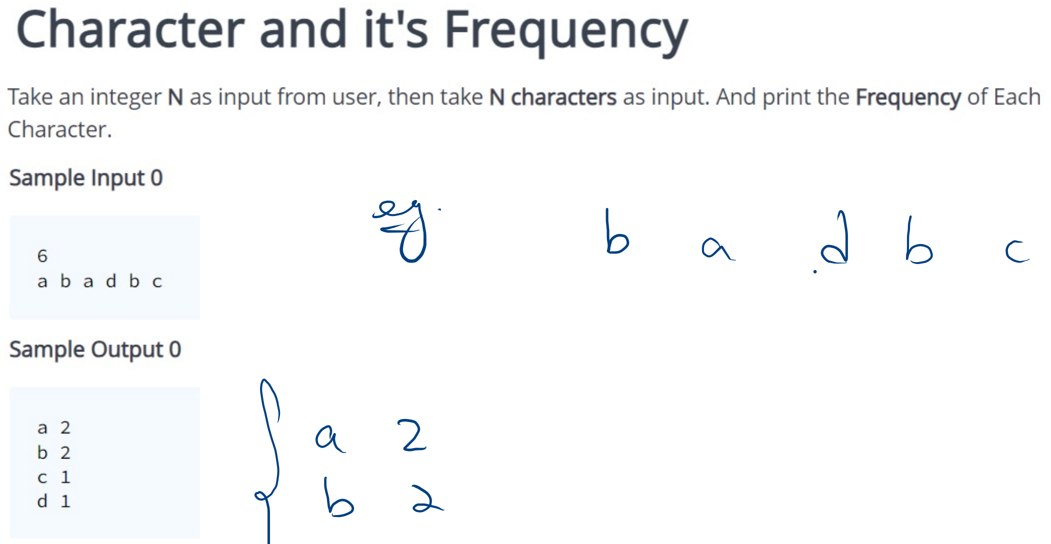


Character and it's Frequency

Take an integer **N** as input from user, then take **N characters** as input. And print the **Frequency** of Each Character.

Sample Input 0

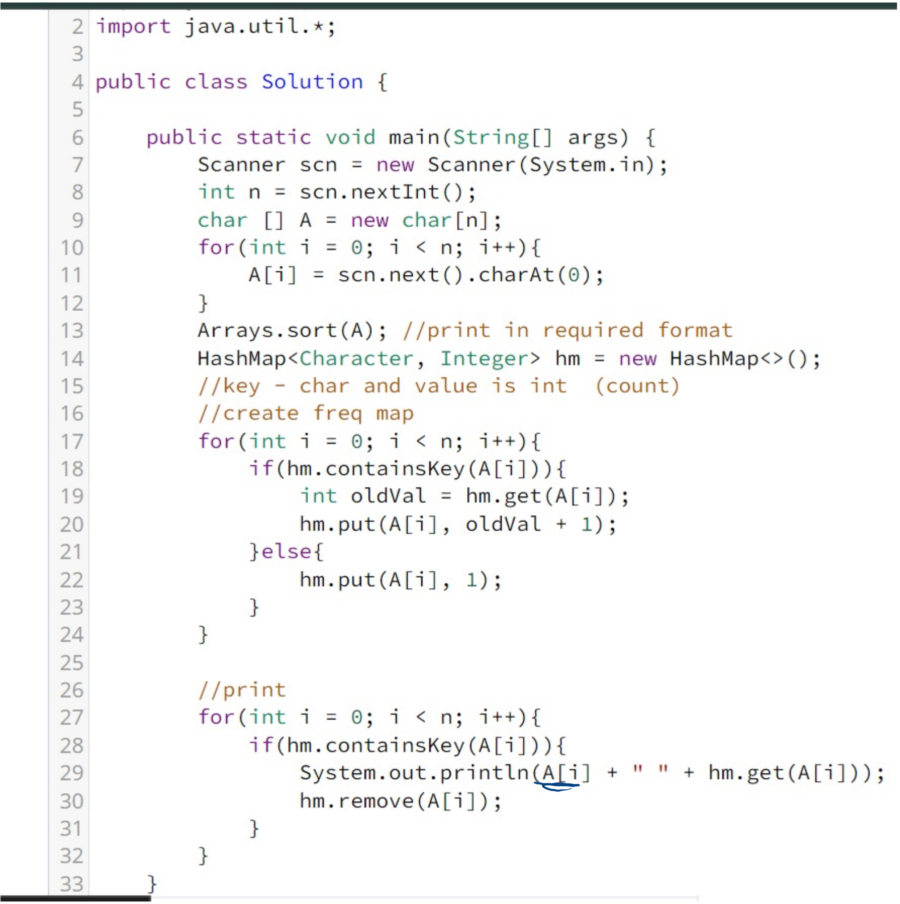
```
6
a b a d b c
```

Sample Output 0

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

eg. b a d b c a

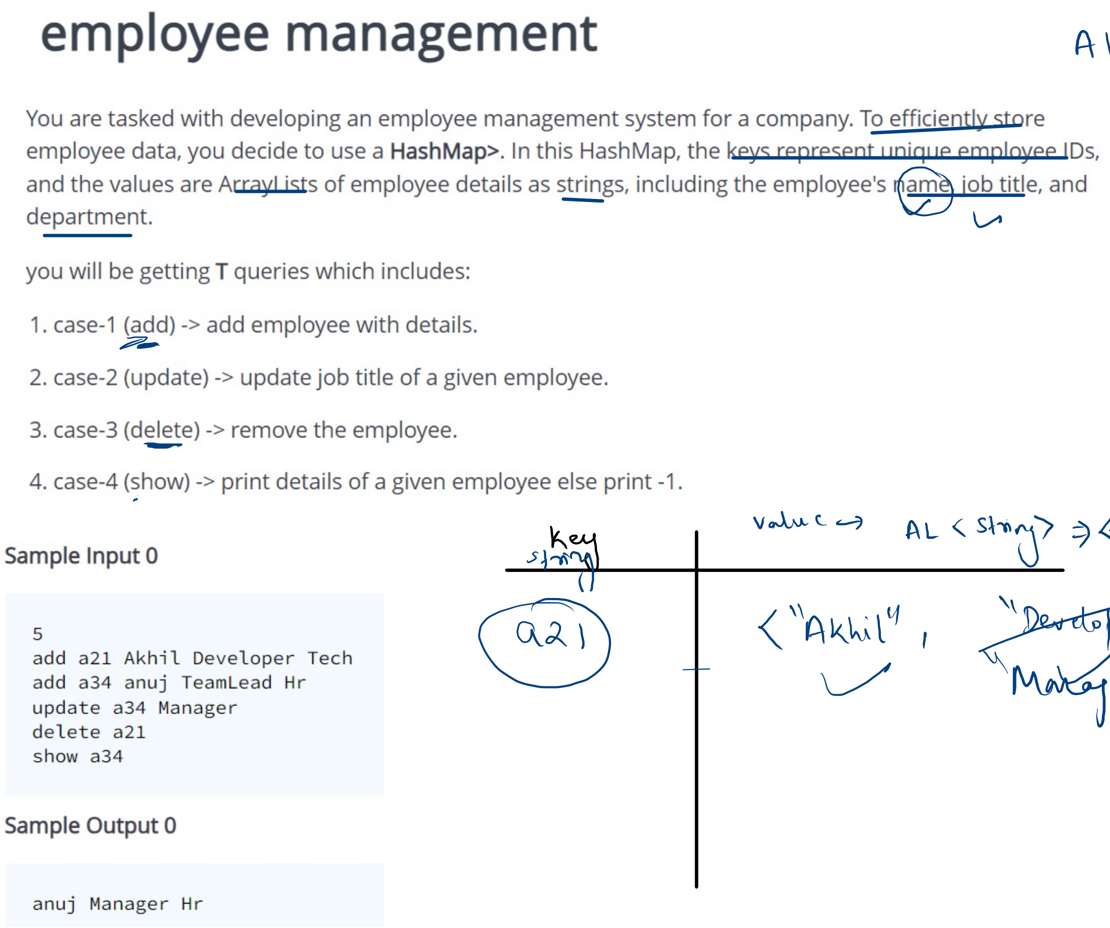
a	2
b	2
c	1
d	1



```java
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        char [] A = new char[n];
        for(int i = 0; i < n; i++){
            A[i] = scn.next().charAt(0);
        }
        Arrays.sort(A); //print in required format
        HashMap<Character, Integer> hm = new HashMap<>();
        //key - char and value is int  (count)
        //create freq map
        for(int i = 0; i < n; i++){
            if(hm.containsKey(A[i])){
                int oldVal = hm.get(A[i]);
                hm.put(A[i], oldVal + 1);
            }else{
                hm.put(A[i], 1);
            }
        }

        //print
        for(int i = 0; i < n; i++){
            if(hm.containsKey(A[i])){
                System.out.println(A[i] + " " + hm.get(A[i]));
                hm.remove(A[i]);
            }
        }
    }
```

a	a	b	b	c	d
0	1	2	3	4	5
					i

a	2
b	2
c	1
d	1



employee management

You are tasked with developing an employee management system for a company. To efficiently store employee data, you decide to use a **HashMap>**. In this HashMap, the keys represent unique employee IDs, and the values are ArrayLists of employee details as strings, including the employee's name, job title, and department.

you will be getting **T** queries which includes:

1. case-1 (add) -> add employee with details.
2. case-2 (update) -> update job title of a given employee.
3. case-3 (delete) -> remove the employee.
4. case-4 (show) -> print details of a given employee else print -1.

Sample Input 0

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

Sample Output 0

```
anuj Manager Hr
```

AL <name, title, dpt.>
String.

key string	value → AL < String > ⇒ <name, title, dpt.>
a21	< "Akhil", ~~"Developer"~~ "Manager", "Tech">

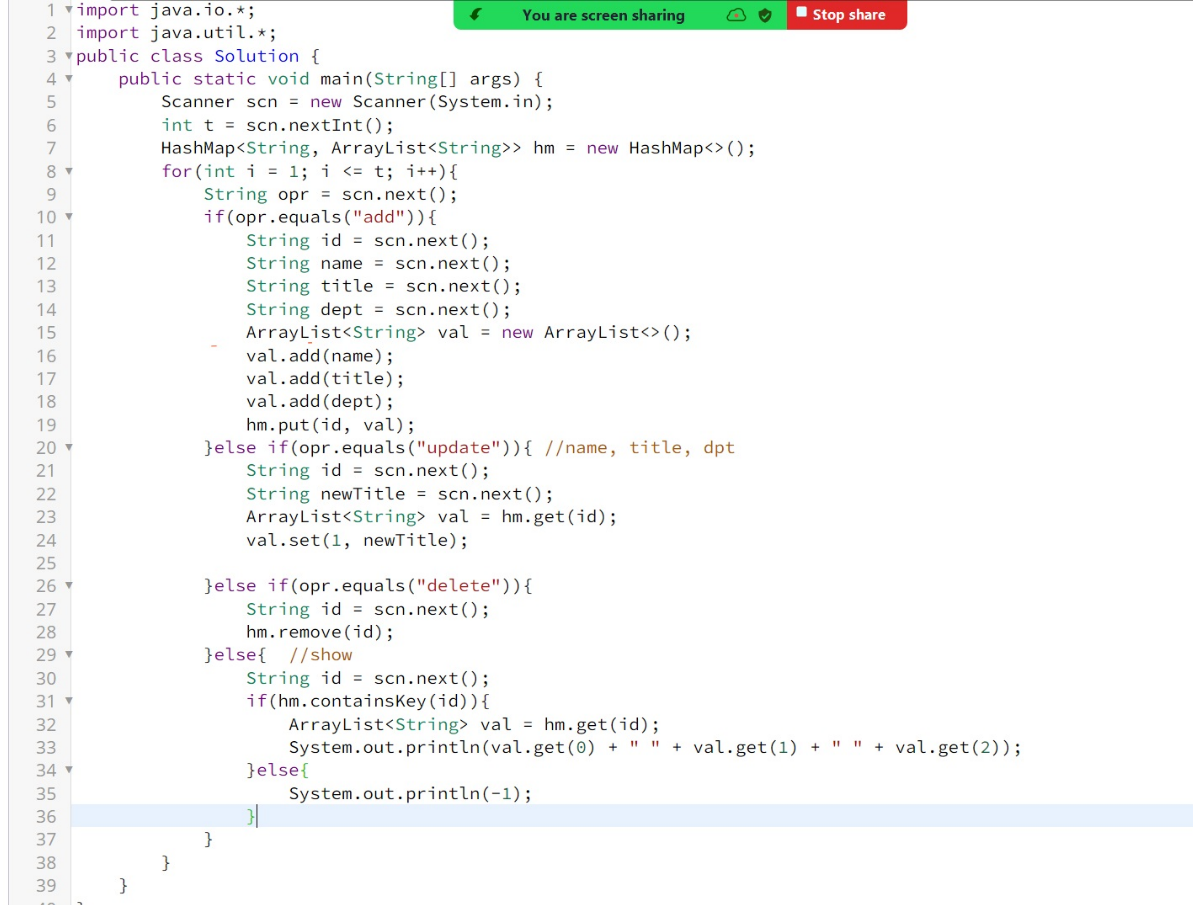
AL<string>

Q21	< "Kanan", "Manager", "Tech" >

string	v/s Integer
"a21"	1002

hm.get("a21") = 1002

Integer a = hmget(a21)



```
import java.io.*;
import java.util.*;
public class Solution {
    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int t = scn.nextInt();
        HashMap<String, ArrayList<String>> hm = new HashMap<>();
        for(int i = 1; i <= t; i++){
            String opr = scn.next();
            if(opr.equals("add")){
                String id = scn.next();
                String name = scn.next();
                String title = scn.next();
                String dept = scn.next();
                ArrayList<String> val = new ArrayList<>();
                val.add(name);
                val.add(title);
                val.add(dept);
                hm.put(id, val);
            }else if(opr.equals("update")){ //name, title, dpt
                String id = scn.next();
                String newTitle = scn.next();
                ArrayList<String> val = hm.get(id);
                val.set(1, newTitle);

            }else if(opr.equals("delete")){
                String id = scn.next();
                hm.remove(id);
            }else{  //show
                String id = scn.next();
                if(hm.containsKey(id)){
                    ArrayList<String> val = hm.get(id);
                    System.out.println(val.get(0) + " " + val.get(1) + " " + val.get(2));
                }else{
                    System.out.println(-1);
                }
            }
        }
    }
}
```

60	20	10	40	50	70	30

$\boxed{x = 50} \rightarrow \underline{O(n)}$.

< 10 20 30 ~~40~~ >

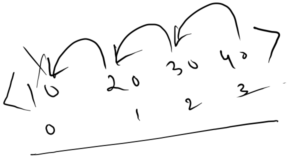
0 1 2 3

$\boxed{O(1)} \rightarrow$ remove from last

< 10 20 30 >

0 1 2

remov(n-1)

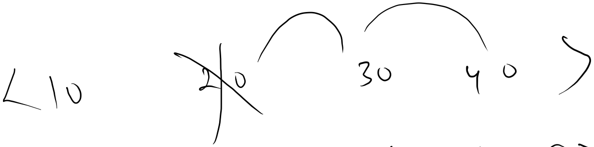


n → $\boxed{n-1}$

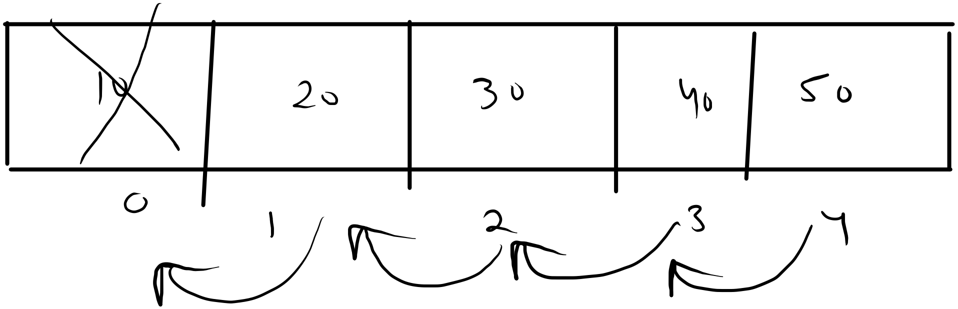
< 20 30 40 >

0 1 2

$\underline{O(n)}$



n/2 $\frac{1}{2}$ O(n).



n-1

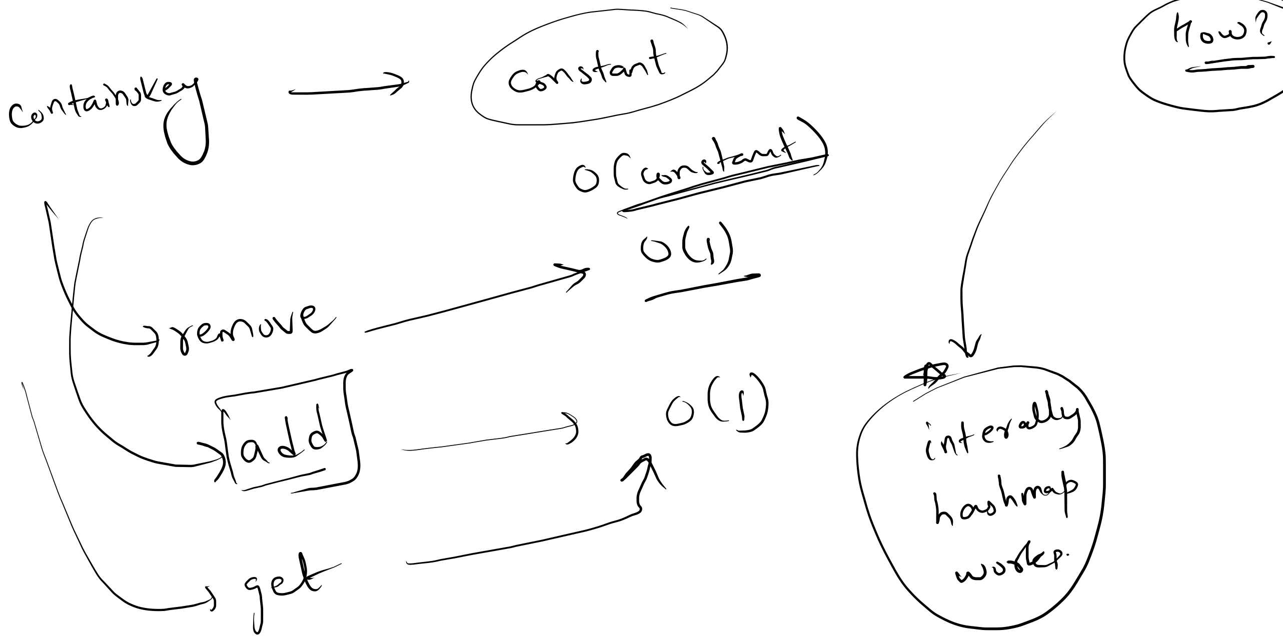
moved to left

20	30	40	50
0	1	2	3

TC = O(n)

O(n-1)

O(n).



containskey → Constant

O(constant)

O(1)

remove

add → O(1)

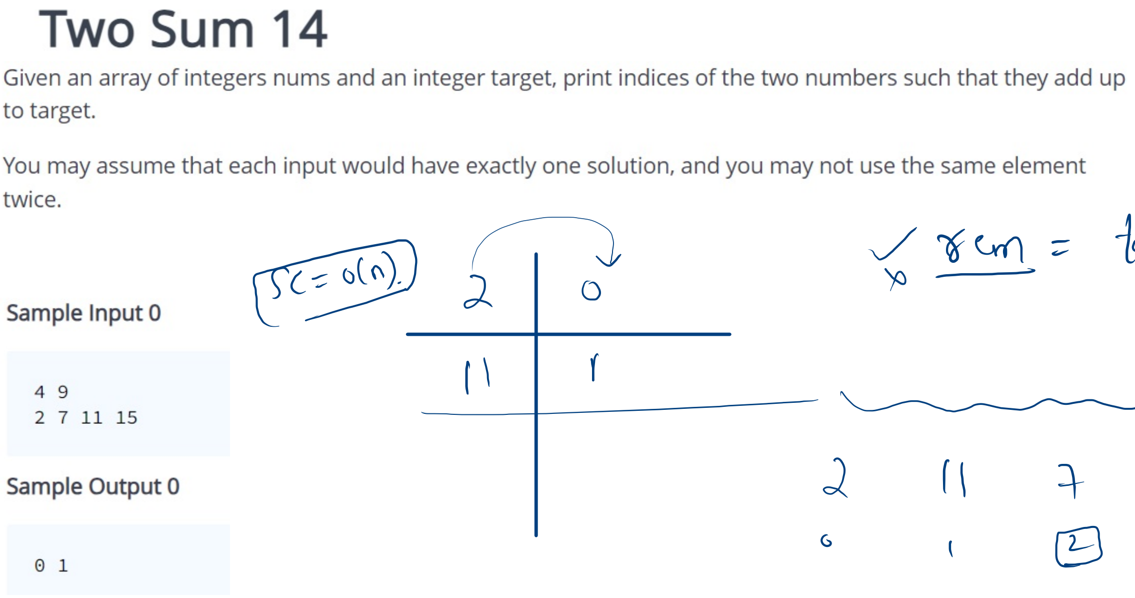
get

How?

interally hashmap works.

Summary,

HM → fast O(1)



Two Sum 14

Given an array of integers nums and an integer target, print indices of the two numbers such that they add up to target.

You may assume that each input would have exactly one solution, and you may not use the same element twice.

Sample Input 0

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

Sample Output 0

```
0 1
```

0 2

$\boxed{SC = O(n)}$

2	0
11	1

✓ rem = tar - A[i]

tar = 9

$\boxed{O(n)}$

2	11	7	15
0	1	$\boxed{2}$	3

i

rem = 2

$\boxed{0 \quad 2}$

A[i], i

tar=9

2	11	15	3	14	7	1
0	1	2	3	4	5	6

i

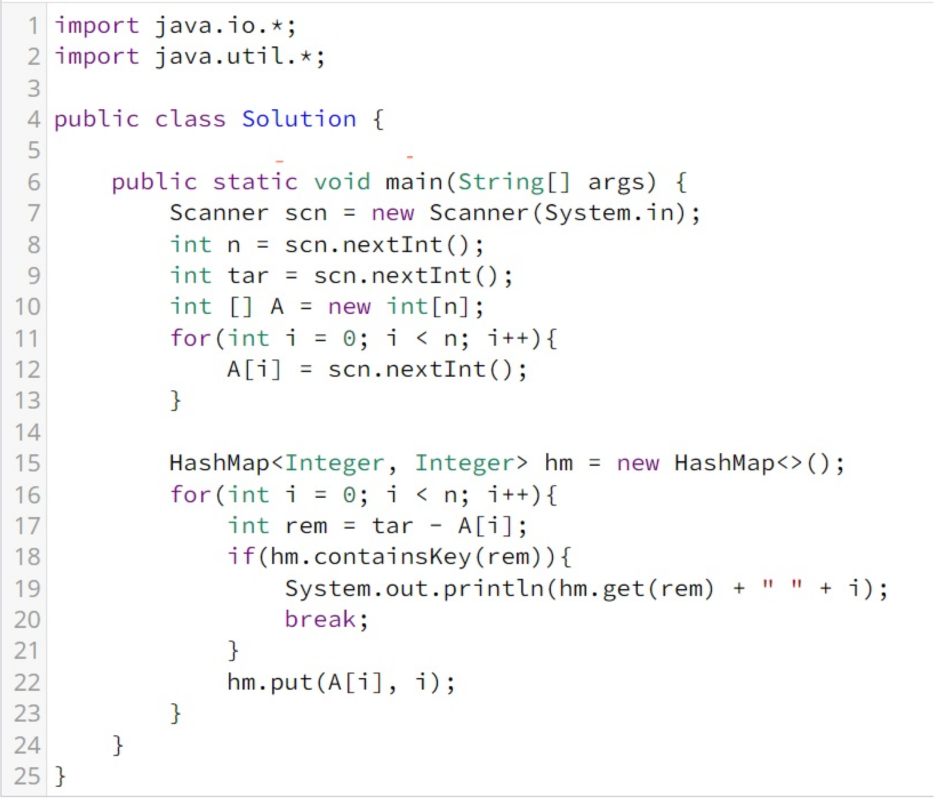
2	0 ✓
11	1
15	2
3	3
14	4

tar - A[i]

rem = 2

hm.get(2)

0 5



```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 tar = scn.nextInt();
        int [] A = new int[n];
        for(int i = 0; i < n; i++){
            A[i] = scn.nextInt();
        }

        HashMap<Integer, Integer> hm = new HashMap<>();
        for(int i = 0; i < n; i++){
            int rem = tar - A[i];
            if(hm.containsKey(rem)){
                System.out.println(hm.get(rem) + " " + i);
                break;
            }
            hm.put(A[i], i);
        }
    }
}
```