$o(n \log n)$ → inbuilt

$\left(O(1) \quad O(\log n) \quad O(n) \quad O(n \log n) \quad O(n^2) \right)$

Binary Search

↳ * Sorted array.

{ Sort + task }

→ $O(n \log n) + O(n) = O(n \log n)$

$O(n^2) + O(n) = O(n^2)$

key = 70

$\log_2 n$

1	2	3	4	5	6	7	~~8~~	9
0	1	2	3	4	5	6	7	8

$9/2 = 4$

$\frac{4}{2} = 2$

$\frac{2}{2} = 1 \qquad \frac{1}{2} = 0$

↑ j ↑ i

10^5 → 1 lakhs

10^6 → 10 lakhs.

$m = 4$

$m = 6$

$m = 7$

$m = 8$

↑ m

Binary Search

	Sort ✓	→	task ✓	}	TC ✓
bubble sort.	$O(n^2)$	+	$O(n)$	}	$O(n^2)$
inbuilt. sorting	$O(n\log n)$	+	$O(n)$	}	$O(n\log n)$*
	$O(n\log n)$	+	$O(n^2)$	}	$O(n^2)$

$\underline{\underline{\text{Sort}}}$ + linear search } TC

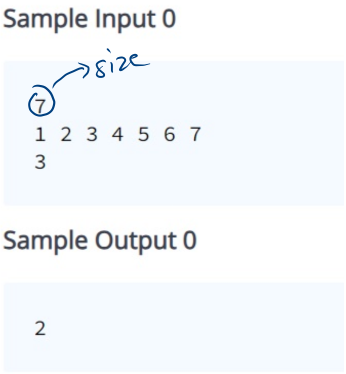
$O(n^2)$ $O(n)$

$$\boxed{n^{2} + n} \longrightarrow \boxed{\underline{\underline{O(n^{2})}}}\,.$$

Binary Search.

-1 → if not present

key = 3



Sample Input 0

7 → size
1 2 3 4 5 6 7
3

Sample Output 0

2

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

i j

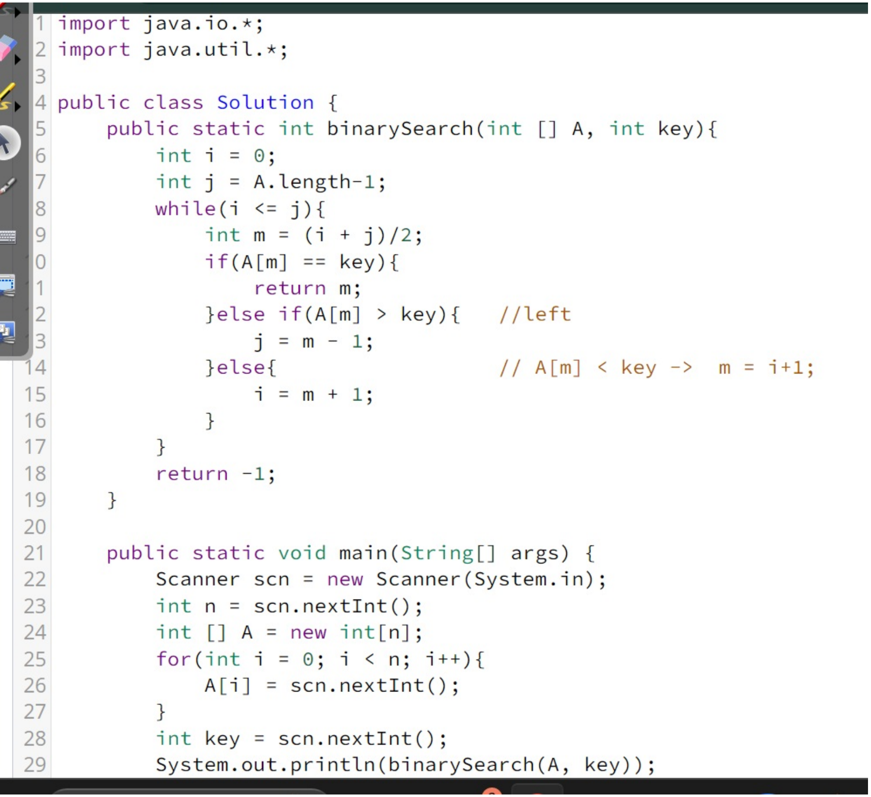
m = i+j/2

m=3.

m=1

m=2.

```
if (A[m] == key)
    ↳ m is ans
else if A[m] > key
    ↳ left → j=m-1
else
    ↳ right → i=m+1
```



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

public class Solution {
    public static int binarySearch(int [] A, int key){
        int i = 0;
        int j = A.length-1;
        while(i <= j){
            int m = (i + j)/2;
            if(A[m] == key){
                return m;
            }else if(A[m] > key){   //left
                j = m - 1;
            }else{                  // A[m] < key ->  m = i+1;
                i = m + 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 key = scn.nextInt();
        System.out.println(binarySearch(A, key));
```

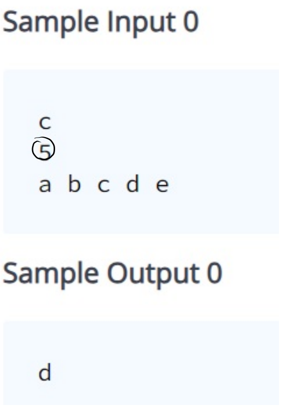
key=3

1 2 3 4 5 6 7

0 1 2 3 4 5 6

↳ 2

Search Character

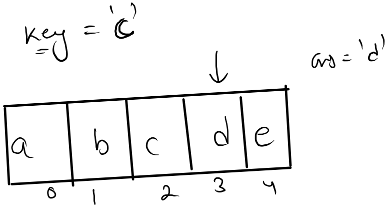


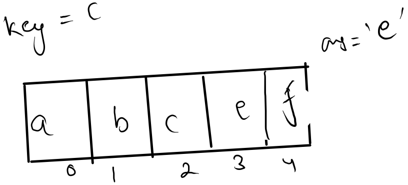
Sample Input 0

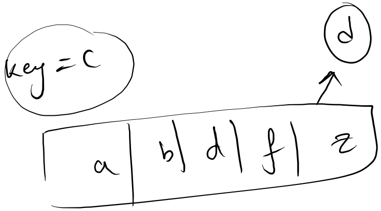
```
c
5
a b c d e
```

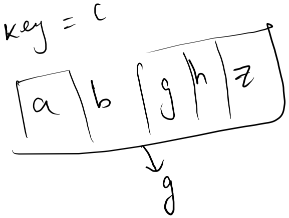
Sample Output 0

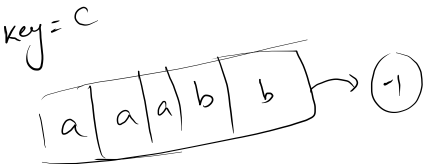
```
d
```











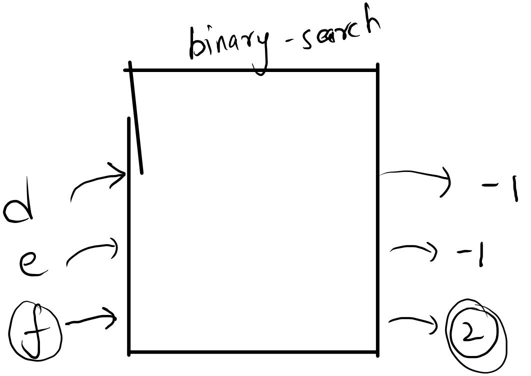
key = c

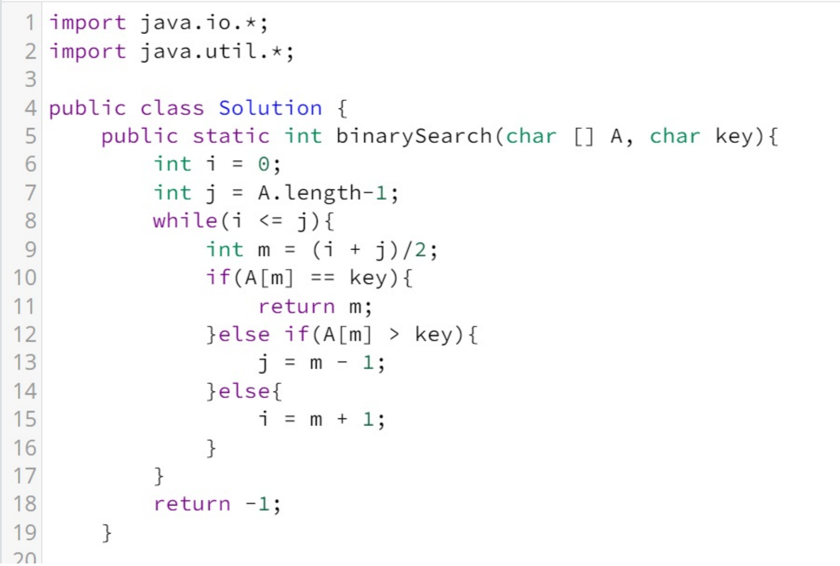
a	b	f	g	h
0	1	2	3	4

d X

e X

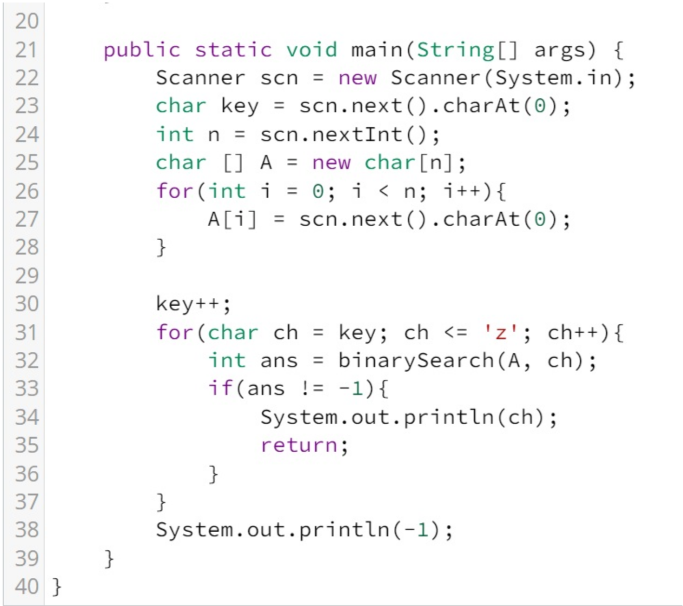
f





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

public class Solution {
    public static int binarySearch(char [] A, char key){
        int i = 0;
        int j = A.length-1;
        while(i <= j){
            int m = (i + j)/2;
            if(A[m] == key){
                return m;
            }else if(A[m] > key){
                j = m - 1;
            }else{
                i = m + 1;
            }
        }
        return -1;
    }
```



```java

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

        key++;
        for(char ch = key; ch <= 'z'; ch++){
            int ans = binarySearch(A, ch);
            if(ans != -1){
                System.out.println(ch);
                return;
            }
        }
        System.out.println(-1);
    }
}
```

key = b

ans = f

a	b	f	g	h
0	1	2	3	4

Find Last Occurrence

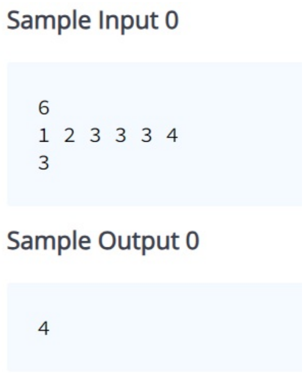
ans = ~~4~~ 7 ✓

$i \le j$

key = 3

1	2	2	2	3	3	3	3	4	5
0	1	2	3	4	5	6	7	8	9

j i
m



Sample Input 0

```
6
1 2 3 3 3 4
3
```

Sample Output 0

```
4
```

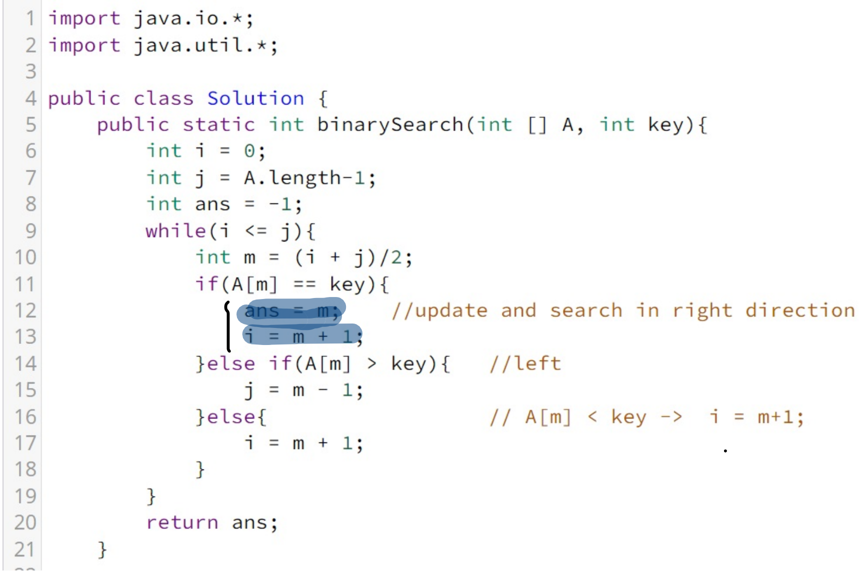
m = 4 ✓

m = 7 ✓

m = 8 ✓

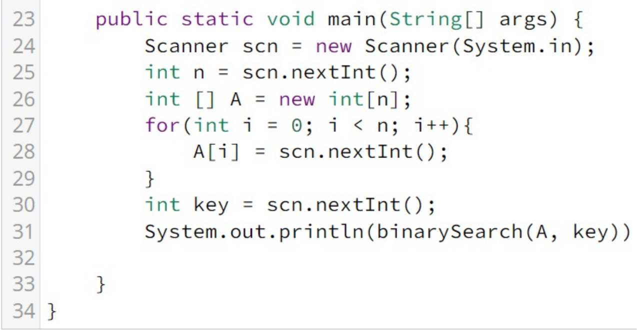
A[m] == key → ans = m

A[m] > key → left



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

public class Solution {
    public static int binarySearch(int [] A, int key){
        int i = 0;
        int j = A.length-1;
        int ans = -1;
        while(i <= j){
            int m = (i + j)/2;
            if(A[m] == key){
                ans = m;       //update and search in right direction
                i = m + 1;
            }else if(A[m] > key){    //left
                j = m - 1;
            }else{                   // A[m] < key ->  i = m+1;
                i = m + 1;
            }
        }
        return ans;
    }
```



```java
    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 key = scn.nextInt();
        System.out.println(binarySearch(A, key))

    }
}
```

first occurence.

key = 2

ans = ~~-1~~ ~~5~~ ~~2~~ 1

	m					li				
1	2	2	2	2	2	2	3	3	4	5
0	1	2	3	4	5	6	7	8	9	10
j	j									
	fi									

A[m] == key

↳ ans = m

left

m = 5

m = 2

m = 0

m = 1