탐색 알고리즘
(Search Algorithm)

탐색 알고리즘

- 배열에서 특정 값을 찾기위한 알고리즘

8	1	5	4	9	7	6	11	2	29	15	81	0

선형 탐색 (Linear Search)

8	1	5	4	9	7	6	11	2	29	15	81	0

선형 탐색 (Linear Search)

8	1	5	4	9	7	6	11	2	29	15	81	0

- 위 배열에서 29을 찾고 싶다면

8	1	5	4	9	7	6	11	2	29	15	81	0

29와 비교

선형 탐색 (Linear Search)

8	1	5	4	9	7	6	11	2	29	15	81	0

- 위 배열에서 29을 찾고 싶다면

8	1	5	4	9	7	6	11	2	29	15	81	0

29와 비교

선형 탐색 (Linear Search)

8	1	5	4	9	7	6	11	2	29	15	81	0

- 위 배열에서 29을 찾고 싶다면

8	1	5	4	9	7	6	11	2	29	15	81	0

29와 비교

선형 탐색 (Linear Search)

8	1	5	4	9	7	6	11	2	29	15	81	0

- 위 배열에서 29을 찾고 싶다면

8	1	5	4	9	7	6	11	2	29	15	81	0

29와 비교

선형 탐색 (Linear Search)

8	1	5	4	9	7	6	11	2	29	15	81	0

- 위 배열에서 29을 찾고 싶다면

8	1	5	4	9	7	6	11	2	29	15	81	0

29와 비교

선형 탐색 (Linear Search)

8	1	5	4	9	7	6	11	2	29	15	81	0

- 위 배열에서 29을 찾고 싶다면

8	1	5	4	9	7	6	11	2	29	15	81	0

29와 비교

선형 탐색 (Linear Search)

8	1	5	4	9	7	6	11	2	29	15	81	0

- 위 배열에서 29을 찾고 싶다면

8	1	5	4	9	7	6	11	2	29	15	81	0

29와 비교

선형 탐색 (Linear Search)

8	1	5	4	9	7	6	11	2	29	15	81	0

- 위 배열에서 29을 찾고 싶다면

8	1	5	4	9	7	6	11	2	29	15	81	0

29와 비교

선형 탐색 (Linear Search)

8	1	5	4	9	7	6	11	2	29	15	81	0

- 위 배열에서 29을 찾고 싶다면

8	1	5	4	9	7	6	11	2	29	15	81	0

29와 비교

선형 탐색 (Linear Search)

8	1	5	4	9	7	6	11	2	29	15	81	0

- 위 배열에서 29을 찾고 싶다면

8	1	5	4	9	7	6	11	2	29	15	81	0

29와 비교

선형 탐색 (Linear Search)

8	1	5	4	9	7	6	11	2	29	15	81	0

- 위 배열에서 29을 찾고 싶다면

8	1	5	4	9	7	6	11	2	29	15	81	0

10번 만에 탐색 완료

선형 탐색 (Linear Search)

8	1	5	4	9	7	6	11	2	29	15	81	0

- 위 배열에서 29을 찾고 싶다면

8	1	5	4	9	7	6	11	2	29	15	81	0

10번 만에 탐색 완료

시간복잡도 : O(N)

문제점

- 배열의 크기가 상당히 클 경우

8	1	5	4	9	7

· · ·

6	11	2	29	15	81	0

1억번째

1억번 비교 필요

이분탐색 (Binary Search)

이분탐색 (Binary Search)

- 큰 배열에서 특정 값을 빠르게 탐색하기 위한 방법

이분탐색 (Binary Search)

- 큰 배열에서 특정 값을 빠르게 탐색하기 위한 방법
- 이분탐색의 필수조건 : 정렬된 배열

이분탐색 (Binary Search)

- 큰 배열에서 특정 값을 빠르게 탐색하기 위한 방법
- 이분탐색의 필수조건 : 정렬된 배열

0	1	2	4	5	6	7	8	9	11	15	29	81

이분탐색 (Binary Search)

- 큰 배열에서 특정 값을 빠르게 탐색하기 위한 방법
- 이분탐색의 필수조건 : 정렬된 배열

0	1	2	4	5	6	7	8	9	11	15	29	81

29 찾기

이분탐색

시작						중간						끝
0	1	2	4	5	6	7	8	9	11	15	29	81

중간 값(7)이 찾으려는 값(29)인지 확인

이분탐색

시작						중간						끝
0	1	2	4	5	6	7	8	9	11	15	29	81

- 중간을 기준으로 2개의 그룹 만들기

0	1	2	4	5	6	7	8	9	11	15	29	81

이분탐색

시작 중간 끝

0	1	2	4	5	6	7	8	9	11	15	29	81

- 중간을 기준으로 2개의 그룹 만들기

0	1	2	4	5	6	7	8	9	11	15	29	81

7보다 작은 그룹 기준 7보다 큰 그룹

이분탐색

시작 중간 끝

0	1	2	4	5	6	7	8	9	11	15	29	81

- 중간을 기준으로 2개의 그룹 만들기

0	1	2	4	5	6

7보다 작은 그룹

29가 없는 그룹
중간값(7) > 찾으려는값(29)

7

기준

8	9	11	15	29	81

7보다 큰 그룹

29가 있는 그룹
중간값(7) < 찾으려는값(29)

이분탐색

중간 값(11)이 찾으려는 값(29)인지 확인

이분탐색

- 중간을 기준으로 2개의 그룹 만들기

이분탐색

- 중간을 기준으로 2개의 그룹 만들기

이분탐색

중간 값(29)이 찾으려는 값(29)인지 확인

이분탐색

3번만에 탐색 완료

이분탐색

3번만에 탐색 완료

시간복잡도 : O(log N)

문제점

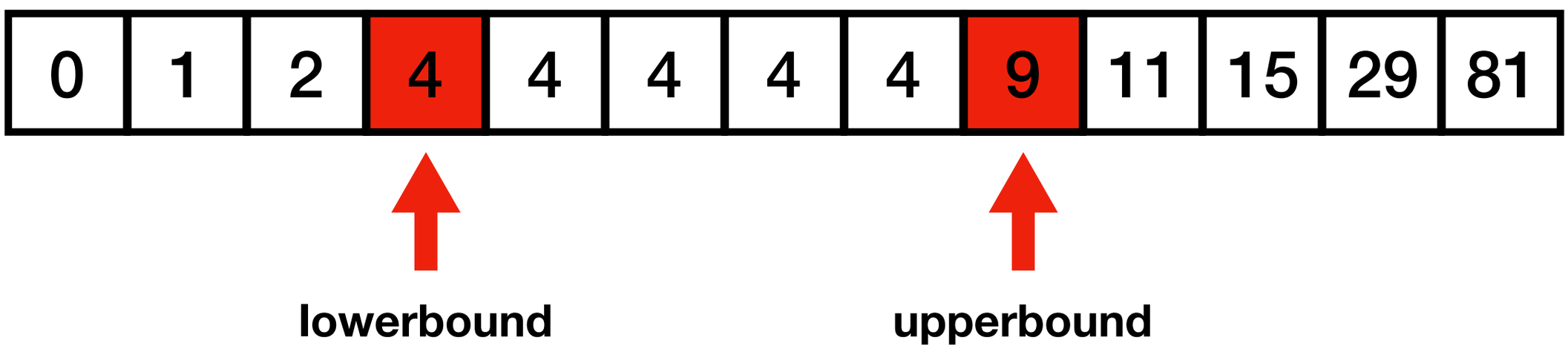
- 같은 값을 가진 원소가 여러 개 존재할 경우

0	1	2	4	4	4	4	4	9	11	15	29	81

어떤 값이 결과로 나올지 모름

LowerBound, UpperBound

찾고자 하는 값 : 4



- lowerbound : 찾고자 하는 값(4) 이상(<=)의 값들 중 처음 등장하는 원소를 탐색

- upperbound : 찾고자 하는 값(4)을 초과(<)하는 값들 중 가장 처음 등장하는 원소를 탐색

LowerBound

시작						중간							끝
0	1	2	4	4	4	4	4	9	11	15	29	81	X

중간 값(4)과 찾으려는 값(4) 비교

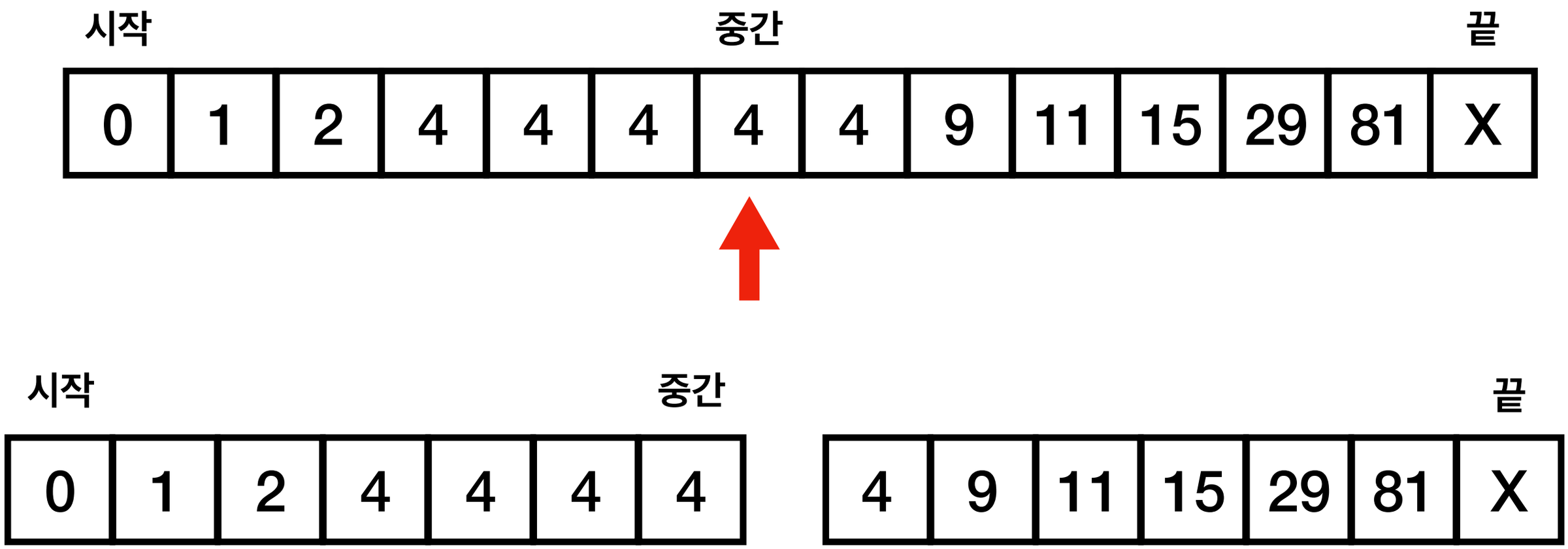
LowerBound

시작						중간							끝
0	1	2	4	4	4	4	4	9	11	15	29	81	X

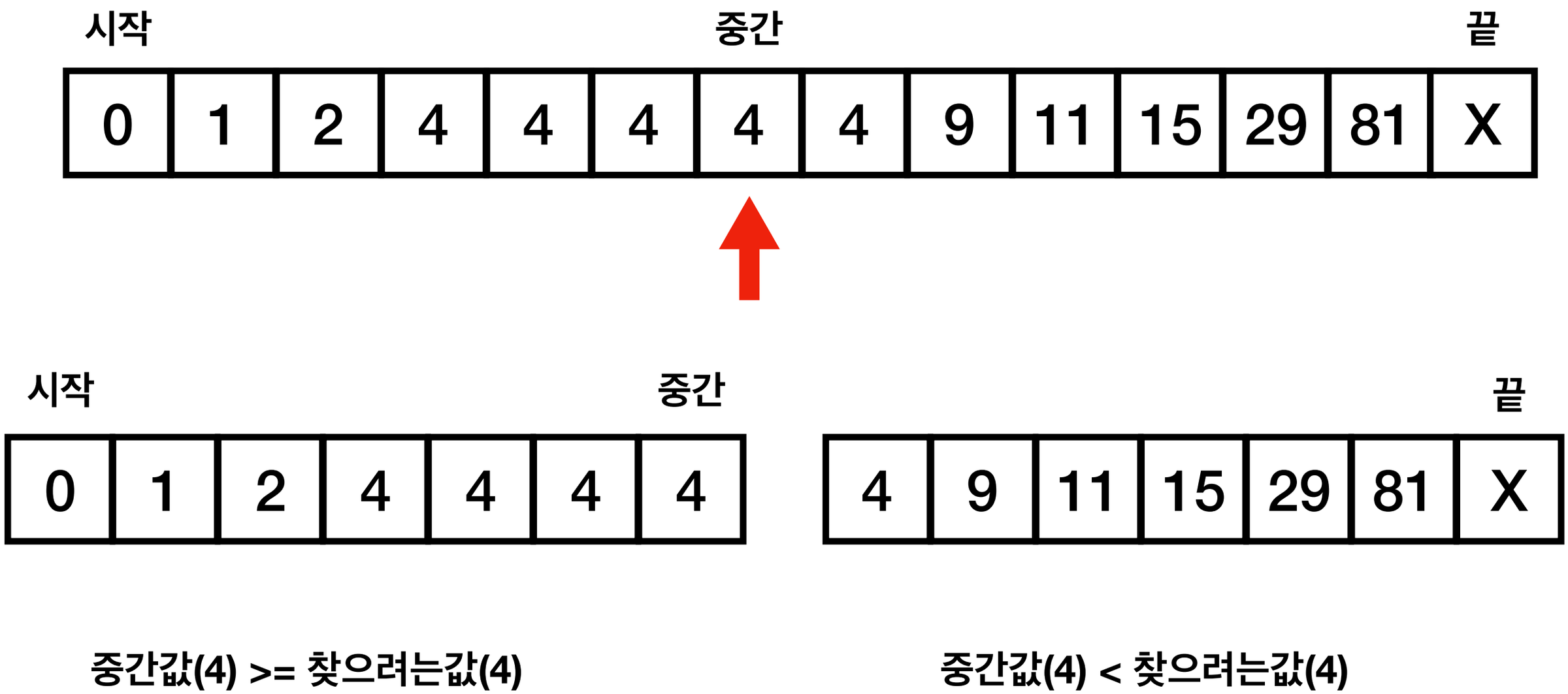
중간 값(4)과 찾으려는 값(4) 비교
(같음)

- **비교 결과가 가지는 의미**
  - **중간 값 < 찾으려는 값** : 중간 값보다 작거나 같은(왼쪽) 원소들 중에는 lowerbound인 원소가 존재할 수 없다.
  - **중간 값 >= 찾으려는 값** : 중간 값보다 작거나 같은(왼쪽) 원소들 중에 lowerbound가 있다.

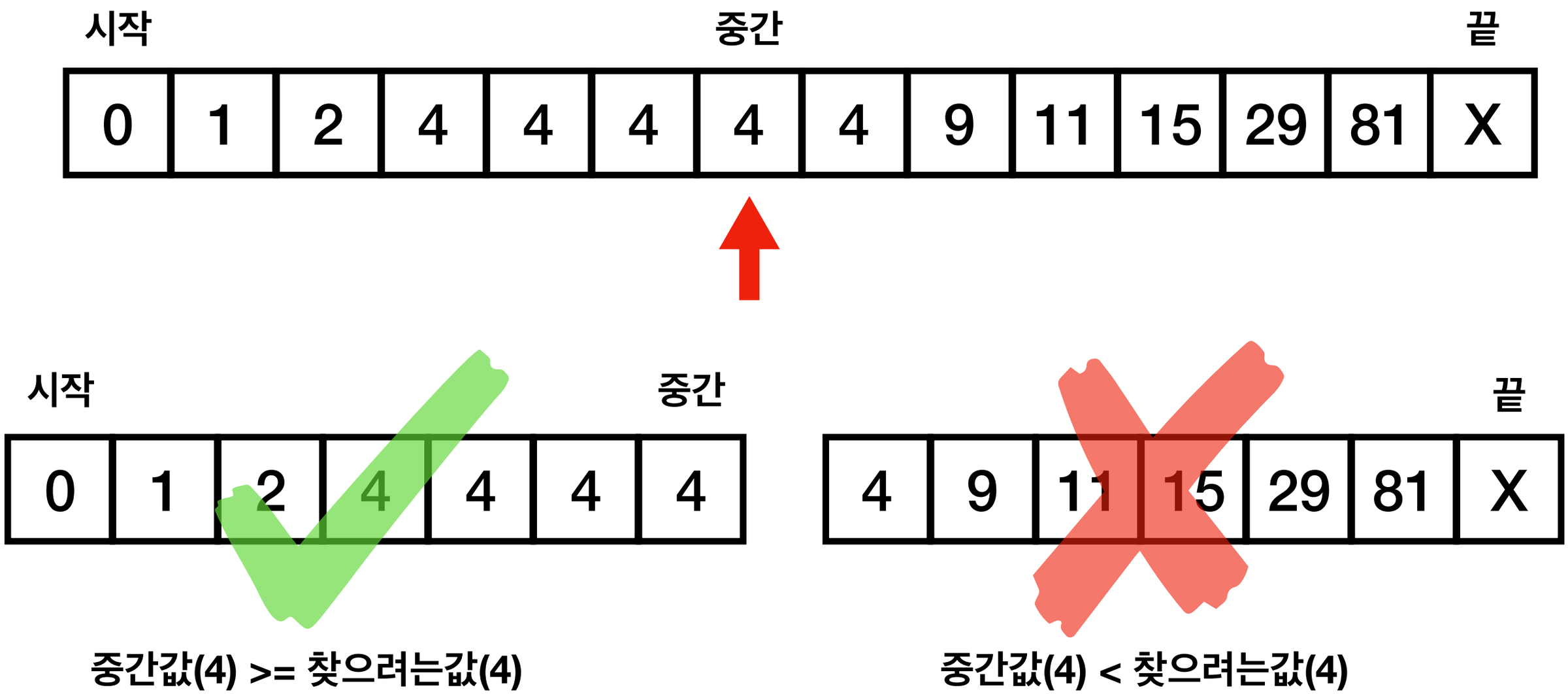
LowerBound



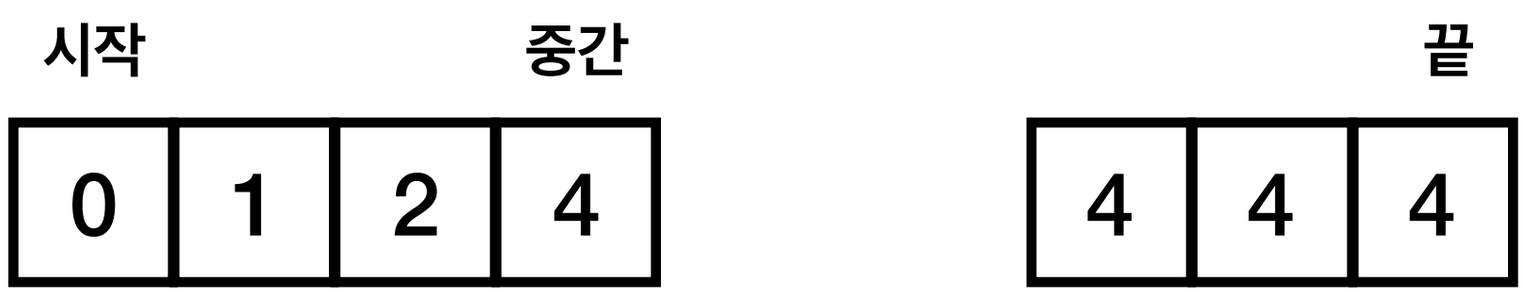
LowerBound


시작
중간
끝
0 1 2 4 4 4 4 4 9 11 15 29 81 X
시작
중간
0 1 2 4 4 4 4
끝
4 9 11 15 29 81 X
중간값(4) >= 찾으려는값(4)
중간값(4) < 찾으려는값(4)

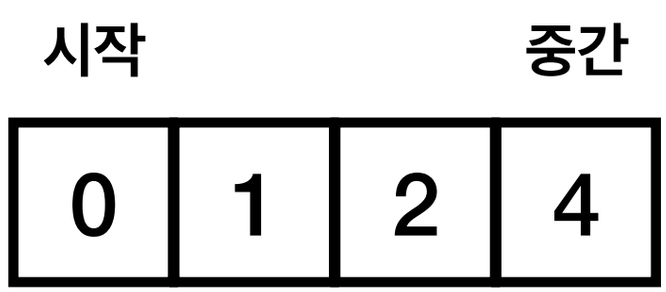
LowerBound

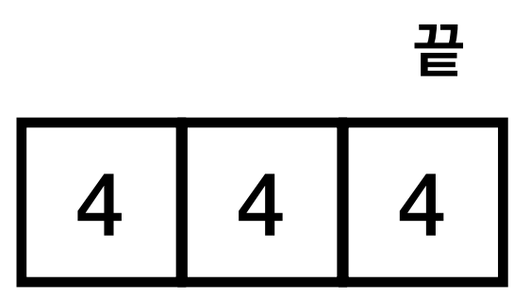

시작
중간
끝
0 1 2 4 4 4 4 4 9 11 15 29 81 X
시작
중간
0 1 2 4 4 4 4
중간값(4) >= 찾으려는값(4)
끝
4 9 11 15 29 81 X
중간값(4) < 찾으려는값(4)

LowerBound



LowerBound

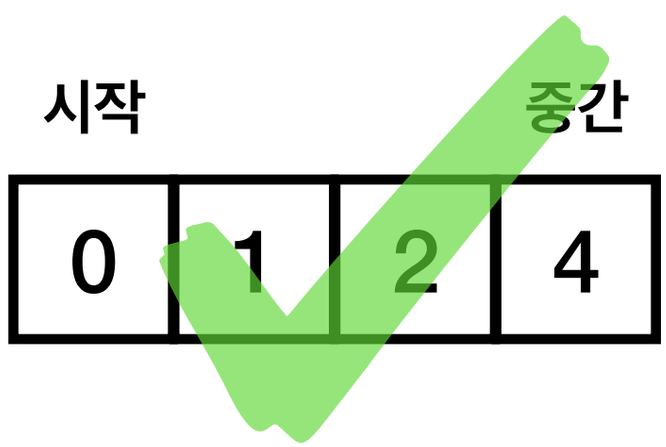




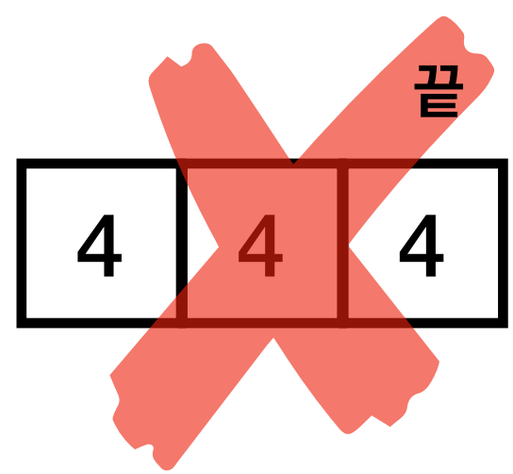
중간값(4) >= 찾으려는값(4)

중간값(4) < 찾으려는값(4)

LowerBound

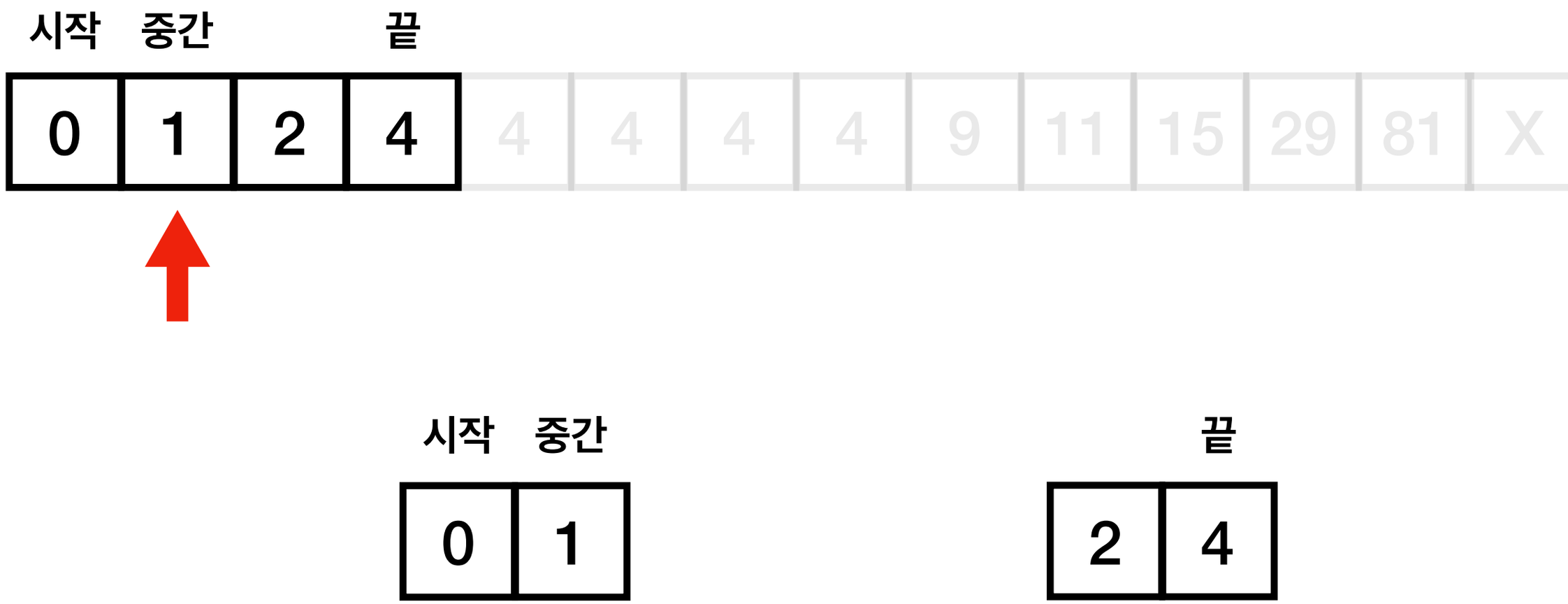


중간값(4) >= 찾으려는값(4)



중간값(4) < 찾으려는값(4)

LowerBound



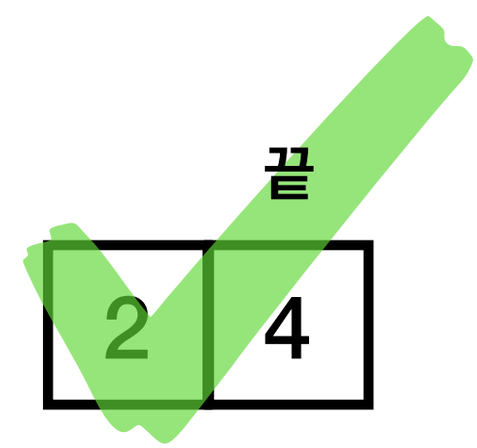
LowerBound

중간값(1) >= 찾으려는값(4)

중간값(1) < 찾으려는값(4)

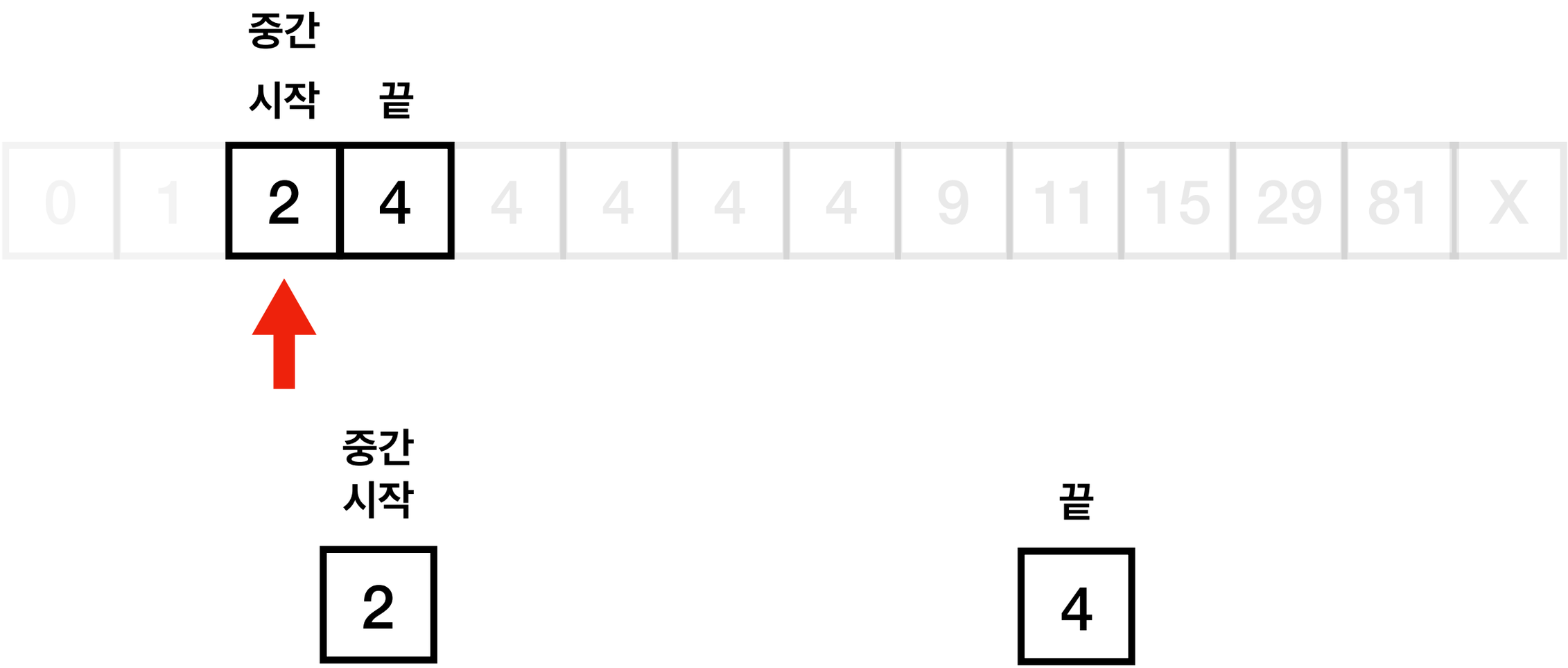
LowerBound

중간값(1) >= 찾으려는값(4)

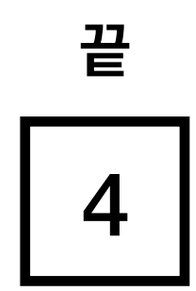


중간값(1) < 찾으려는값(4)

LowerBound



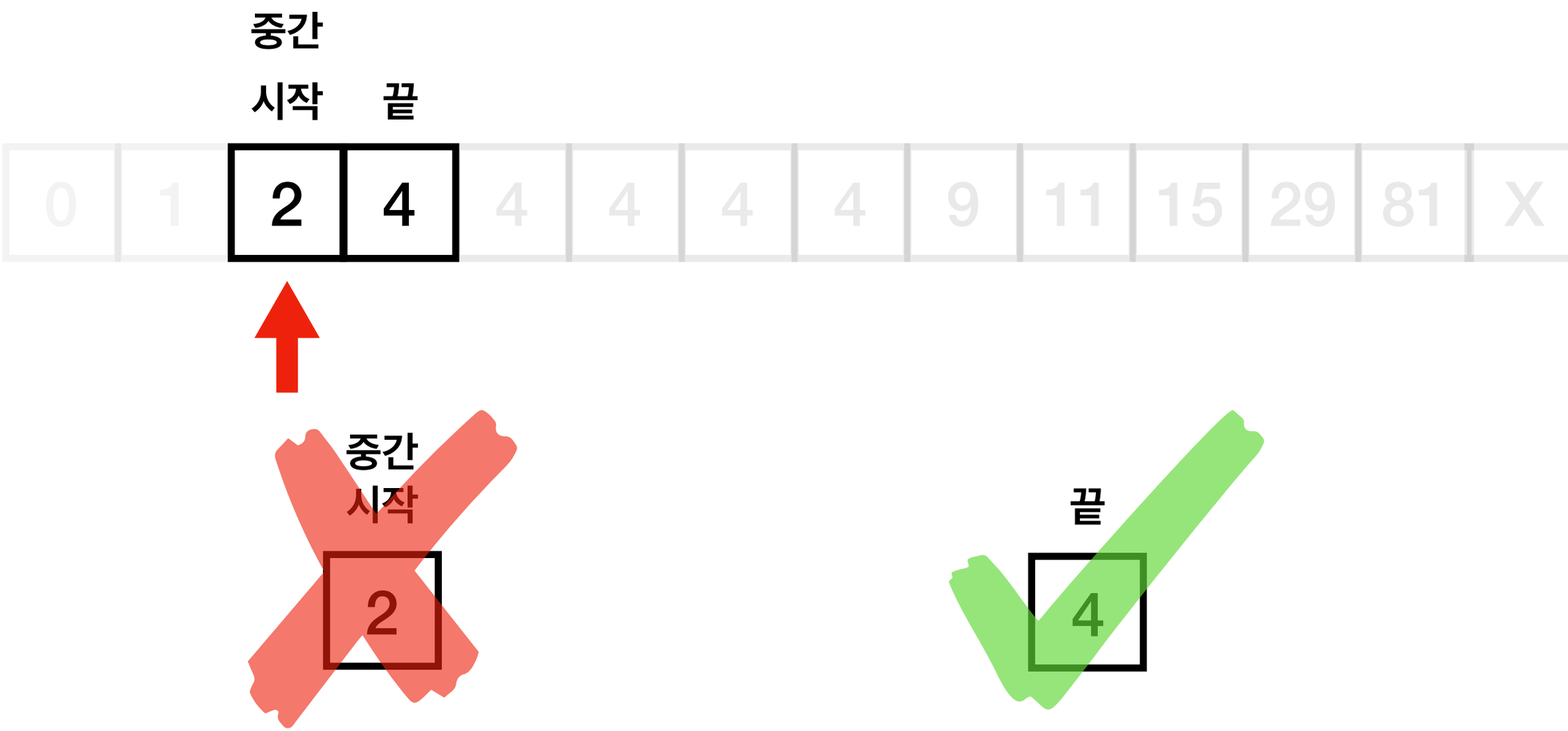
LowerBound



중간값(2) >= 찾으려는값(4)

중간값(2) < 찾으려는값(4)

LowerBound



중간값(2) >= 찾으려는값(4)

중간값(2) < 찾으려는값(4)

LowerBound

중간

시작

끝

0	1	2	4	4	4	4	4	9	11	15	29	81	X

LowerBound

중간

시작

끝

0	1	2	4	4	4	4	4	9	11	15	29	81	X

LowerBound

시간복잡도 : O(log N)

UpperBound

시작						중간							끝
0	1	2	4	4	4	4	4	9	11	15	29	81	X

중간 값(4)과 찾으려는 값(4) 비교

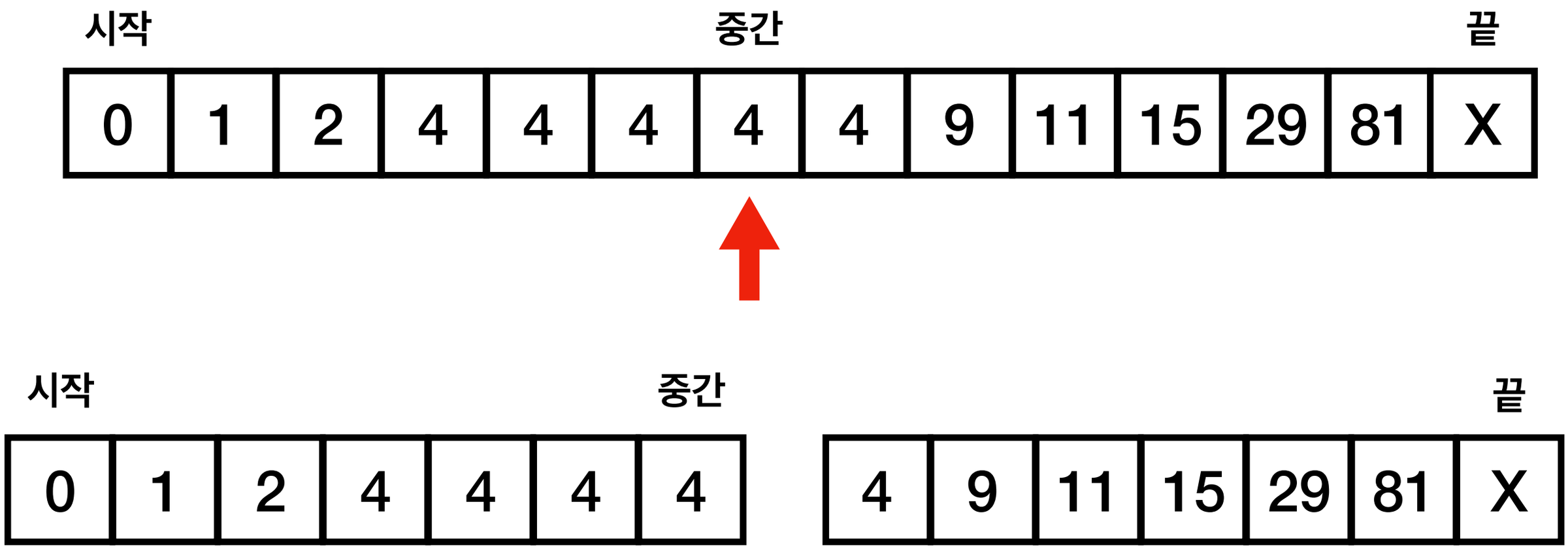
UpperBound

시작						중간							끝
0	1	2	4	4	4	4	4	9	11	15	29	81	X

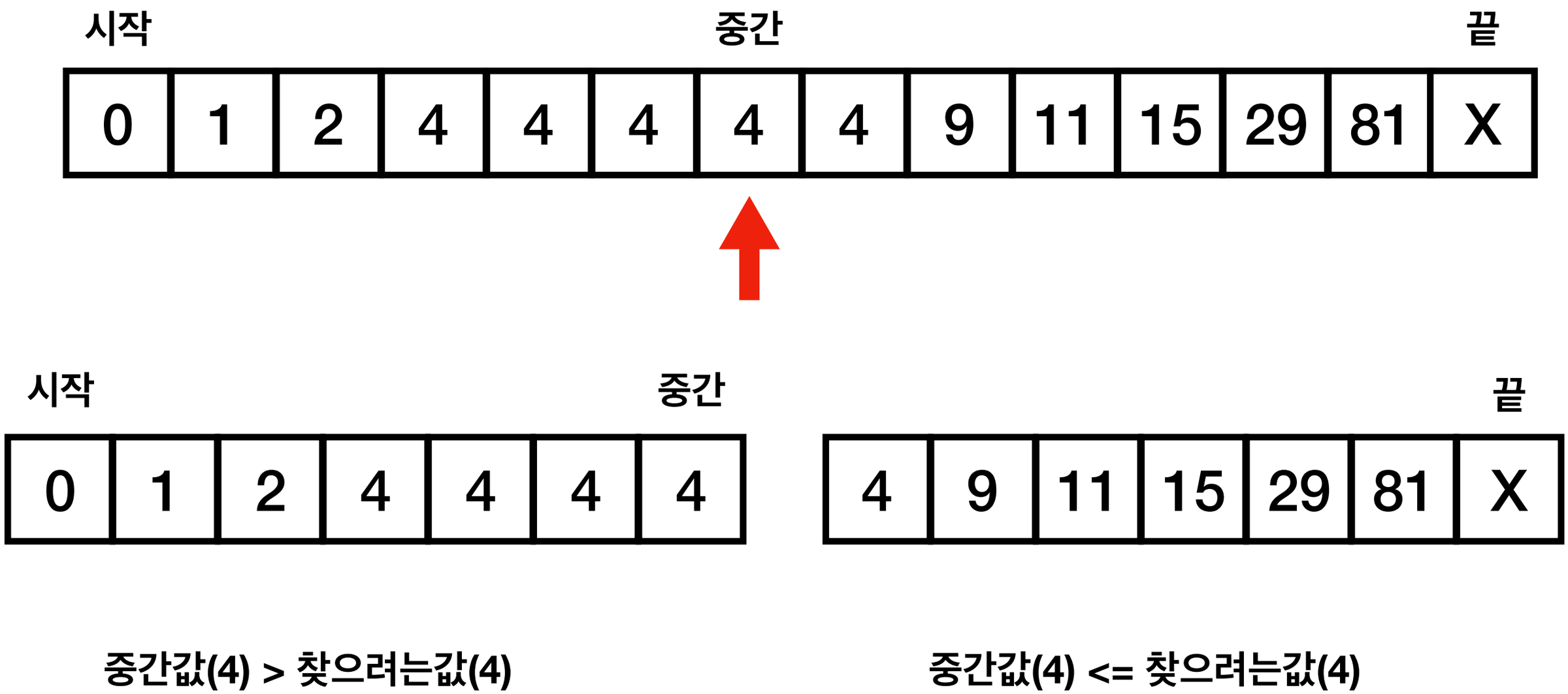
중간 값(4)과 찾으려는 값(4) 비교
(같음)

- **비교 결과가 가지는 의미**
  - **중간 값 <= 찾으려는 값** : 중간 값보다 작거나 같은(왼쪽) 원소들 중에는 upperbound 인 원소가 존재할 수 없다.
  - **중간 값 > 찾으려는 값** : 중간 값보다 작거나 같은(왼쪽) 원소들 중에 upperbound가 있다.

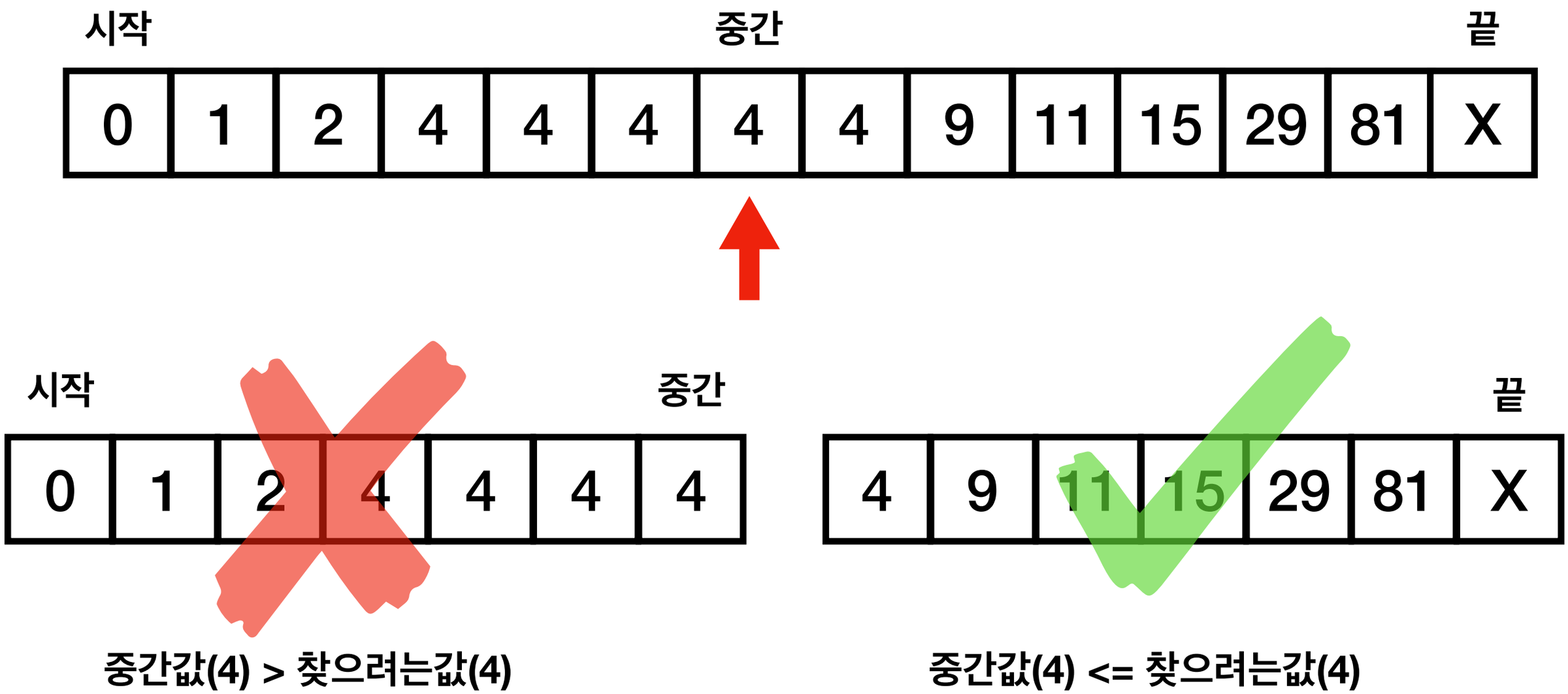
UpperBound



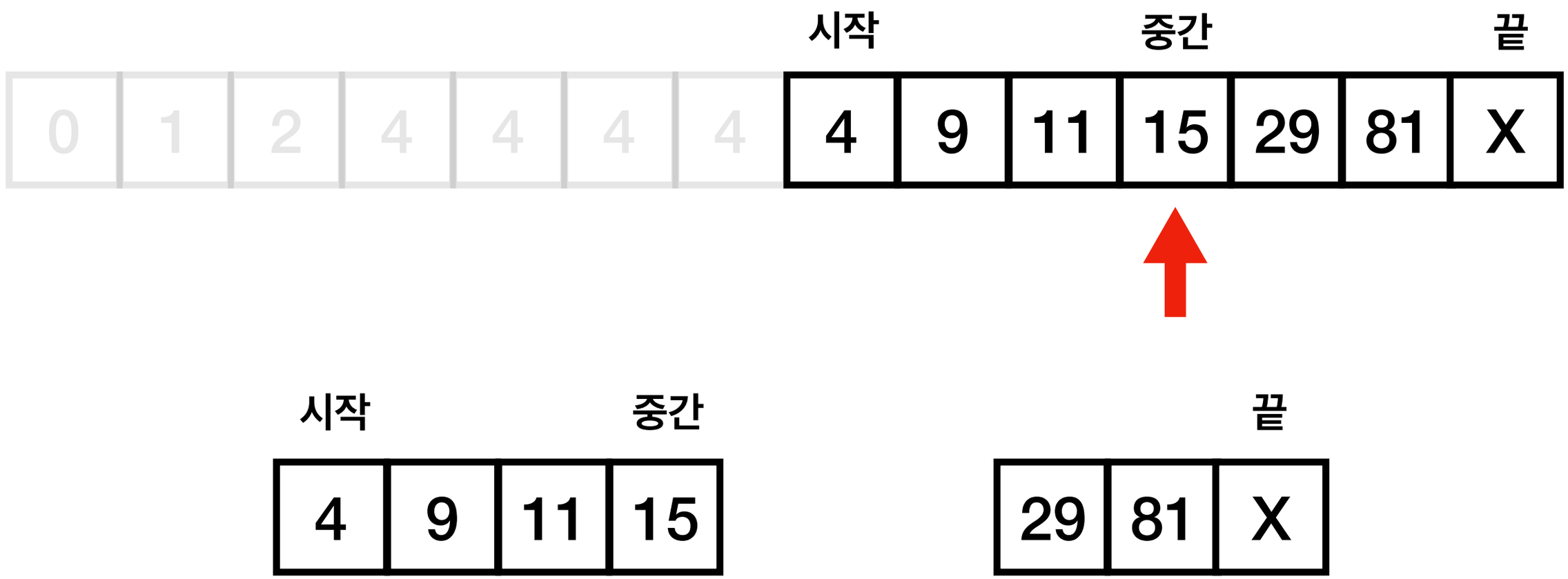
UpperBound


시작
중간
끝
0 1 2 4 4 4 4 4 9 11 15 29 81 X
시작
중간
0 1 2 4 4 4 4
끝
4 9 11 15 29 81 X
중간값(4) > 찾으려는값(4)
중간값(4) <= 찾으려는값(4)

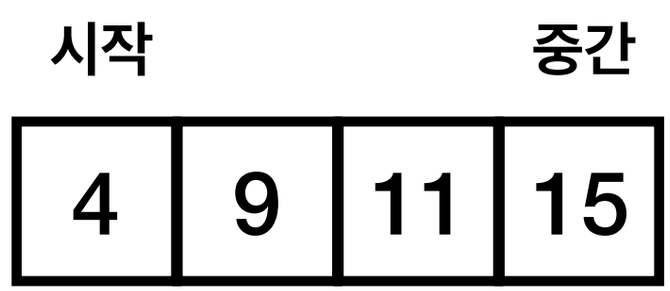
UpperBound


시작
중간
끝
0 1 2 4 4 4 4 4 9 11 15 29 81 X
시작
중간
0 1 2 4 4 4 4
끝
4 9 11 15 29 81 X
중간값(4) > 찾으려는값(4)
중간값(4) <= 찾으려는값(4)

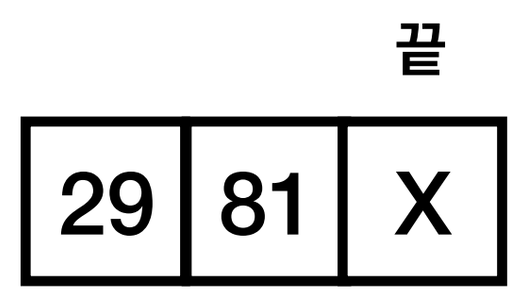
UpperBound



UpperBound

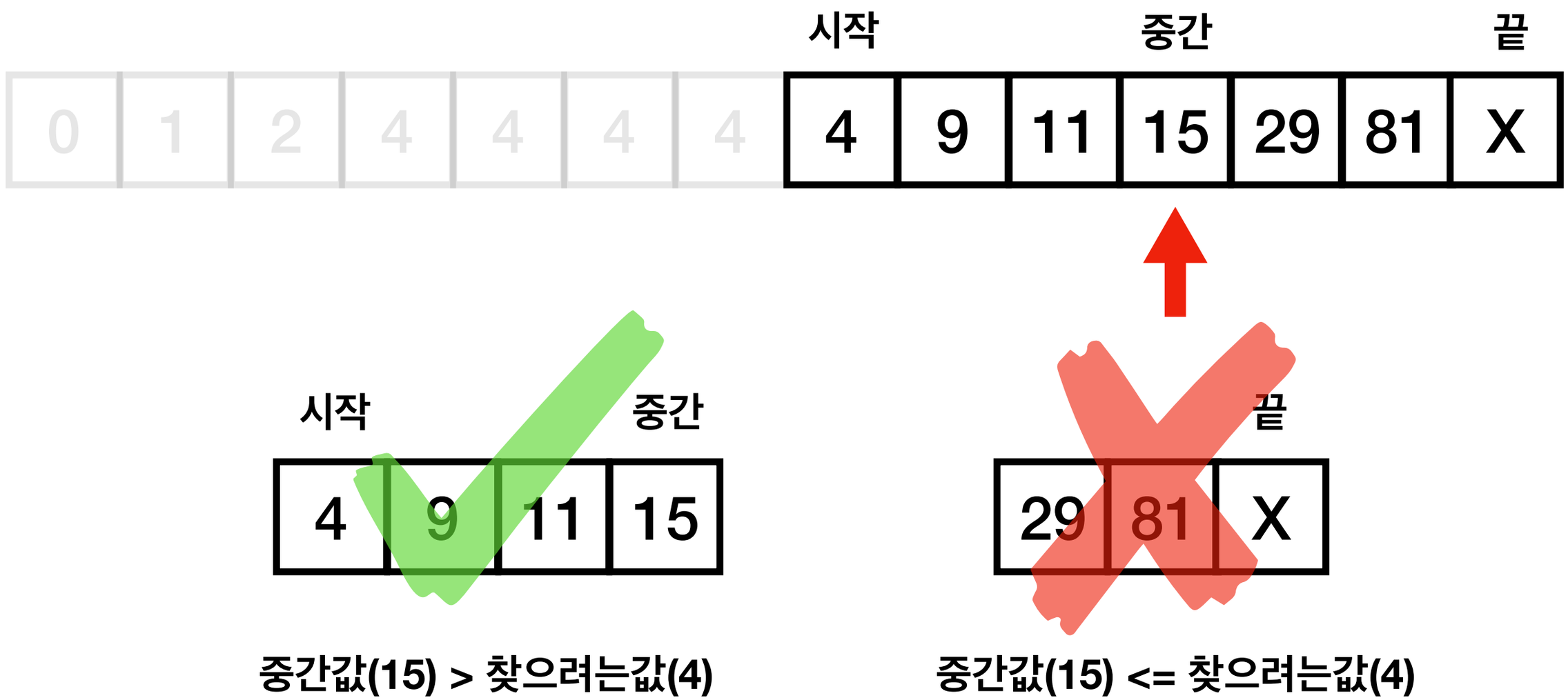


중간값(15) > 찾으려는값(4)

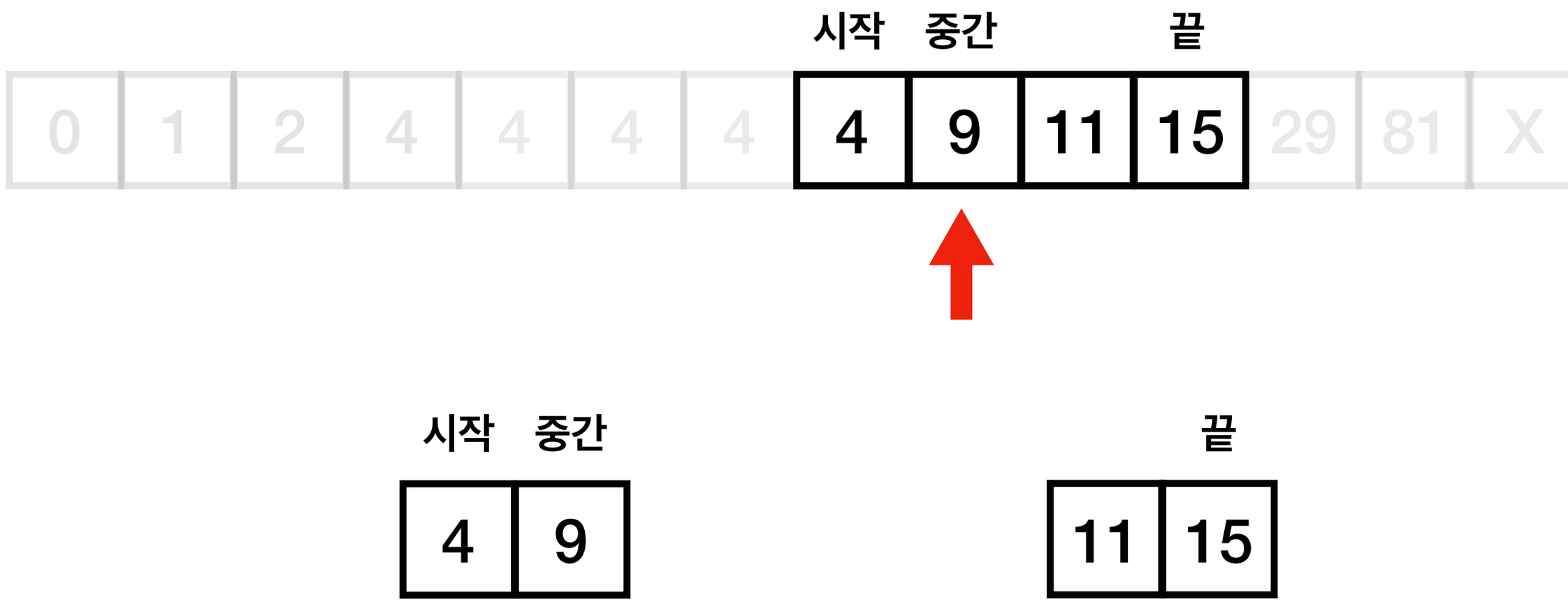


중간값(15) <= 찾으려는값(4)

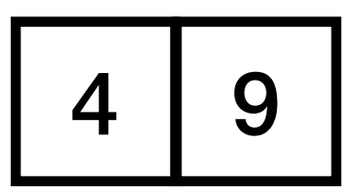
UpperBound

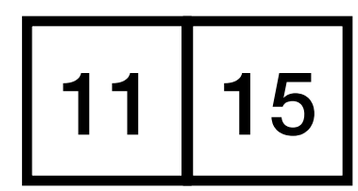

시작
중간
끝
0
1
2
4
4
4
4
4
9
11
15
29
81
X
시작
중간
4
9
11
15
중간값(15) > 찾으려는값(4)
끝
29
81
X
중간값(15) <= 찾으려는값(4)

UpperBound



UpperBound



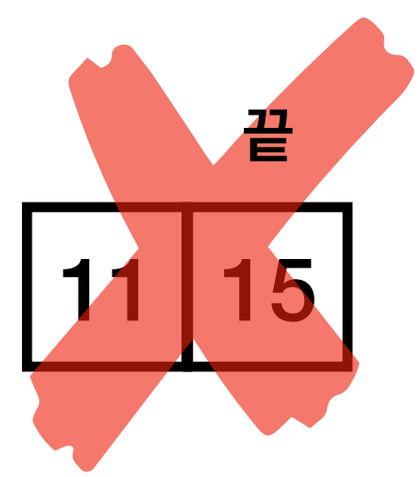


중간값(9) > 찾으려는값(4)

중간값(9) <= 찾으려는값(4)

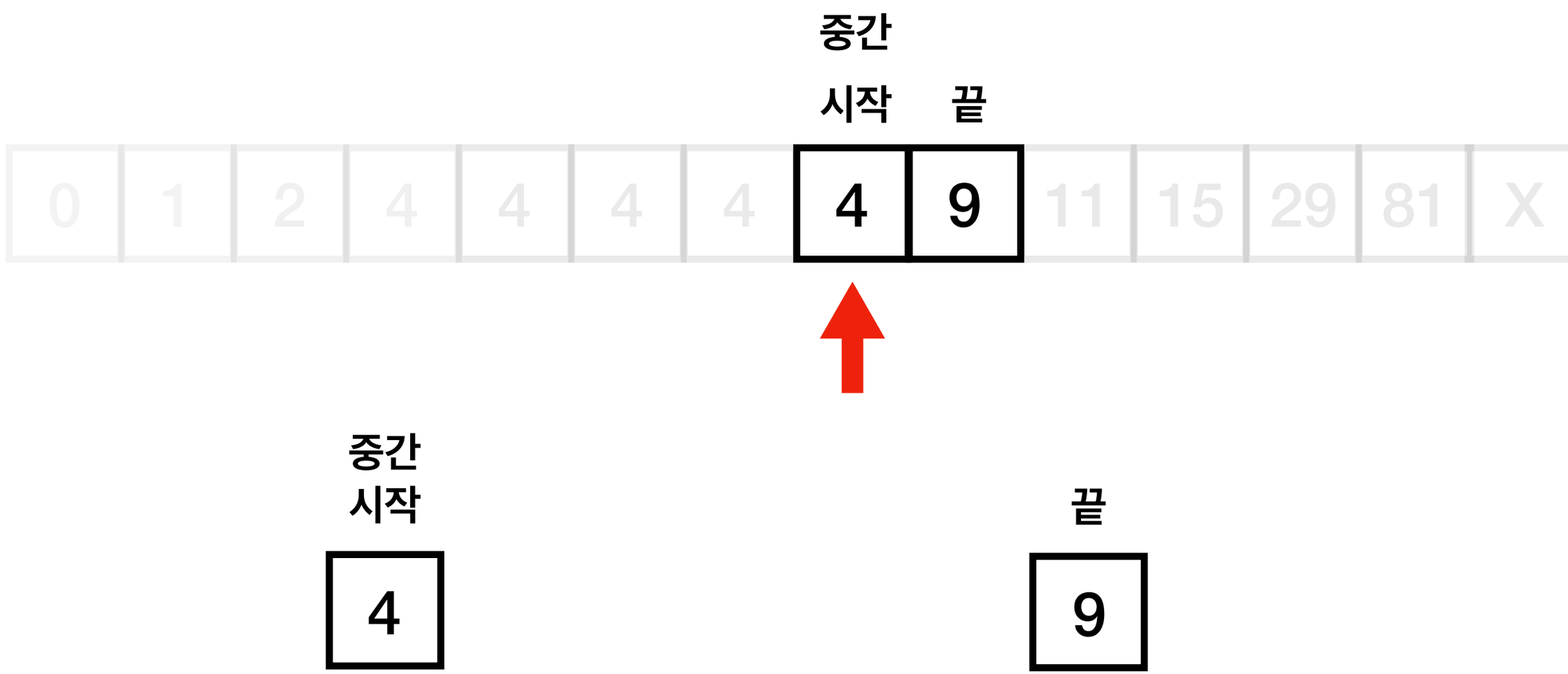
UpperBound

중간값(9) > 찾으려는값(4)

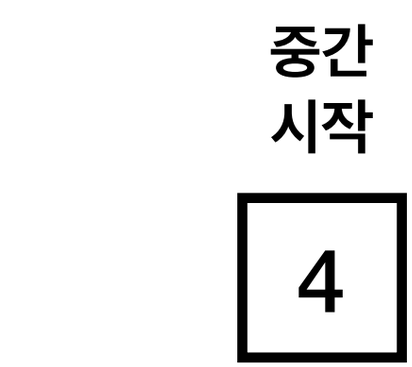


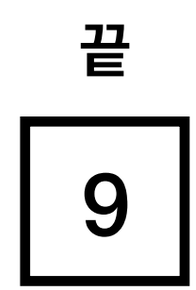
중간값(9) <= 찾으려는값(4)

UpperBound



UpperBound

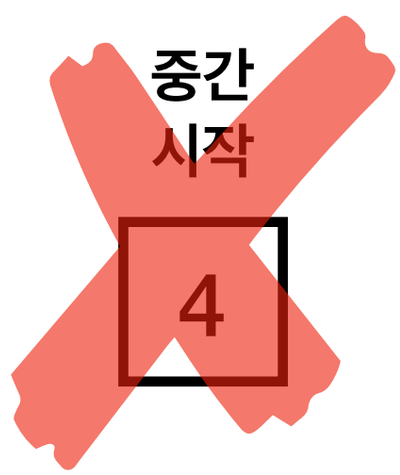




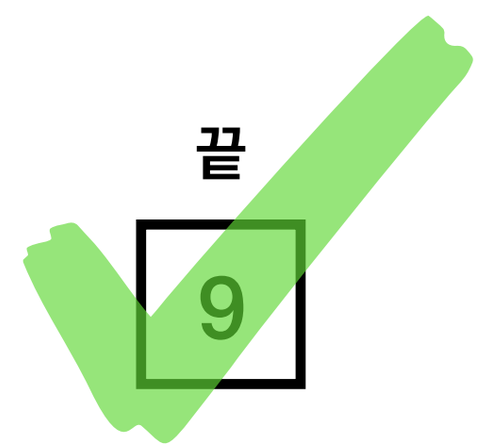
중간값(4) > 찾으려는값(4)

중간값(4) <= 찾으려는값(4)

UpperBound



중간값(4) > 찾으려는값(4)



중간값(4) <= 찾으려는값(4)

UpperBound

중간

시작

끝

0	1	2	4	4	4	4	4	9	11	15	29	81	X

UpperBound

시간복잡도 : O(log N)

LowerBound, UpperBound

LowerBound, UpperBound

- 초기 배열의 끝점이 X인 이유

0	1	2	4	4	4	4	4	9	11	15	29	81	X

LowerBound, UpperBound

- 초기 배열의 끝점이 X인 이유

0	1	2	4	4	4	4	4	9	11	15	29	81	X

lowerbound 또는 upperbound가 존재하지 않을 경우 탐색되는 위치

LowerBound, UpperBound

- 끝 지점이 X인 이유

0	1	2	4	4	4	4	4	9	11	15	29	81	X

lowerbound 또는 upperbound가 존재하지 않을 경우 탐색되는 위치

- 주어진 배열에서 특정 값 구간 구하기

0	1	2	4	4	4	4	4	9	11	15	29	81	X

LowerBound, UpperBound

- 끝 지점이 X인 이유

0	1	2	4	4	4	4	4	9	11	15	29	81	X

lowerbound 또는 upperbound가 존재하지 않을 경우 탐색되는 위치

- 주어진 배열에서 특정 값 구간 구하기

0	1	2	4	4	4	4	4	9	11	15	29	81	X

lowerbound **upperbound**

STL
LowerBound, UpperBound

- lowerbound

```
template <class ForwardIterator, class T>
  ForwardIterator lower_bound (ForwardIterator first, ForwardIterator last, const T& val)
```

```
low_it = lower_bound (v.begin(), v.end(), 20);
```

- upperbound

```
template <class ForwardIterator, class T>
  ForwardIterator upper_bound (ForwardIterator first, ForwardIterator last, const T& val)
```

```
up_it = upper_bound (v.begin(), v.end(), 20);
```

연습문제

https://www.acmicpc.net/problem/1920