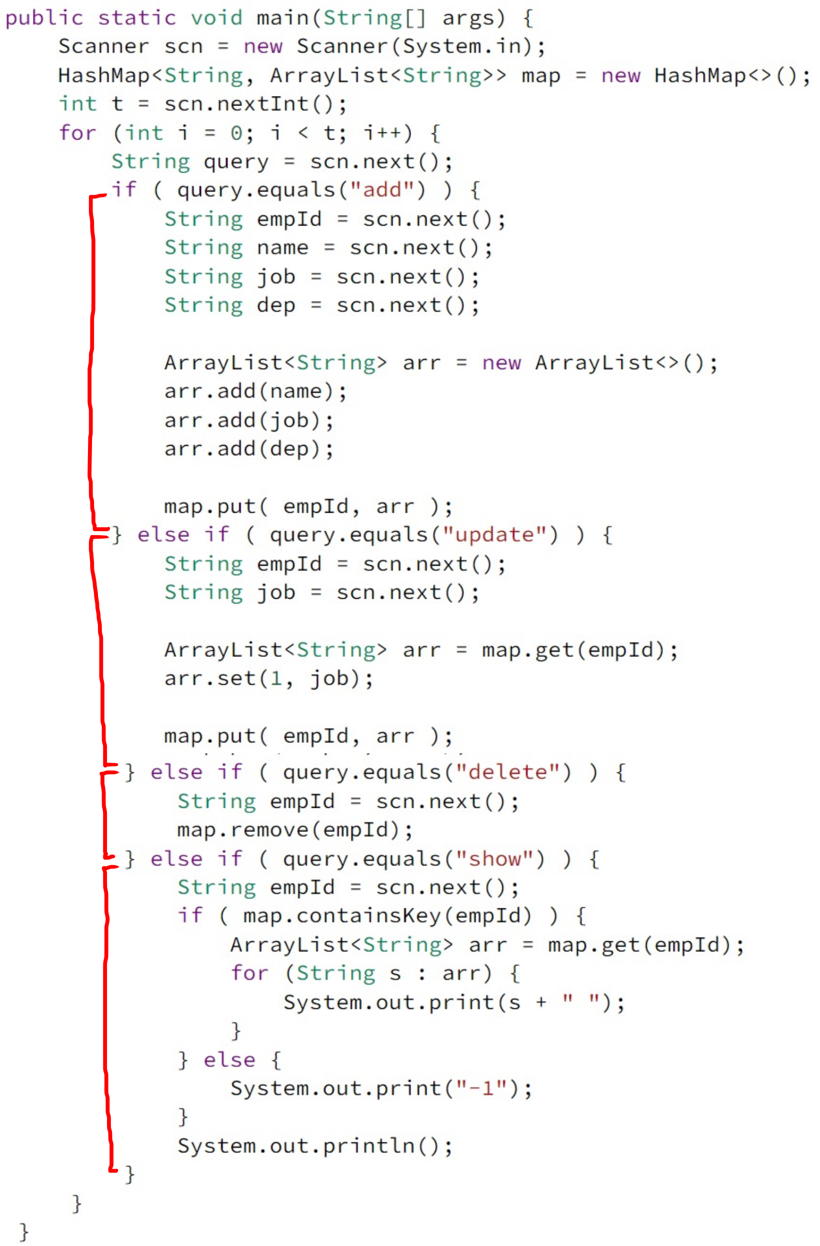


~~a21 → "Akhil" "Developer" "Tech"~~
0 1 2

a34 → "anuj" "Manager" "Hr"
0 1 2

empId = "a34"
name = "anuj"
job = "TeamLead"
dep = "Hr"

code



```
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    HashMap<String, ArrayList<String>> map = new HashMap<>();
    int t = scn.nextInt();
    for (int i = 0; i < t; i++) {
        String query = scn.next();
        if ( query.equals("add") ) {
            String empId = scn.next();
            String name = scn.next();
            String job = scn.next();
            String dep = scn.next();

            ArrayList<String> arr = new ArrayList<>();
            arr.add(name);
            arr.add(job);
            arr.add(dep);

            map.put( empId, arr );
        } else if ( query.equals("update") ) {
            String empId = scn.next();
            String job = scn.next();

            ArrayList<String> arr = map.get(empId);
            arr.set(1, job);

            map.put( empId, arr );
         } else if ( query.equals("delete") ) {
             String empId = scn.next();
             map.remove(empId);
         } else if ( query.equals("show") ) {
             String empId = scn.next();
             if ( map.containsKey(empId) ) {
                 ArrayList<String> arr = map.get(empId);
                 for (String s : arr) {
                     System.out.print(s + " ");
                 }
             } else {
                 System.out.print("-1");
             }
             System.out.println();
         }
     }
 }
```

⇒ Variation of hashmap

→ HashSet (repeated values are not allowed in hashset)

set

3
1
5
7
10

```
HashSet<Integer> set = new HashSet<>();
set.add(value);
set.remove(value);
set.contains(value);
set.size(); / set.isEmpty();
```

Note:- hashset is used to identify the duplicacy

Imp

Unique Number of Occurrences

arr = [3 , 5 , 5 , 7 , 3 , 3 , 3]

↑

map

3 → 4

5 → 2

7 → 1

size = 3

= =

set

4

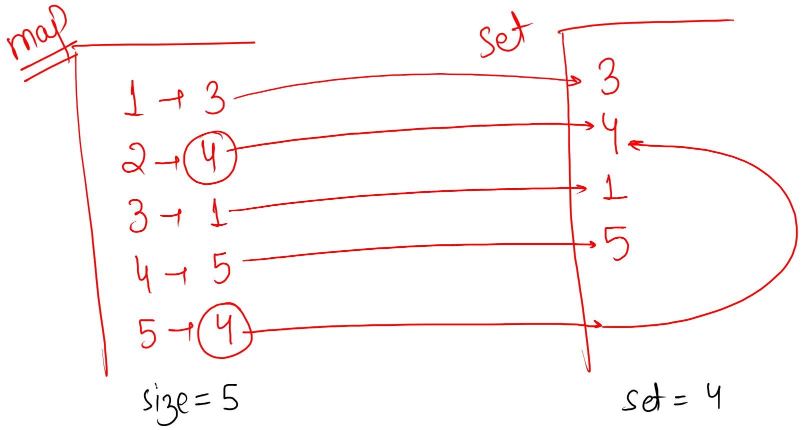
2

1

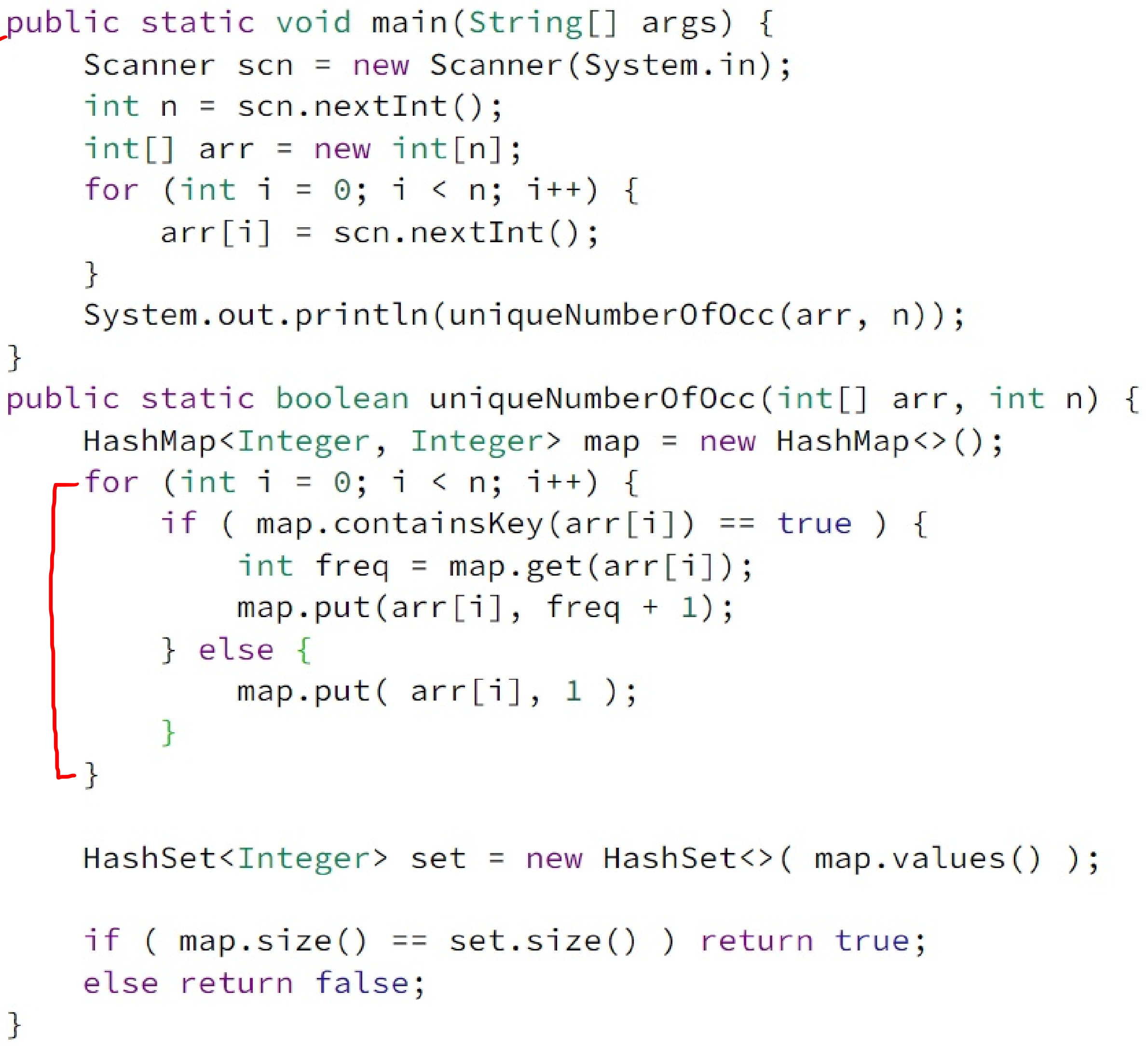
size = 3

map.values()

ex :-



code



```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int[] arr = new int[n];
    for (int i = 0; i < n; i++) {
        arr[i] = scn.nextInt();
    }
    System.out.println(uniqueNumberOfOcc(arr, n));
}
public static boolean uniqueNumberOfOcc(int[] arr, int n) {
    HashMap<Integer, Integer> map = new HashMap<>();
    for (int i = 0; i < n; i++) {
        if ( map.containsKey(arr[i]) == true ) {
            int freq = map.get(arr[i]);
            map.put(arr[i], freq + 1);
        } else {
            map.put( arr[i], 1 );
        }
    }

    HashSet<Integer> set = new HashSet<>( map.values() );

    if ( map.size() == set.size() ) return true;
    else return false;
}
```

Two Sum 14

n = 4 , target = 9

ans = 0 , 1

arr = [2 , 7 , 11 , 15]

$$\begin{array}{cccc} 2, & 7, & 11, & 15 \\ 0 & 1 & 2 & 3 \\ & \uparrow & & \\ & i & & \end{array}$$

num1 + num2 = target

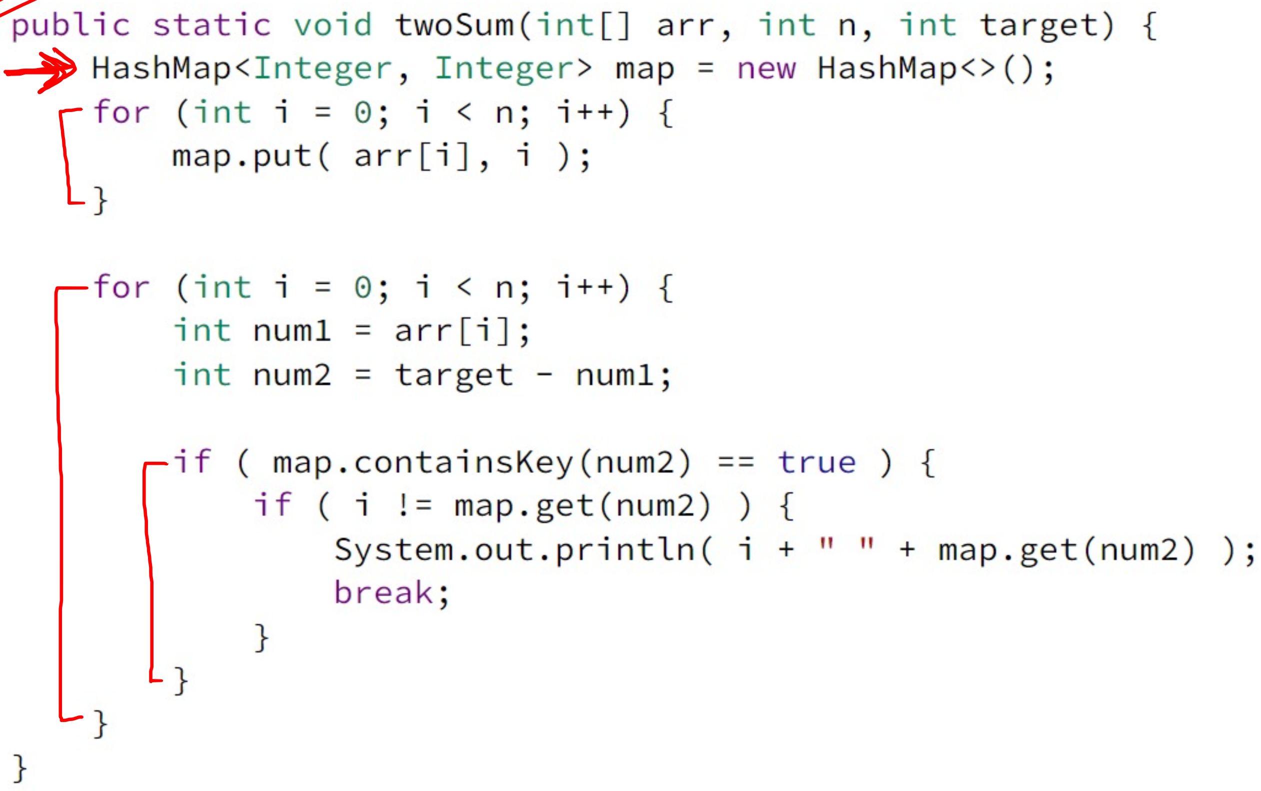
num2 = target - num1

map

Integer	→	Integer
2	→	0
7	→	1
11	→	2
15	→	3

$\boxed{\text{find num2} = 7}$

Code



```java
public static void twoSum(int[] arr, int n, int target) {
    HashMap<Integer, Integer> map = new HashMap<>();
    for (int i = 0; i < n; i++) {
        map.put( arr[i], i );
    }

    for (int i = 0; i < n; i++) {
        int num1 = arr[i];
        int num2 = target - num1;

        if ( map.containsKey(num2) == true ) {
            if ( i != map.get(num2) ) {
                System.out.println( i + " " + map.get(num2) );
                break;
            }
        }
    }
}
```

$T.C = O(n)$, $S.C = O(n)$

dry run

$$arr = \begin{array}{cccc} \downarrow & & & \\ [\,2, & 7, & 11, & 15\,] \\ 0 & 1 & 2 & 3 \end{array}$$

map

2 → 0

7 → 1

11 → 2

15 → 3

num1 = 2

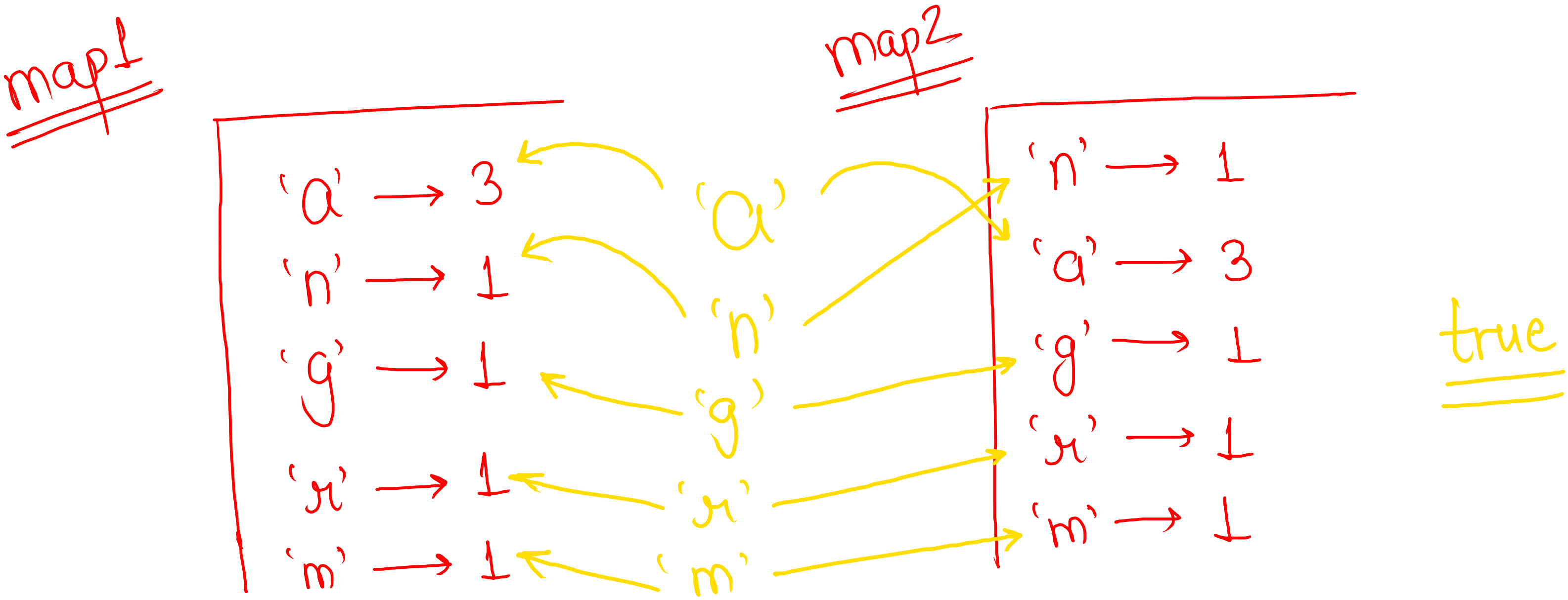
num2 = 7

Valid Anagram 5

(each char should be having same freq)

s = "anagram"

t = "nagaram"



map1

'a' → 3
'n' → 1
'g' → 1
'r' → 1
'm' → 1

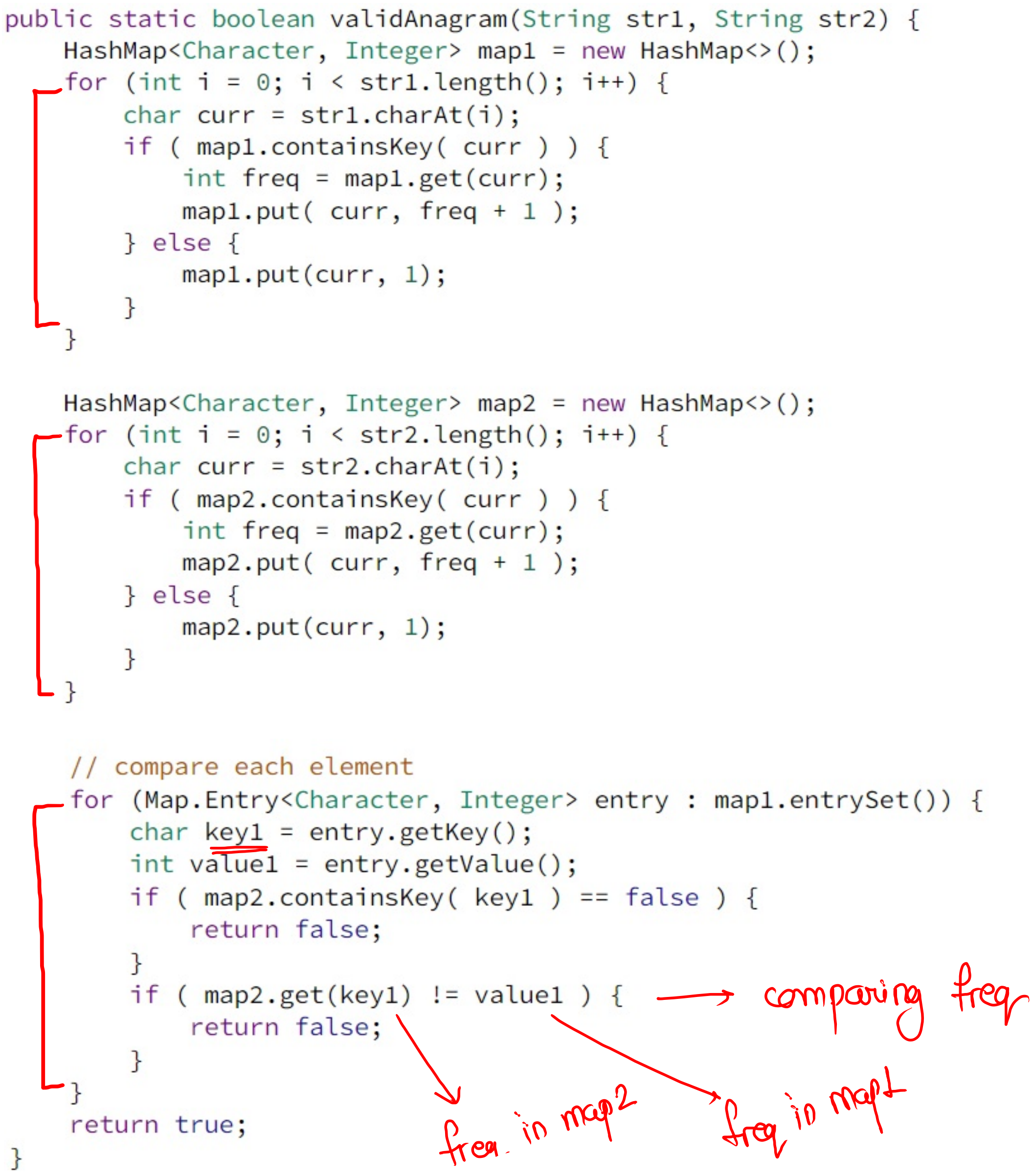
map2

'n' → 1
'a' → 3
'g' → 1
'r' → 1
'm' → 1

true

Note :- check each char freq from map1 to map2

code



```java
public static boolean validAnagram(String str1, String str2) {
    HashMap<Character, Integer> map1 = new HashMap<>();
    for (int i = 0; i < str1.length(); i++) {
        char curr = str1.charAt(i);
        if ( map1.containsKey( curr ) ) {
            int freq = map1.get(curr);
            map1.put( curr, freq + 1 );
        } else {
            map1.put(curr, 1);
        }
    }

    HashMap<Character, Integer> map2 = new HashMap<>();
    for (int i = 0; i < str2.length(); i++) {
        char curr = str2.charAt(i);
        if ( map2.containsKey( curr ) ) {
            int freq = map2.get(curr);
            map2.put( curr, freq + 1 );
        } else {
            map2.put(curr, 1);
        }
    }

    // compare each element
    for (Map.Entry<Character, Integer> entry : map1.entrySet()) {
        char key1 = entry.getKey();
        int value1 = entry.getValue();
        if ( map2.containsKey( key1 ) == false ) {
            return false;
        }
        if ( map2.get(key1) != value1 ) {
            return false;
        }
    }
    return true;
}
```

n

n

n

T.C = O(n)

S.C = O(n)

comparing freq

freq. in map2

freq in map1