Ques given an array with duplicate values. we have to find how many target elements are present in array. in log(N)

(sort)

arr =

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

target = 2

soln

```
int idx1 = BSLB( );
int idx2 = BSUB( );
ans = idx2 - idx1 + 1;
```

Search insert position

arr =

0	1	2	3	4	5	6	7	8	9	10	11
1	3	7	8	9	10	12	15	25	35	36	40

ei

si

mid

target = 20

ans = 8

psudo code

```
int si=0, ei=n-1;
while ( si <= ei) {
    int mid = (si+ei)/2;
    if( arr[mid] == target){
        return mid;
    } else if (arr[mid] < target){
        si = mid+1;
    } else {
        ei = mid-1;
    }
}
return si;
```

only word changed from template

Code

```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int[] arr = new int[n];
    for (int i = 0; i < n; i++) {
        arr[i] = scn.nextInt();
    }
    int target = scn.nextInt();
    int ans = binarySearch(arr, n, target);
    System.out.println(ans);
}
public static int 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 ) {
            return mid;
        } else if ( arr[mid] > target ) {
            ei = mid - 1;
        } else {
            si = mid + 1;
        }
    }
    return si;
}
```

$T.C = O(\log(n))$

Note:- advantage of Binary Search

we don't even need an array to use B.S

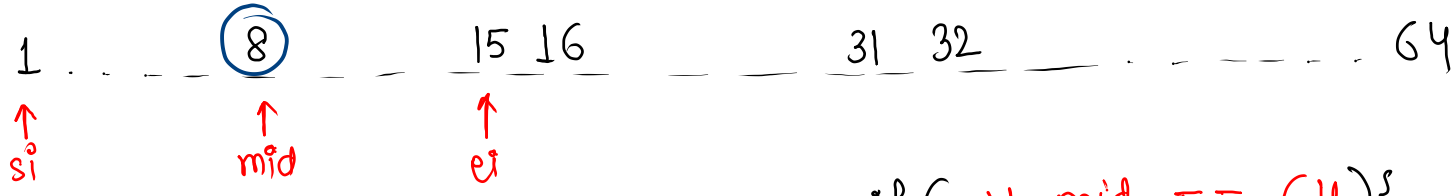
(We can use B.S on an imaginary sequence)

Find Square Root

int n = 64 , ans = $\sqrt{64}$ = 8

si = 1 , ei = 64

imaginary sequence




```
if(mid*mid == 64){
    return mid;
}else if (mid*mid > 64){
    ei = mid-1;
}else if (mid*mid < 64){
    si = mid+1;
}
```

Ex:- n = 25 , ans = 5

n = 41 , ans = 6

n = 126 , ans = 11

n = 30 , si = 1, ei = 30

1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30

ei si

mid

mid = 15, 15*15 > 30

mid = 7, 7*7 > 30

mid = 3, 3*3 < 30

mid = 5, 5*5 < 30

mid = 6 6*6 > 30

psudo code

```
si = 1, ei = n;
while (si <= ei){
    mid = (si + ei)/2;
    if (mid*mid == n){
        return mid;
    } else if (mid*mid > n){
        ei = mid-1;
    } else if (mid*mid < n){
        si = mid+1;
    }
}
return ei;
```

Code

```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int ans = squareRoot(n);
    System.out.println(ans);
}
public static int squareRoot(int n) {
    int si = 1;
    int ei = n;
    while (si <= ei) {
        int mid = (si + ei) / 2;
        if ( mid * mid == n ) {
            return mid;
        } else if ( mid * mid < n ) {
            si = mid + 1;
        } else if ( mid * mid > n ) {
            ei = mid - 1;
        }
    }
    return ei;
}
```

T.C = O(log(n))

Find The Index of Rotation

arr =

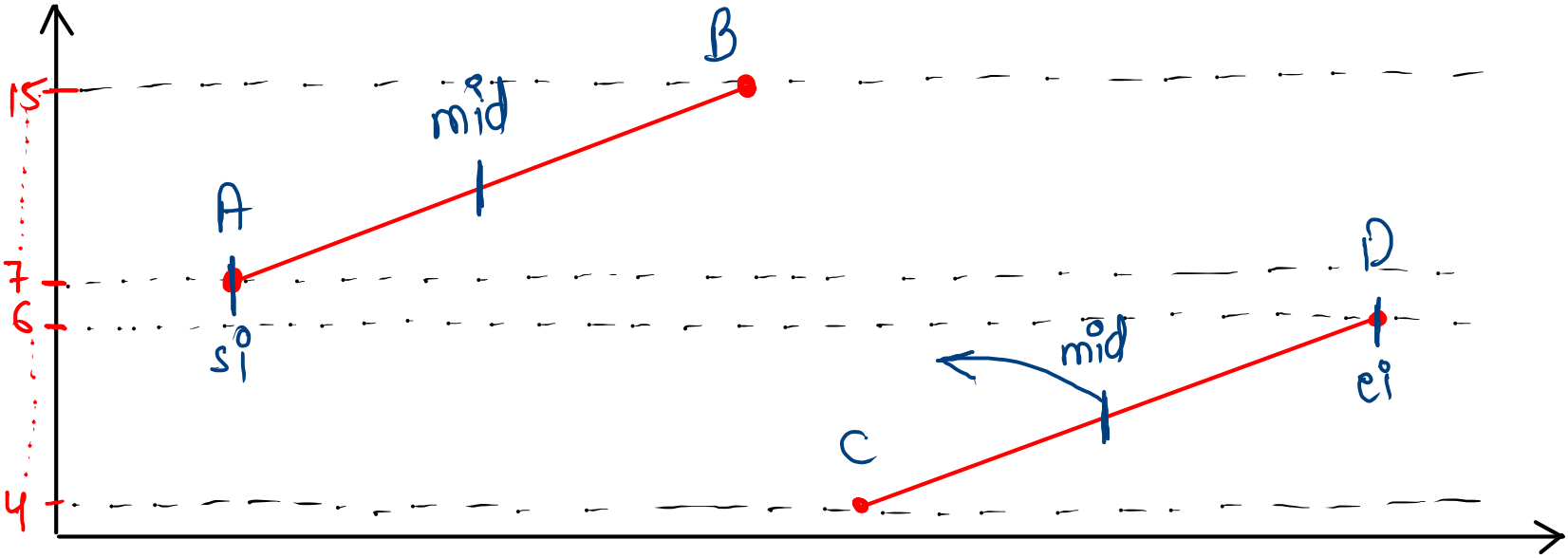
0	1	2	3	4	5	6	7	8	9	10	11
7	8	9	10	11	12	13	14	15	4	5	6

, ans = 8

si ↑ (0)

ei ↑ (11)

target = 9



template

```
int si=0, ei=n-1;
while( si<=ei){
    int mid = (si+ei)/2;
    if( arr[mid] < arr[mid-1] &&
        arr[mid] < arr[mid+1] ){
        return
    } else if ( arr[mid] < arr[ei]){
        ei= mid-1;
    } else if( arr[mid] > arr[si] ){
        si = mid+1
    }
}
return -1;
```

rotation

pseudo code