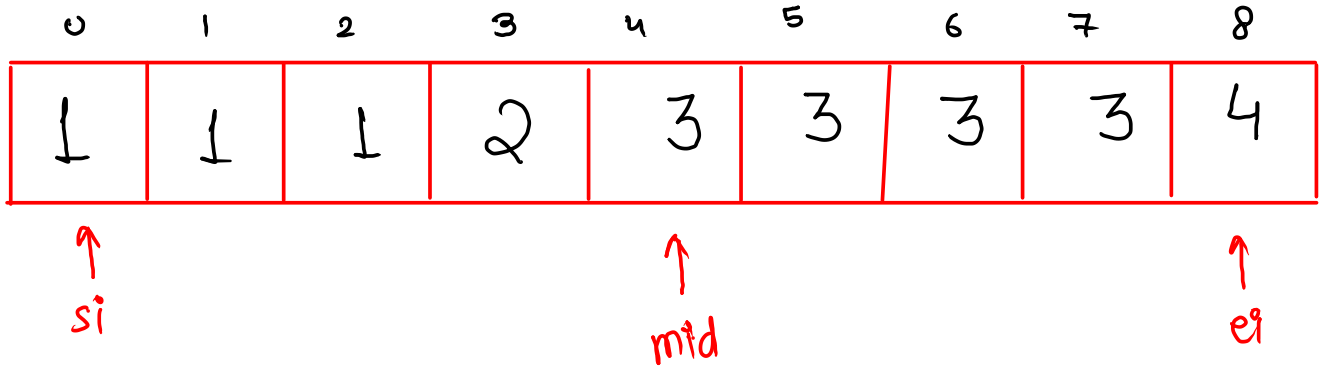
# Find Last Occurrence

$O(\log N)$



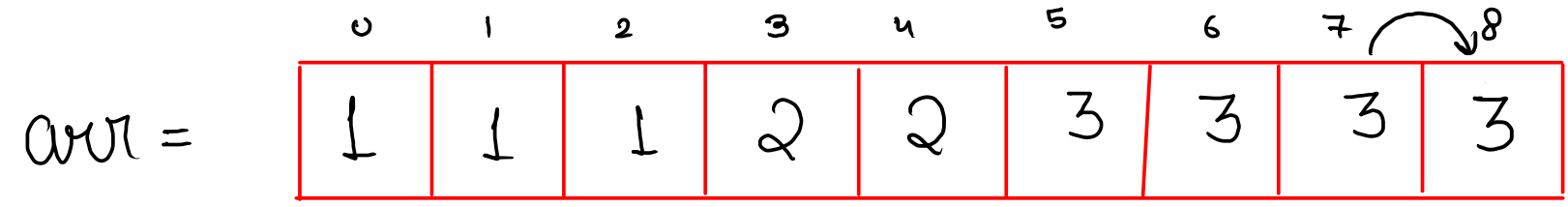

target = 3

dry run

si = 0, ei = 8, mid = 4

si = 5, ei = 8, mid = 6

si = 7, ei = 8, mid = 7




si ei

mid

target = 3

psudo code

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

a

b

c

dry run

si= 0, ei= 8, mid= 4

si= 5, ei= 8, mid= 6

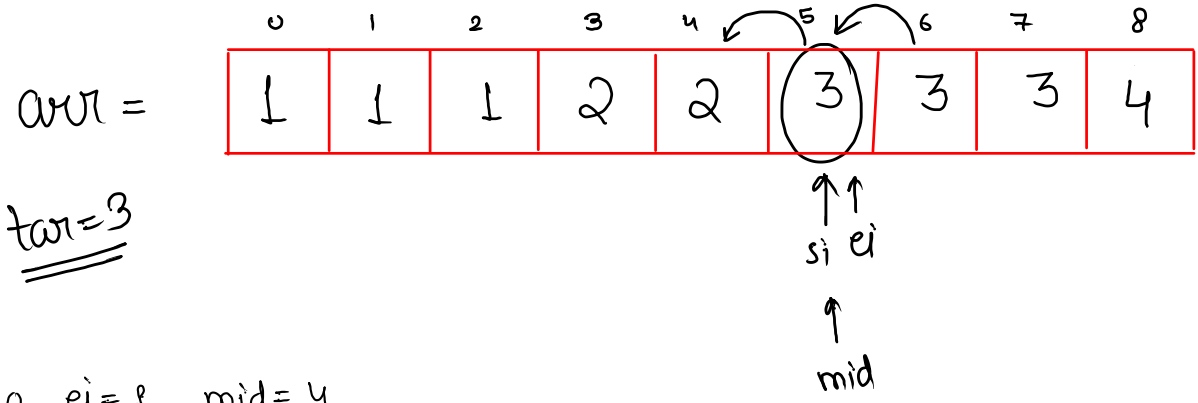
si= 7, ei= 8, mid= 7

return 7

# Binary Search Upper Bound (BSUB)

code

```java
public static int BSUB(int n, int[] arr, int target) {
    int si = 0;
    int ei = n - 1;
    while ( si <= ei ) {
        int mid = (si + ei) / 2;
        if ( arr[mid] == target ) {

            if ( mid < arr.length - 1 && arr[mid] == arr[mid + 1] ) {
                si = mid + 1;
            } else {
                return mid;
            }

        } else if ( arr[mid] < target ) {
            si = mid + 1;
        } else {
            ei = mid - 1;
        }
    }
    return -1;
}
```




si = 0, ei = 8, mid = 4

si = 5, ei = 8, mid = 6

si = 5, ei = 5, mid = 5

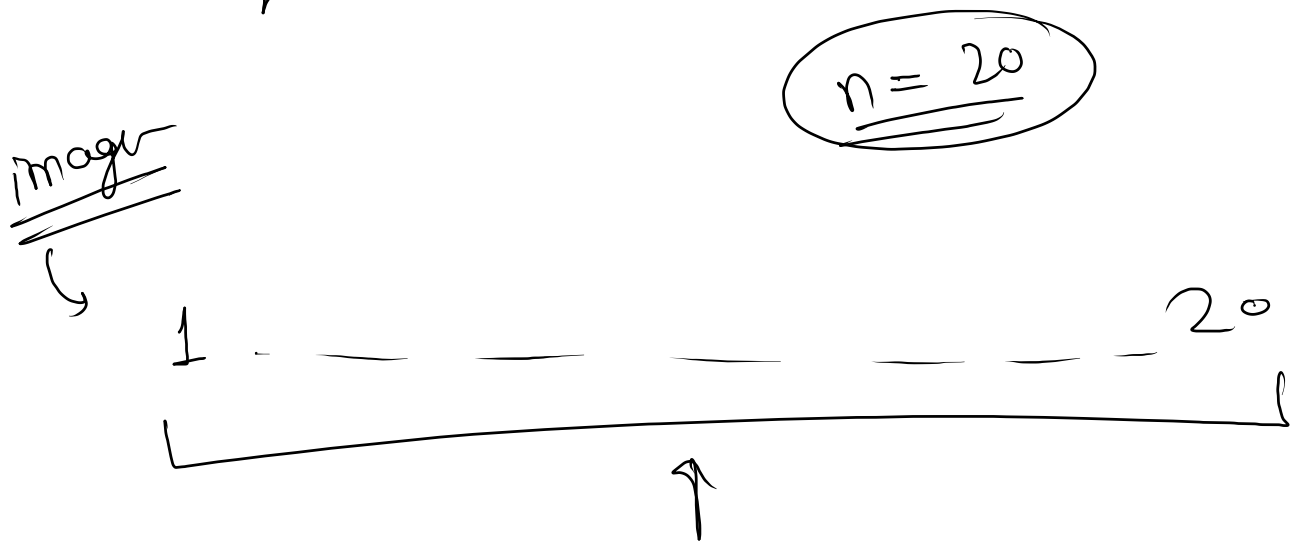
Binary Search
Lower Bound
(BSLB)

```java
public static int BSUB(int n, int[] arr, 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 {
                return mid;
            }

        } else if ( arr[mid] < target ) {
            si = mid + 1;
        } else {
            ei = mid - 1;
        }
    }
    return -1;
}
```

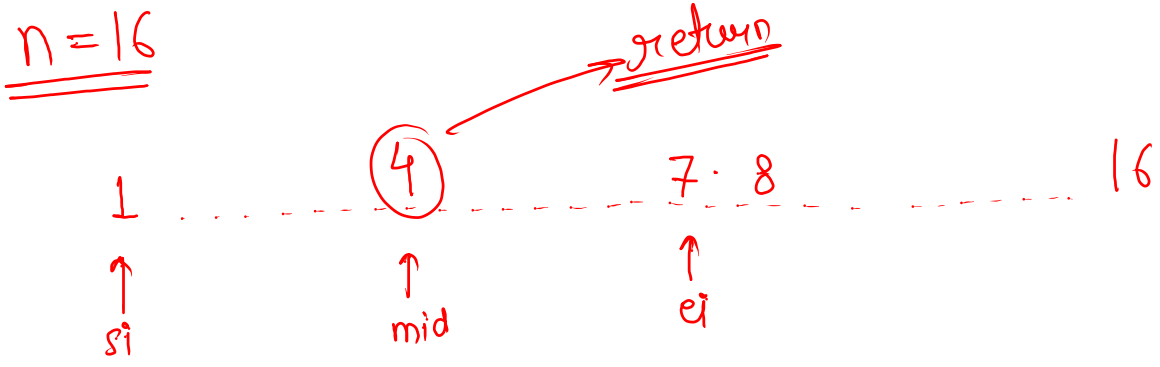
Note:- Binary search need not to applied on given sequence.

means.

it can be applied on any imaginary sequence.



# Find Square Root



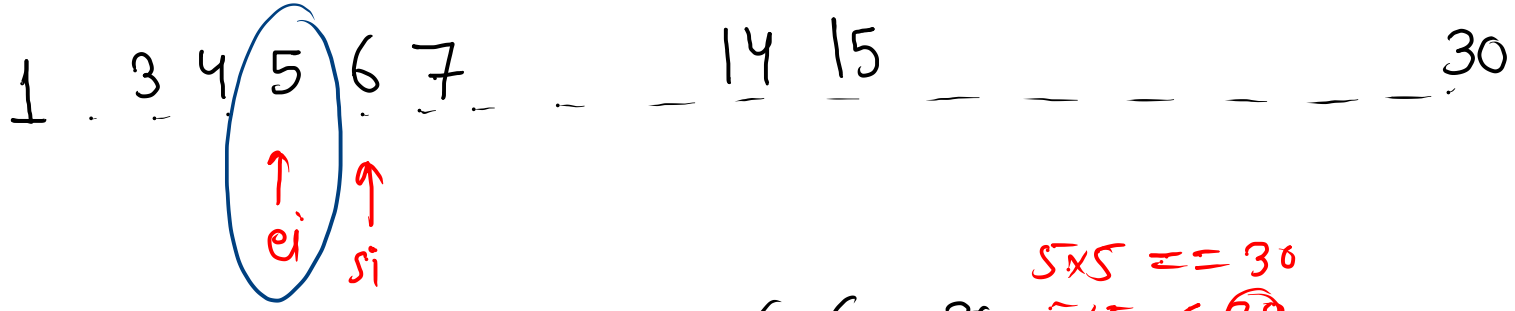

si = 1, ei = 16, mid = 8

si = 1, ei = 7, mid = 4

8 x 8 == 16

4 x 4 == 16

Input $n = 30$




while(si<=ei)

si = 1, ei = 30, mid = 15

si = 1, ei = 14, mid = 7

si = 1, ei = 6, mid = 3

si = 4, ei = 6, mid = 5

si = 6, ei = 6, mid = 6

$6 \times 6 == 30$

$6 \times 6 < 30$

$6 \times 6 > 30$

$5 \times 5 == 30$

$5 \times 5 < 39$

$5 \times 5 > 30$

$15 \times 15 \geq 30$

target

$7 \times 7 > 30$

$3 \times 3 == 30$

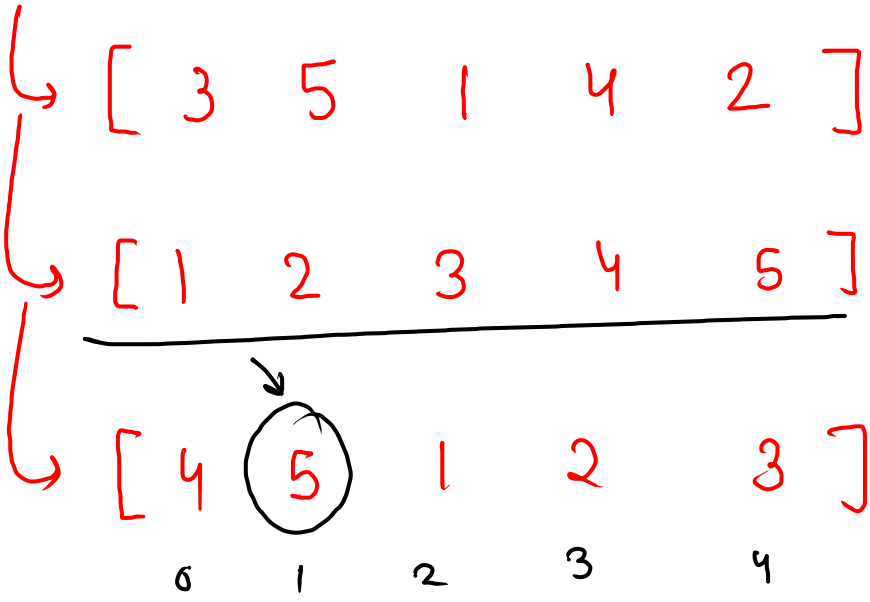
$3 \times 3 > 30$

$3 \times 3 < 30$

Code

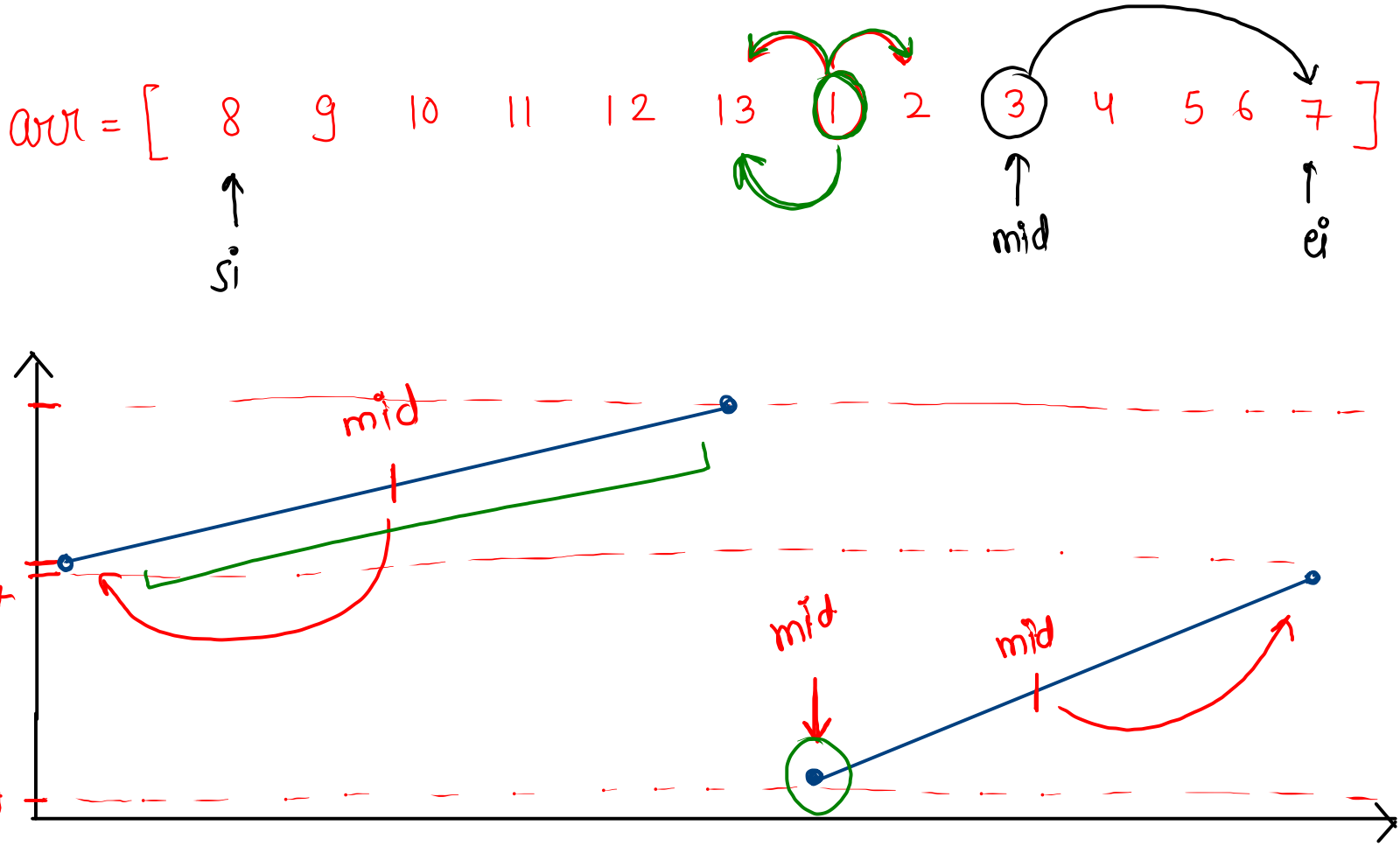
```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    System.out.println(squareRoot(n));
}

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

# Find The Index of Rotation




using B.S

O(log N)


arr = [ 8 9 10 11 12 13 1 2 3 4 5 6 7 ]
si
mid
ei
13
8
7
1
mid
mid
mid