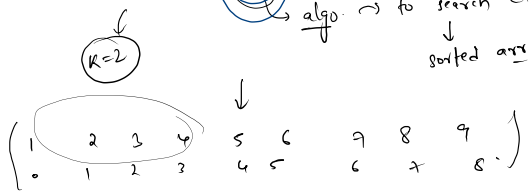
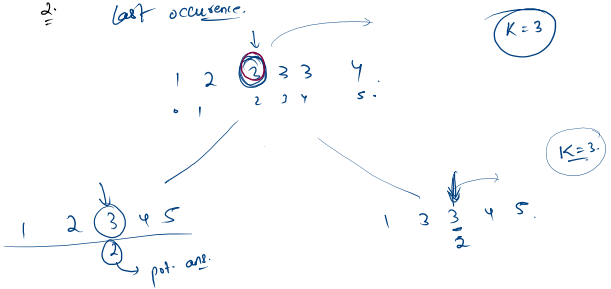
Revision.

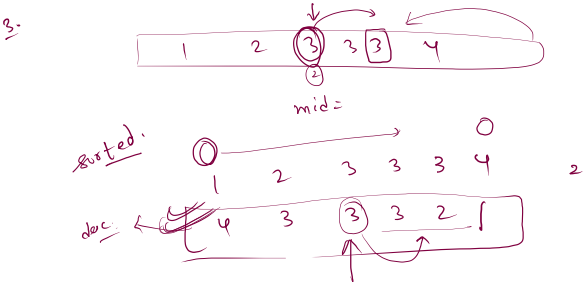
1. B.S. → algo. → to search element.
↓
sorted array.



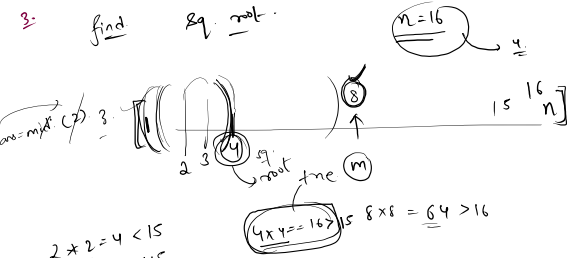

2. Last occurrence.




3.




3. find Sq. root.



# 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"**.

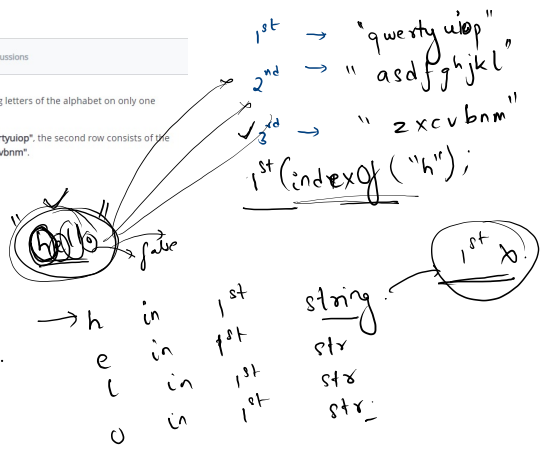
If such word is not present in array, then print **-1**.

### Sample Input 0

```
4
hello
alaska
dad
peace
```

### Sample Output 0

```
alaska dad
```





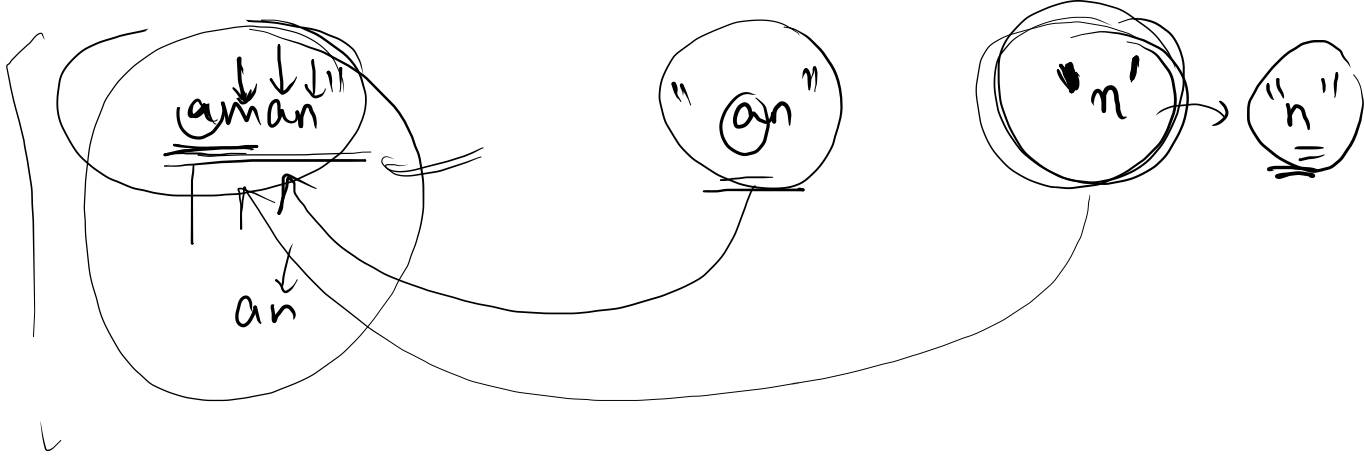
```
public static boolean checkInRow(String curr, String row){
    for(int i = 0; i < curr.length(); i++){
        char ch = curr.charAt(i);
        if(row.indexOf(ch) == -1){
            return false;
        }
    }
    return true;
}
```


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);
    }

}
```


aman
an
"an"
'n'
"n"

# Find The Index of Rotation

Problem | Submissions | Leaderboard | Discussions

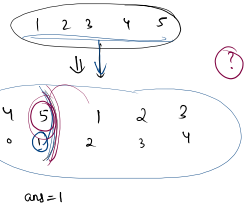
Given a sorted and rotated array. Find the index at which the array is rotated using binary search.

Sample Input 0

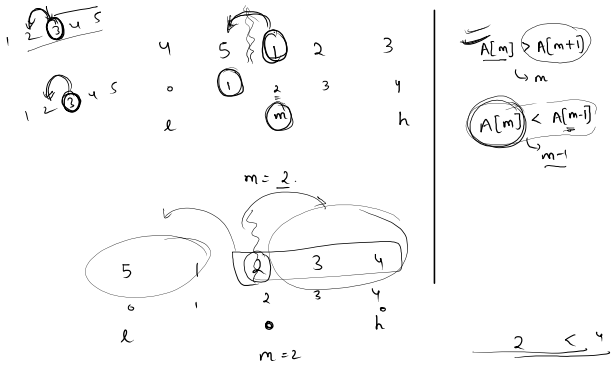
```
5
1 2 3 4 5
```

Sample Output 0

```
1
```




Que: find the max ele in rotated sorted array.



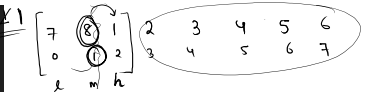




2 < 4

```java
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




m = 3.

m = 1

2 < 6

eliminating half

```java
    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

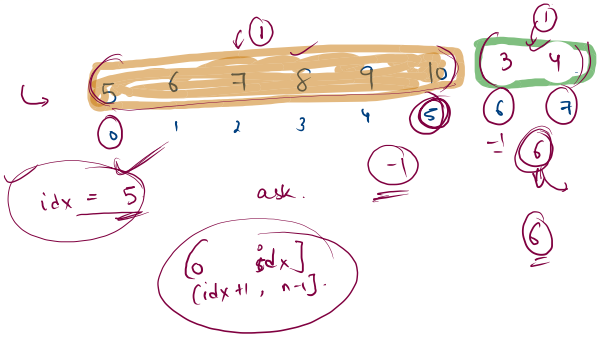
Given a sorted and rotated array arr[] of size N and a key, the task is to find the key in the array and return the index of the key.

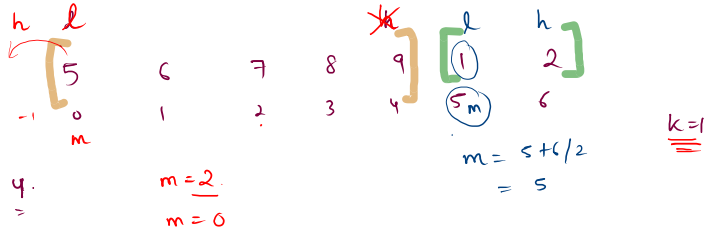
**Sample Input 0**

```
8
5 6 7 8 9 10 1 2
1
```

**Sample Output 0**

```
6
```




```java
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;

    }
```

```java
public static int binarySearch(int [] A, int low, int high, int x){
    while(low <= high){
        int mid = (low + high) / 2;
        if(A[mid] == x){
            return mid;
        }
        else if (A[mid] > x){
            high = mid - 1;
        }
        else{
            low = mid + 1;
        }
    }

    return -1;
}

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();
    }
    int k = scn.nextInt();
    int idx = solve(A);     // idx -> idx of max element in my rotated sorted array?

    int ans1 = binarySearch(A, 0, idx, k);
    int ans2 = binarySearch(A, idx+1, n-1, k);

    int ans = ans1 == -1 ? ans2 : ans1;
    System.out.println(ans);



}
```