Search Character

(sorted)

arr = ['b', 'e', 'q', 's', 'y']

(char)

ch = 'x' , ans = y

ch = 'm' , ans = q

ch = 'f' , ans = q

ch = 'z' , ans = -1

ch = 'a' , ans = b

ch = b, ans = e

ch = b, b > x ✗

e e > x, ✗

q q > x, ✗

s s > x, ✗

y y > x ✓

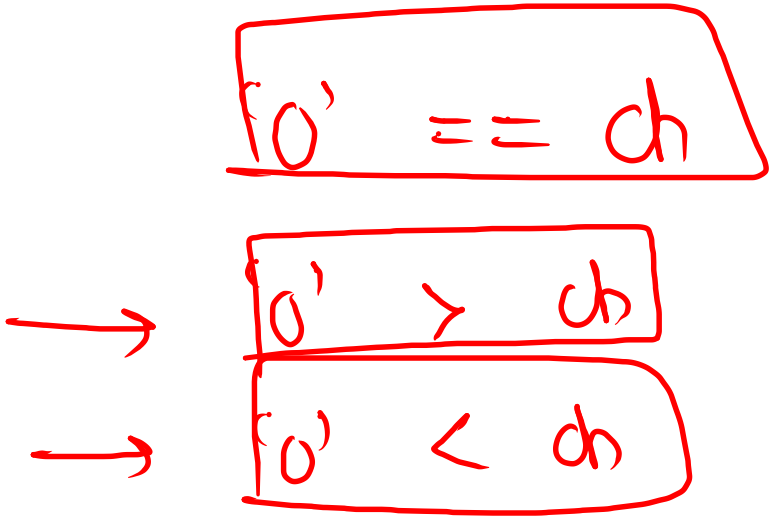
example

0	1	2	3	4	5	6	7	8	9	10	11	12
'a'	'b'	'd'	'f'	'h'	'k'	'o'	'p'	'q'	't'	'v'	'w'	'x'

arr = ['a', 'b', 'd', 'f', 'h', 'k', 'o', 'p', 'q', 't', 'v', 'w', 'x']

si ↑ (0) mid ↑ (6) ei ↑ (12)

ch = w



psudo code

until (si <= ei)

mid = (si + ei) / 2 ;

- arr[mid] == ch , si = mid+1
- arr[mid] > ch , ei = mid-1
- arr[mid] < ch , si = mid+1

example

0	1	2	3	4	5	6	7	8	9	10	11	12
'a'	'b'	'd'	'f'	'h'	'k'	'o'	'p'	'q'	't'	'v'	'w'	'x'

arr = ['a', 'b', 'd', 'f', 'h', 'k', 'o', 'p', 'q', 't', 'v', 'w', 'x']

ei si

mid

ch = 'y'

```
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    char ch = scn.next().charAt(0);
    int n = scn.nextInt();
    char[] arr = new char[n];
    for (int i = 0; i < n; i++) {
        arr[i] = scn.next().charAt(0);
    }

    searchCharacter(arr, n, ch);
}

public static void searchCharacter(char[] arr, int n, char ch) {
    int si = 0;
    int ei = n - 1;
    while ( si <= ei ) {
        int mid = (si + ei) / 2;
        if ( arr[mid] <= ch ) {
            si = mid + 1;    // shift range to right
        } else {
            ei = mid - 1;    // shift range to left
        }
    }

    if (si == n) {
        System.out.println( -1 );
    } else {
        System.out.println( arr[si] );
    }
}
```

si = 0, ei = 12

mid = (0+12)/2 = 6 'o' <= 'y'

mid = (7+12)/2 = 9 't' <= 'y'

mid = (10+12)/2 = 11 w <= y

mid = (12+12)/2 = 12 x <= y

example

ch = 'c'

cwor ch

mid = (0+12)/2 = 6 , 'o' > c

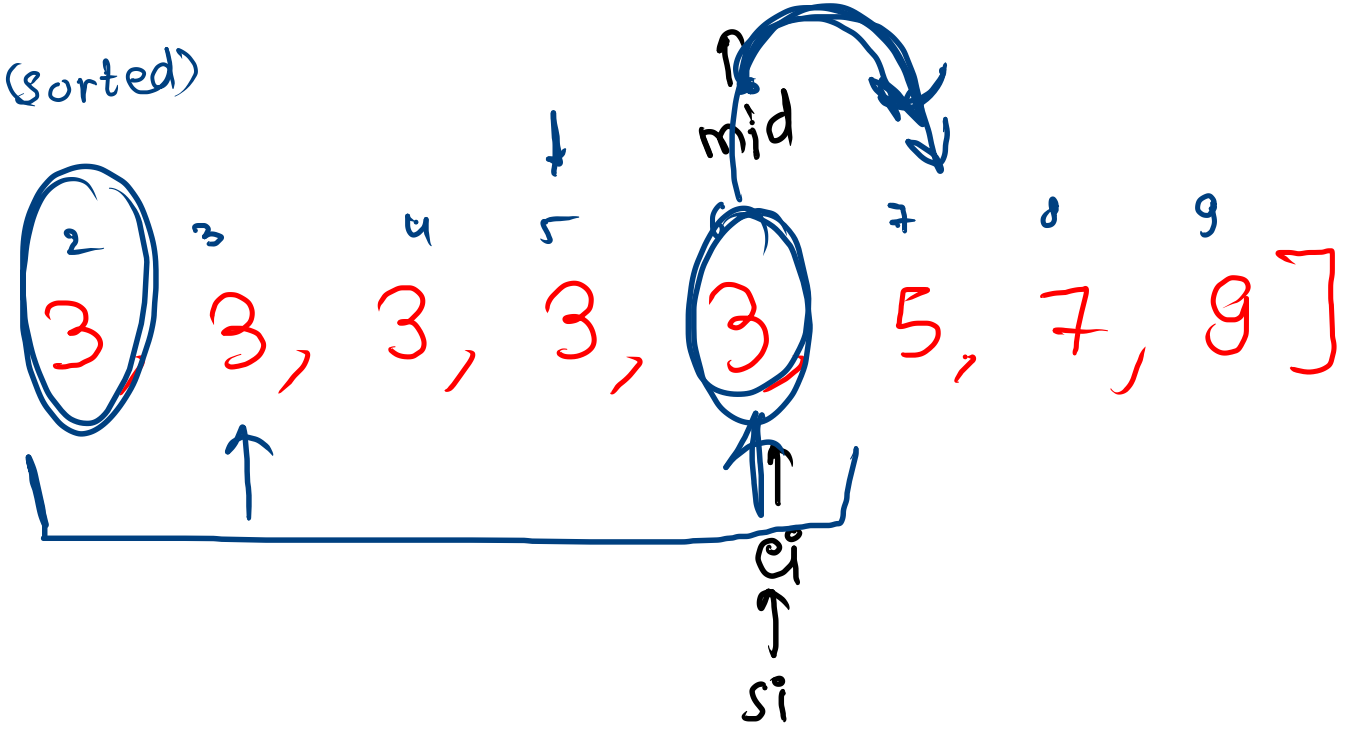
mid = (0+5)/2 = 2 , 'd' > c

mid = (0+1)/2 = 0 'a' <= c

mid = (1+1)/2 = 1 'b' <= c

ans = 'd'

Find Last Occurrence



arr = [1, 2, 3, 3, 3, 3, 3, 5, 7, 9]

target = 3

mid = (0+9)/2 = 4

mid = (5+9)/2 = 7

mid = (5+6)/2 = 5

mid = (6+6)/2 = 6

Binary Search Upper Bound (BSUB)

```java
public static void binarySearch(int[] arr, int n, int target) {
    int si = 0;
    int ei = n - 1;
    while ( si <= ei ) {
        int mid = (si + ei) / 2;
        if ( arr[mid] == target ) {
            if ( mid + 1 < arr.length && arr[mid] == arr[mid + 1]) {
                si = mid + 1;
            } else {
                System.out.println(mid);
                return;
            }
        } else if ( arr[mid] > target ) {
            ei = mid - 1;
        } else {
            si = mid + 1;
        }
    }
    System.out.println(-1);
}
```

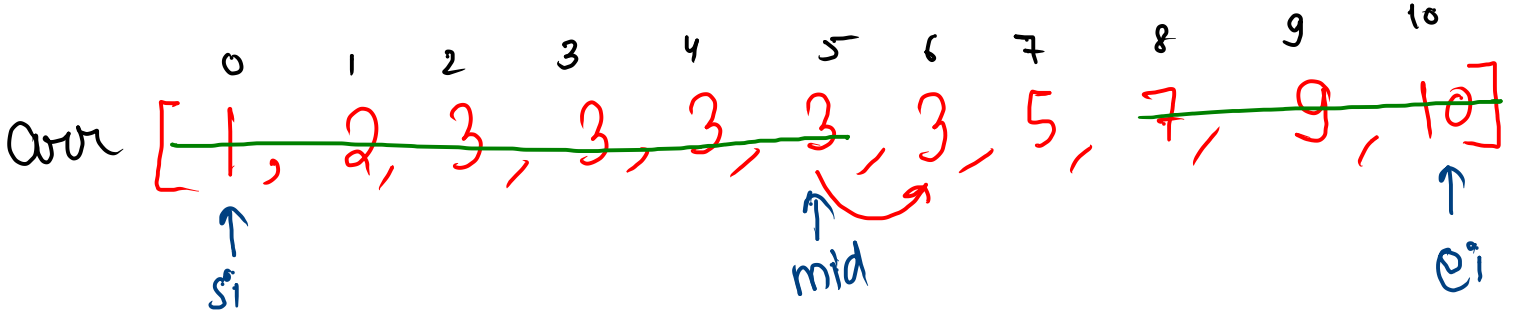
si = 0, ei = 10 t = 3

mid = 5

mid = (6+10)/2 = 8

mid = (6+7)/2 = 6

mid = 6

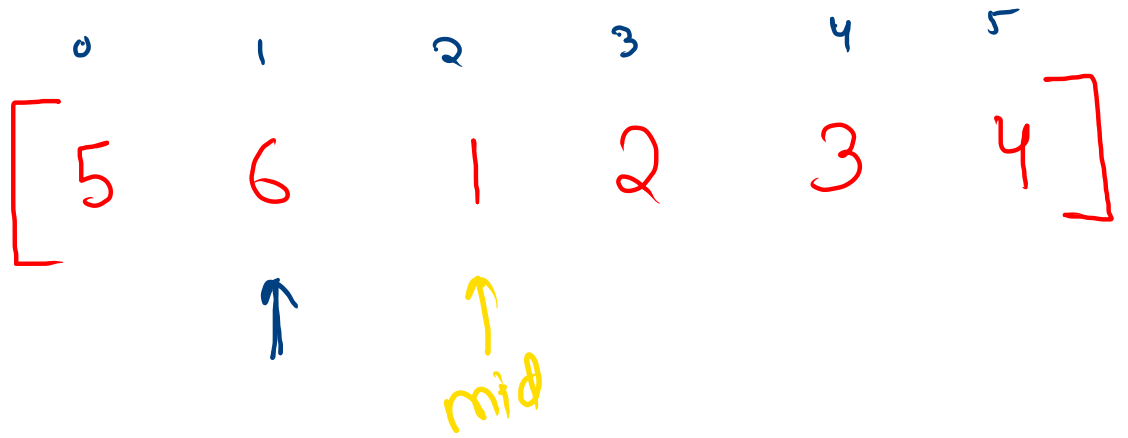


⇒ Binary Search Lower Bound (BSLB)

```java
public static void binarySearch(int[] arr, int n, int target) {
    int si = 0;
    int ei = n - 1;
    while ( si <= ei ) {
        int mid = (si + ei) / 2;
        if ( arr[mid] == target ) {
            if ( mid - 1 >= 0 && arr[mid] == arr[mid - 1] ) {
                ei = mid - 1;
            } else {
                System.out.println(mid);
                return;
            }
        } else if ( arr[mid] > target ) {
            ei = mid - 1;
        } else {
            si = mid + 1;
        }
    }
    System.out.println(-1);
}
```

Find The Index of Rotation

Example 1



ans = 1

Example 2

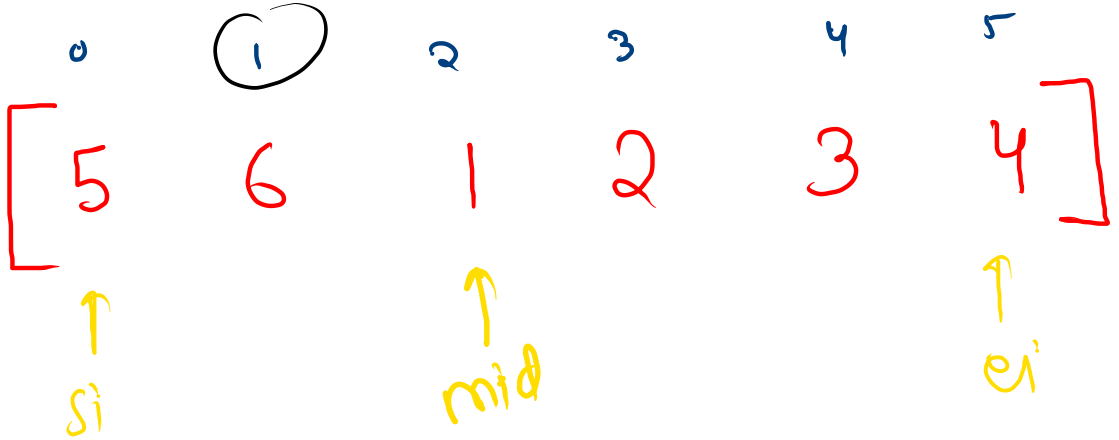
→ [2 4 5 6 7]

1

→ [6 7 2 4 5]

ans = 1

Example 1



ans = 1

```
if ( arr[mid] > arr[mid+1] ){
        Syso( mid)
    return
} else if ( arr[mid] > arr[0])
        si = mid+1 ;
} else {
        ei = mid ;
}
```