1497. Check If Array Pairs Are Divisible by k

Medium 1650 89 Add to List Share

```
Input: arr = [1,2,3,4,5,10,6,7,8,9], k = 5
Output: true
```

mod e%k

n = 10^5

mod	freq
1	✗ 2
2	✗ 2
3	✗ 2
4	✗ 2
0	✗ 2

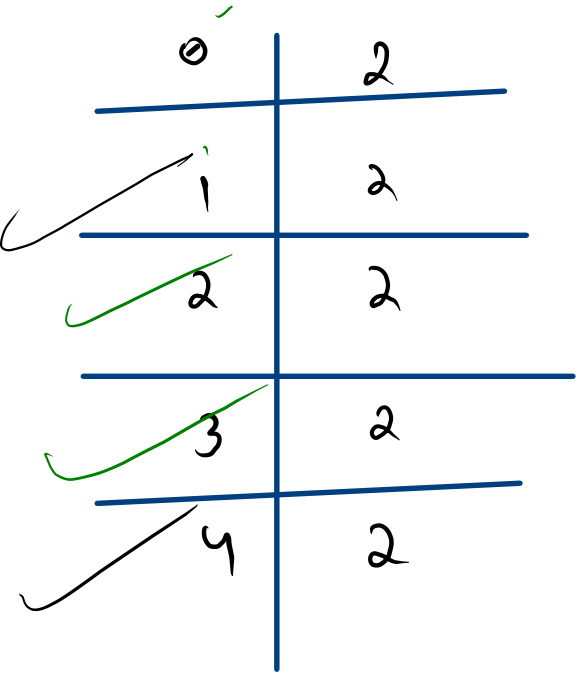
-2 + 7 = 5

-2 % 5 = -2 + k = 3 + 2 = -5 7 % 5 = 2

3

-e % k + k

$k = 5$



$e^{0} \% k == 1$

$e^{1} \% k == 4$

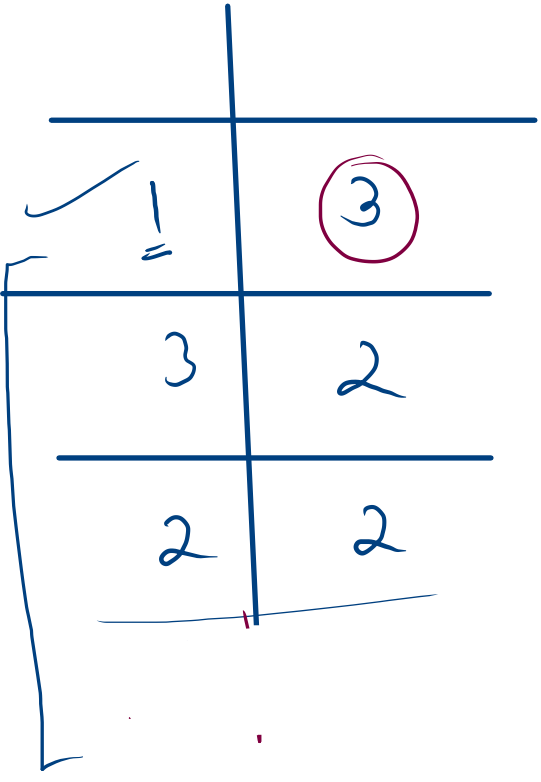
m m

+ +

4 4

m m

k = 5



1 → 4

val 1 = 3

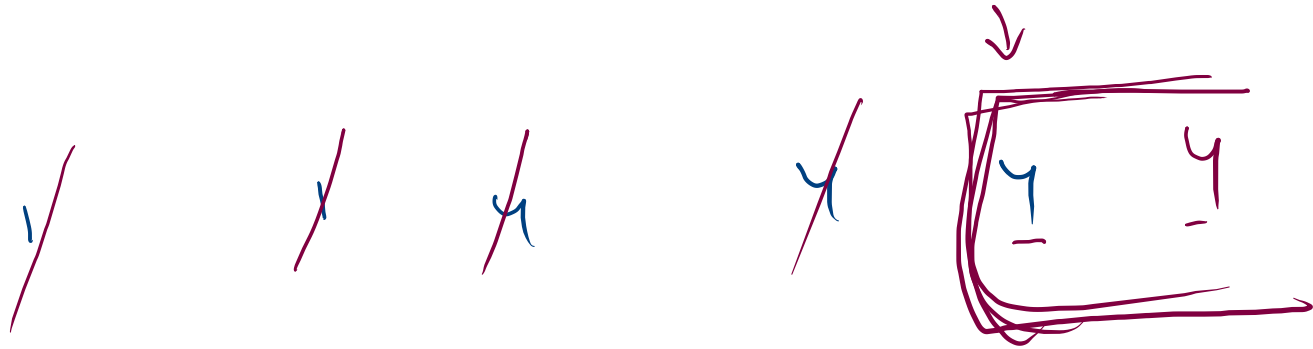
val 2 = 0 null.

3 != 0

false.

4 missing.

hm.get(4) = null

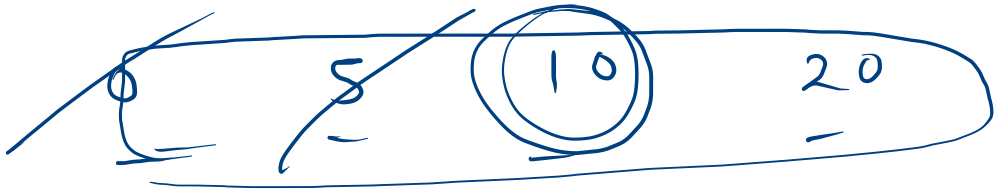


false.

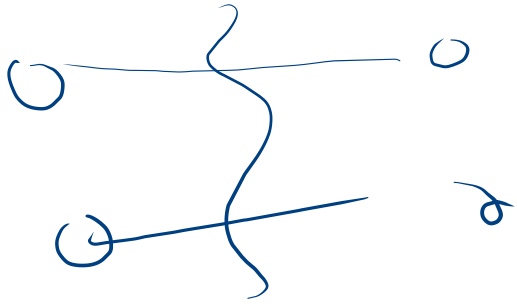
1	2
4	4

ans → false.

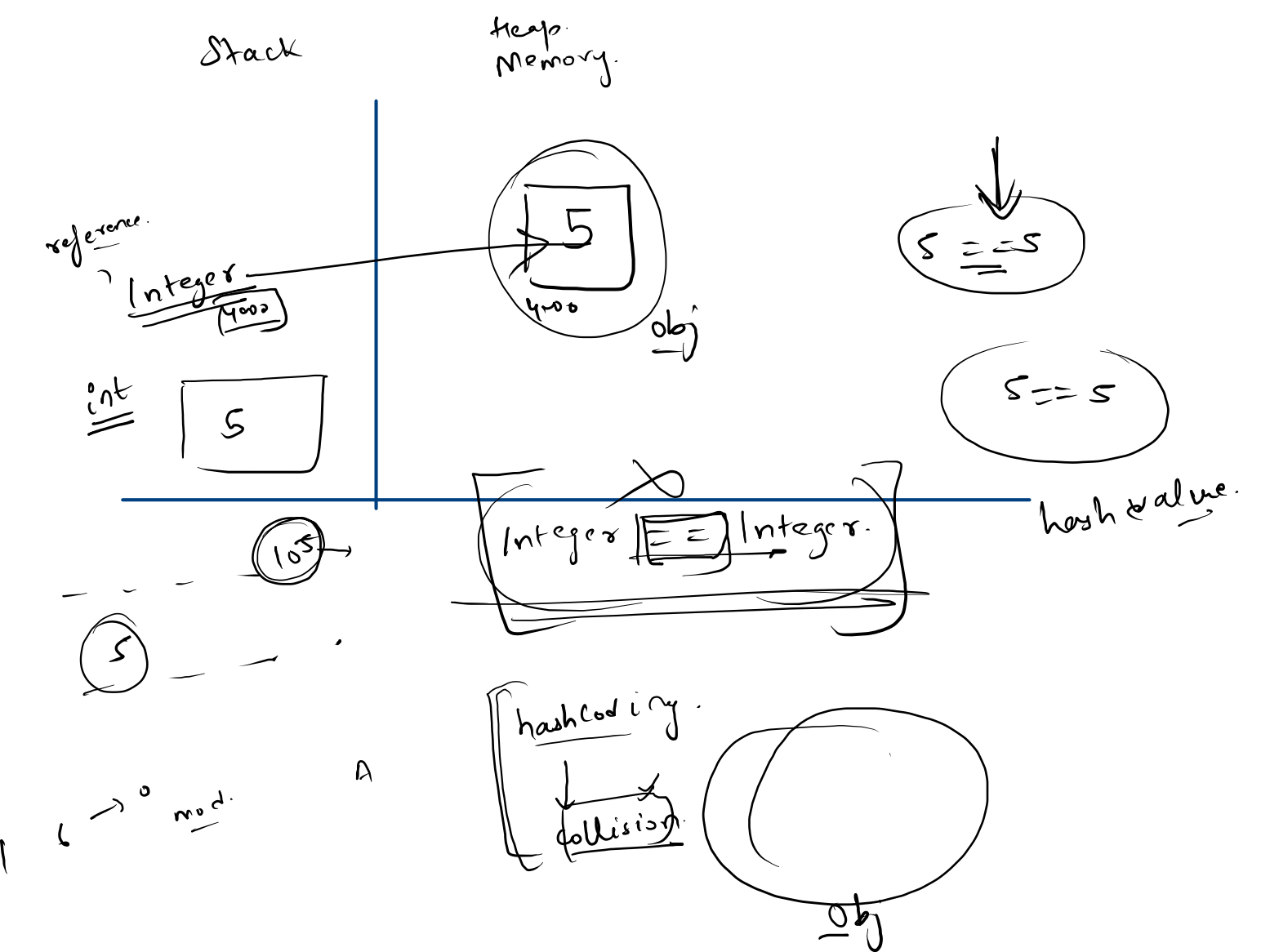
K = 5



0% K	5% K	10%
0	0	0



```java
class Solution {
    public boolean canArrange(int[] arr, int k) {
        HashMap<Integer, Integer> hm = new HashMap<>();
        for(int e : arr){
            int mod = e%k;
            if(mod < 0)
                mod += k;
            hm.put(mod, hm.getOrDefault(mod, 0)+1);
        }
        //zero
        if(hm.containsKey(0) && hm.get(0)% 2 != 0){
            return false;
        }
        for(int i = 1; i < k; i++){
            int val1 = hm.getOrDefault(i, 0);
            int val2 = hm.getOrDefault(k-i, 0);
            if(val1 != val2){
                return false;
            }
        }

        return true;
    }
}
```


Stack
Heap Memory
reference
Integer
4000
5
4000
obj
int
5
5 == 5
5 == 5
hash value
Integer == Integer
105
5
hashCoding
collision
obj

Longest Substring Without Repeating Characters 6

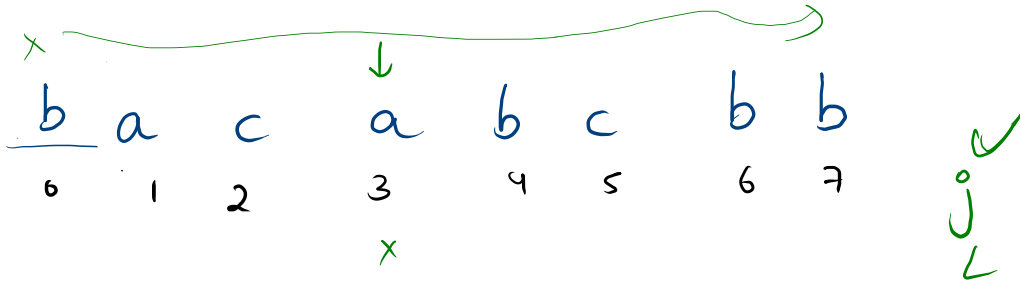
You are given a **string**, print the length of **Longest Substring** Without Repeating Characters.

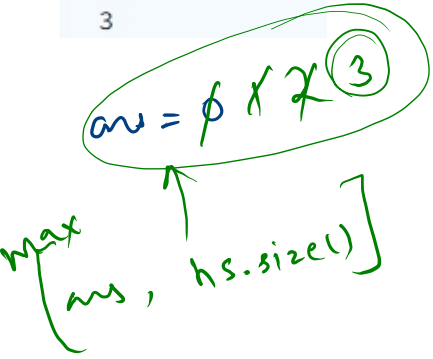
Sample Input 0

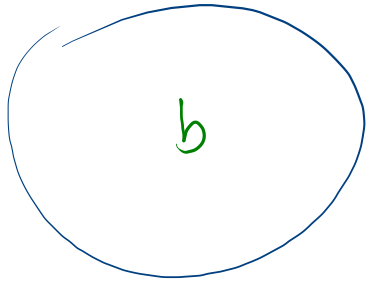
```
abcabcbb
```

Sample Output 0

```
3
```







```java
    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        String s = scn.next();

        HashSet<Character> hs = new HashSet<>();
        int i = 0;
        int j = 0;
        int ans = 0;

        while(j < s.length()){
            char ch = s.charAt(j);
            if(hs.contains(ch)){
                //remove
                hs.remove(s.charAt(i));
                i++;
            }
            else{
                //include
                hs.add(s.charAt(j));
                j++;
            }
            ans = Math.max(ans, hs.size());

        }
        System.out.println(ans);
    }
}
```

Ans = 0 ✗ 3

s →

b a c a b

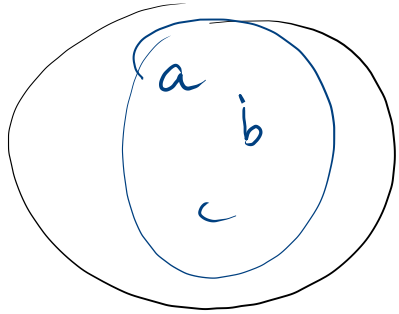
0 1 2 3 4

i

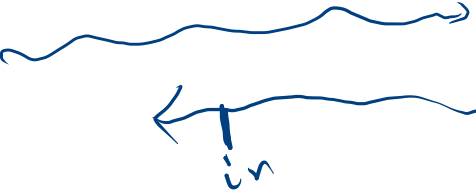
j

③

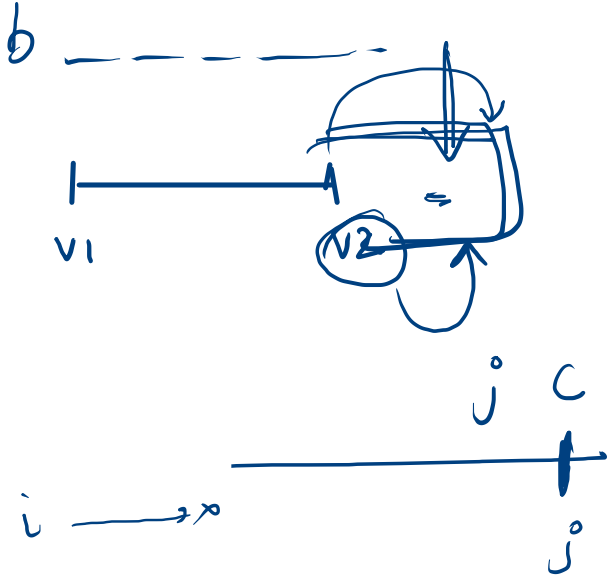
```java
class Solution {
    public int lengthOfLongestSubstring(String s) {
        HashSet<Character> hs = new HashSet<>();
        int i = 0;
        int j = 0;
        int ans = 0;

        while(j < s.length()){
            char ch = s.charAt(j);
            if(hs.contains(ch)){
                //remove
                hs.remove(s.charAt(i));
                i++;
            }
            else{
                //include
                hs.add(s.charAt(j));
                j++;
            }
            ans = Math.max(ans, hs.size());

        }
        return ans;
    }
}
```

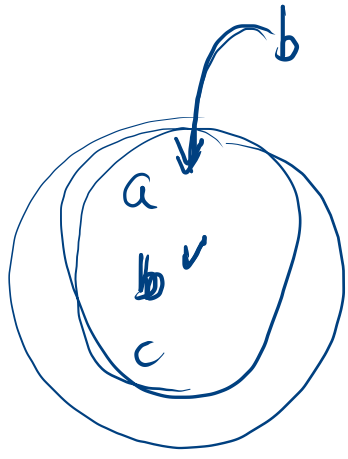


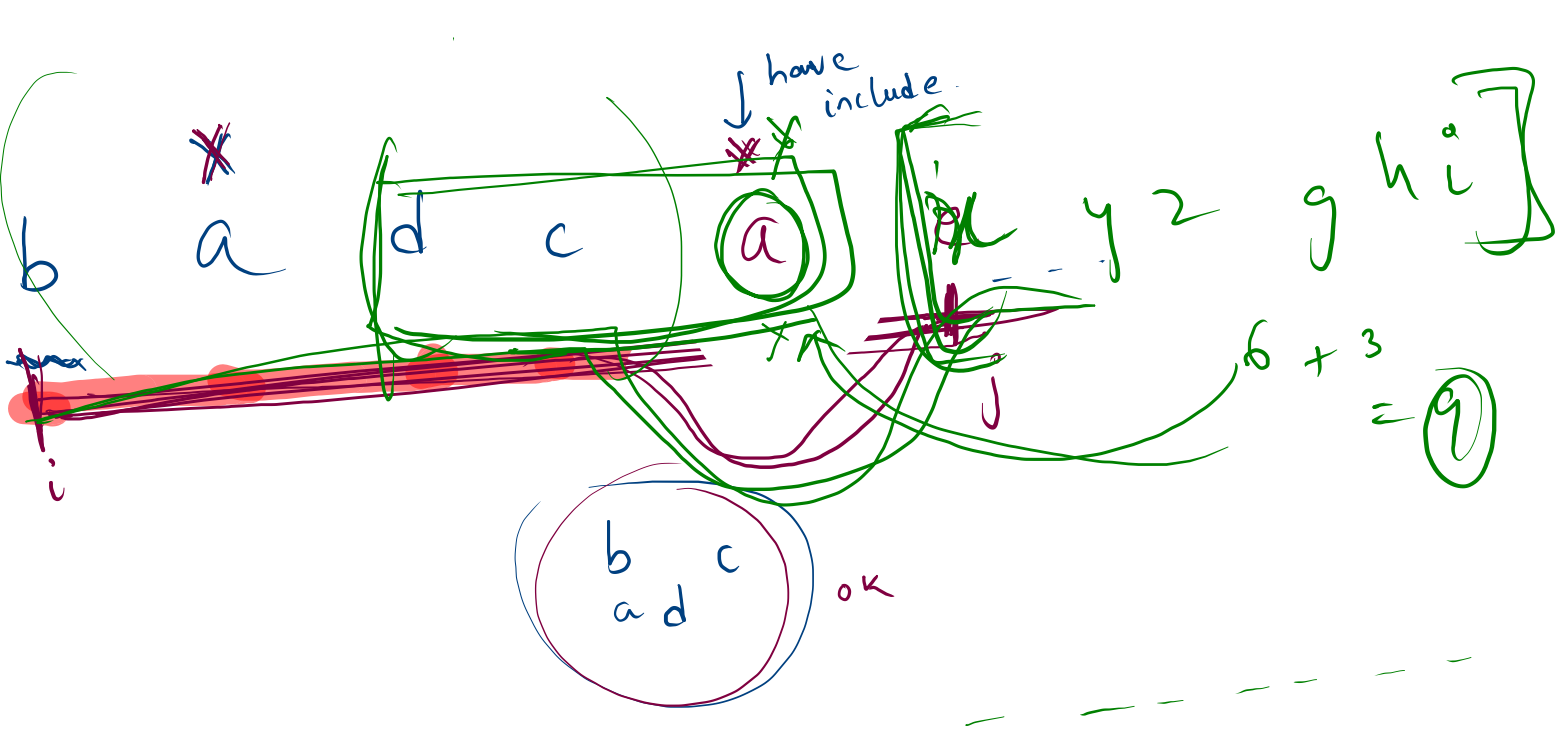
abc



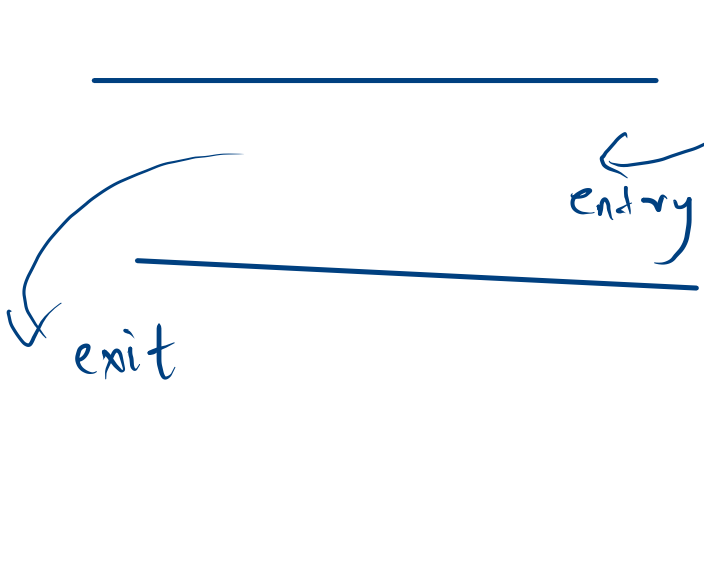
abcb



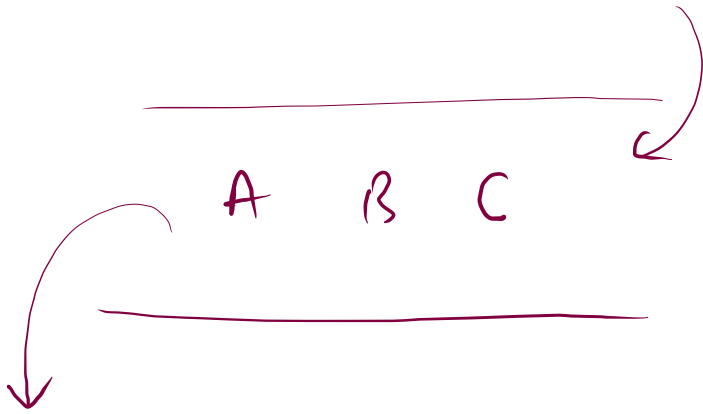



have
include
b
a
d
c
a
y
z
g
h
i
6 + 3
= 9
i
b c
a d
ok

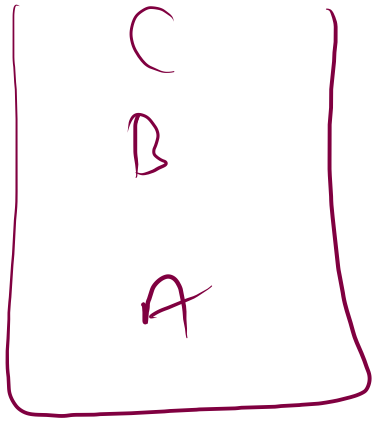
Queue. ⟶ FIFO

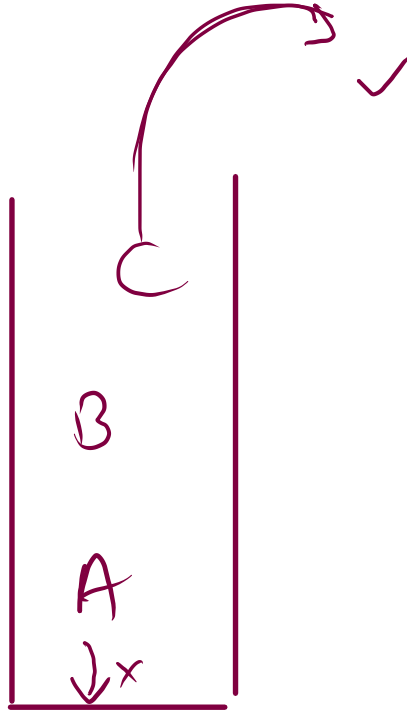


A B C



Stack → LIFO





```java
import java.util.*;
public class Main
{
    public static void main(String[] args) {
        Queue<Integer> qu = new LinkedList<>();
        //add
        qu.add(10);
        qu.add(20);
        qu.add(30);


        //get: front element (first or peek)
        System.out.println(qu.peek());

        //size
        System.out.println(qu.size());

        //remove
        System.out.println(qu.remove());
        System.out.println(qu.peek());
        System.out.println(qu.size());

    }
}

```

Queue Syntax Learning

Problem | Submissions | Leaderboard | Discussions

1. **Declare an Empty** $queue\ s$.
2. Take Single Integer T as input.
3. For next T Lines format $(case, x(optional))$

- case 1. $Print$ the $size$ of the $queue$ in a separate line.
- case 2. $Remove$ an element from the queue. If the queue is empty then print -1 in a separate line.
- case 3. Add Integer x to the $queue\ s$.
- case 4. $Print$ an element at the $front$ of the $queue$. If queue is empty print -1 in a seperate line.

Sample Input 0

```
5
1
2
3 9
4
1
```

Sample Output 0

```
0
-1
9
1
```

```java
public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        Queue<Integer> qu = new LinkedList<>();
        int t = scn.nextInt();
        for(int i = 1; i <= t; i++){
            int caseNo = scn.nextInt();
            if(caseNo == 1){
                System.out.println(qu.size());
            }else if(caseNo == 2){
                if(qu.size() == 0){
                    System.out.println(-1);
                }
                else{
                    qu.remove();
                }
            }else if(caseNo == 3){
                int x = scn.nextInt();
                qu.add(x);
            }else if(caseNo == 4){
                if(qu.size() == 0){
                    System.out.println(-1);
                }
                else{
                    System.out.println(qu.peek());
                }
            }
```