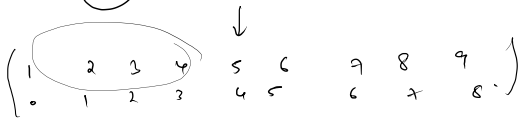


Revision.

1. BS \rightarrow algo. \rightarrow to search element.
 \downarrow
 sorted array.

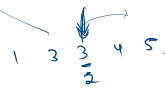
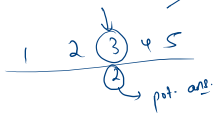
$K=2$



2. last occurrence.

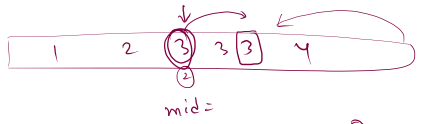


$K=3$



$K=3$

3.



sorted.



dec.



3.

find sq. root.

$n=16 \rightarrow 4$



$2 \times 2 = 4 < 15$
 $3 \times 3 = 9 < 15$

$4 \times 4 = 16 > 15$
 $8 \times 8 = 64 > 16$

American Keyboard

Problem	Submissions	Leaderboard	Discussions
---------	-------------	-------------	-------------

Given an array of strings words, print the words that can be typed using letters of the alphabet on only one row of American keyboard like the image below.

In the American keyboard: the first row consists of the characters "qwertyuiop", the second row consists of the characters "asdfghjkl", and the third row consists of the characters "zxcvbnm".

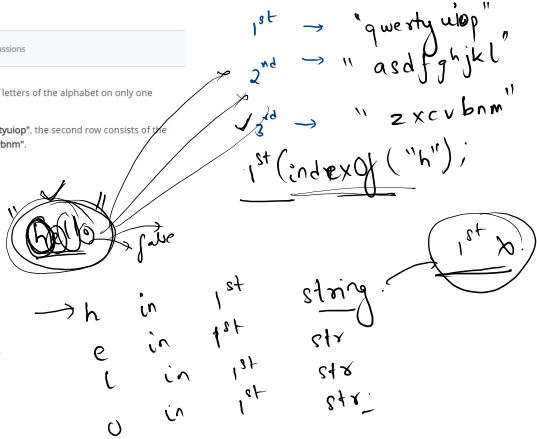
If such word is not present in array, then print -1.

Sample Input 0

```
4
hello
aLaska
dad
peace
```

Sample Output 0

```
aLaska dad
```



```
public static boolean checkInRow(String curr, String row){
    for(int i = 0; i < curr.length(); i++){
        char ch = curr.charAt(i);
        if(row.indexOf(ch) == -1){
            return false;
        }
    }
    return true;
}
```

"hello", "qwertyuiop"

3m, O(m)

O(nm).

func x 3

O(nm).

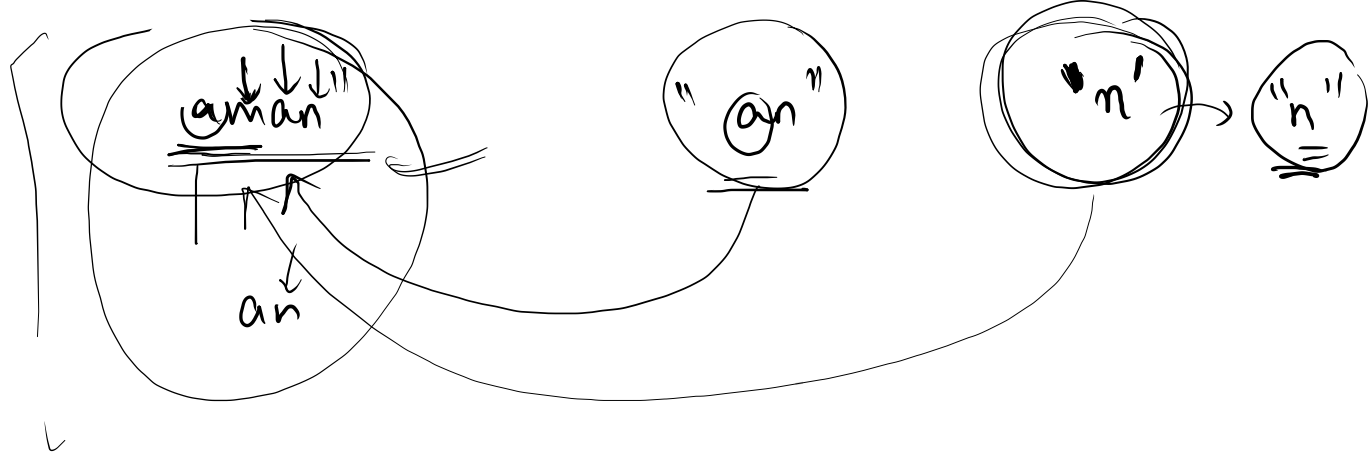
```
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();

    String [] arr = new String[n];
    for(int i = 0; i < arr.length; i++){
        arr[i] = scn.next();
    }

    String row1 = "qwertyuiop";
    String row2 = "asdfghjkl";
    String row3 = "zxcvbnm";
    boolean wordPresent = false;

    for(int i = 0; i < n; i++){
        String curr = arr[i];
        boolean a1 = checkInRow(curr, row1);
        boolean a2 = checkInRow(curr, row2);
        boolean a3 = checkInRow(curr, row3);

        if(a1 || a2 || a3){
            System.out.print(curr + " ");
            wordPresent = true;
        }
    }
    if(wordPresent == false){
        System.out.println(-1);
    }
}
```



Find The Index of Rotation

Problem	Submissions	Leaderboard	Discussions
---------	-------------	-------------	-------------

Given a sorted and rotated array. Find the index at which the array is rotated using binary search.

Sample Input 0

0 1 2 3 4

Sample Output 0

0

1 2 3 4 5



?

eg.

4 5 1 2 3
0 1 2 3 4

ans = 1

find the max ele in rotated sorted array.

1 2 3 4 5

4 5 1 2 3
0 1 2 3 4

$A[m] > A[m+1]$

$\hookrightarrow m$

$A[m] < A[m-1]$

$\hookrightarrow m-1$

$m = 2$

5 1 2 3 4
0 1 2 3 4

$m = 2$

$2 < 4$

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

public class Solution {
    public static int solve(int [] A){
        int low = 0;
        int high = A.length-1;
        int ans = -1;

        while(low <= high){
            int mid = (low + high) / 2;
            if(A[mid] > A[mid+1]){
                ans = mid;
                break;
            }
            else if (A[mid] < A[mid - 1]){
                ans = mid-1;
                break;
            }
            else if(A[low] < A[mid]){
                low = mid + 1;
            }
            else if(A[mid] < A[high]){
                high = mid-1;
            }
        }
        return ans;
    }
}
```

ans = 1

7 8 1 2 3 4 5 6
0 1 2 3 4 5 6 7

$m = 3$

$2 < 6$

$m = 1$

eliminating half.

```
public static void main(String[] args) {
    //this que is same as find the max in sorted rotated array

    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();
    }

    int idx = solve(A); //point of rotation or idx of max ele
    System.out.println(idx);
}
```

Find Element in Rotated Array

Given a sorted and rotated array `arr[]` of size `N` and a key, the task is to find the key in the array and return the index of the key.

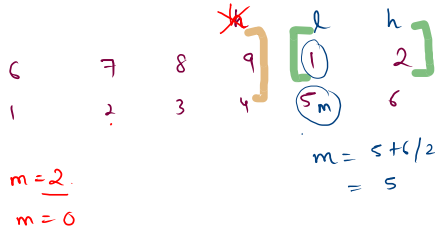
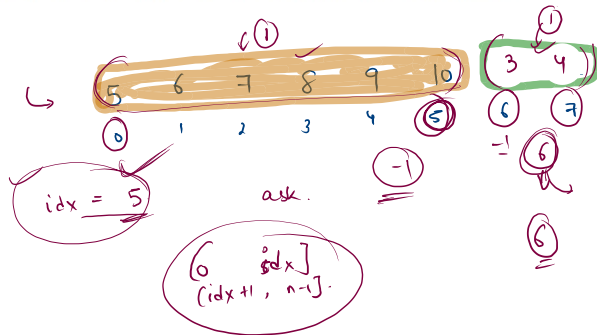
Sample Input 0

```
8
5 6 7 8 9 10 12
1
```

Sample Output 0

```
6
```

Evaluation 0



k=1

```
1 import java.io.*;
2 import java.util.*;
3
4 public class Solution {
5     public static int solve(int [] A){
6         int low = 0;
7         int high = A.length-1;
8         int ans = -1;
9
10        while(low <= high){
11            int mid = (low + high) / 2;
12            if(A[mid] > A[mid-1]){
13                ans = mid;
14                break;
15            }
16            else if (A[mid] < A[mid - 1]){
17                ans = mid-1;
18                break;
19            }
20            else if(A[low] <= A[mid]){
21                low = mid + 1;
22            }
23            else if(A[mid] <= A[high]){
24                high = mid-1;
25            }
26        }
27        return ans;
28    }
29
30    public static int binarySearch(int [] A, int low, int high, int x){
31        while(low <= high){
32            int mid = (low + high) / 2;
33            if(A[mid] == x){
34                return mid;
35            }
36            else if (A[mid] > x){
37                high = mid - 1;
38            }
39            else{
40                low = mid + 1;
41            }
42        }
43        return -1;
44    }
45
46    public static void main(String[] args) {
47        Scanner sc = new Scanner(System.in);
48        int n = sc.nextInt();
49        int [] A = new int[n];
50        for(int i = 0; i < n; i++){
51            A[i] = sc.nextInt();
52        }
53        int k = sc.nextInt();
54        int idx = solve(A); // idx -> idx of max element in my rotated sorted array?
55        int ans1 = binarySearch(A, 0, idx, k);
56        int ans2 = binarySearch(A, idx+1, n-1, k);
57
58        int ans = ans1 == -1 ? ans2 : ans1;
59        System.out.println(ans);
60    }
61 }
```

```
1 public static int binarySearch(int [] A, int low, int high, int x){
2     while(low <= high){
3         int mid = (low + high) / 2;
4         if(A[mid] == x){
5             return mid;
6         }
7         else if (A[mid] > x){
8             high = mid - 1;
9         }
10        else{
11            low = mid + 1;
12        }
13    }
14    return -1;
15 }
16
17 public static void main(String[] args) {
18     Scanner sc = new Scanner(System.in);
19     int n = sc.nextInt();
20     int [] A = new int[n];
21     for(int i = 0; i < n; i++){
22         A[i] = sc.nextInt();
23     }
24     int k = sc.nextInt();
25     int idx = solve(A); // idx -> idx of max element in my rotated sorted array?
26     int ans1 = binarySearch(A, 0, idx, k);
27     int ans2 = binarySearch(A, idx+1, n-1, k);
28
29     int ans = ans1 == -1 ? ans2 : ans1;
30     System.out.println(ans);
31 }
32 }
```