SORT (1)

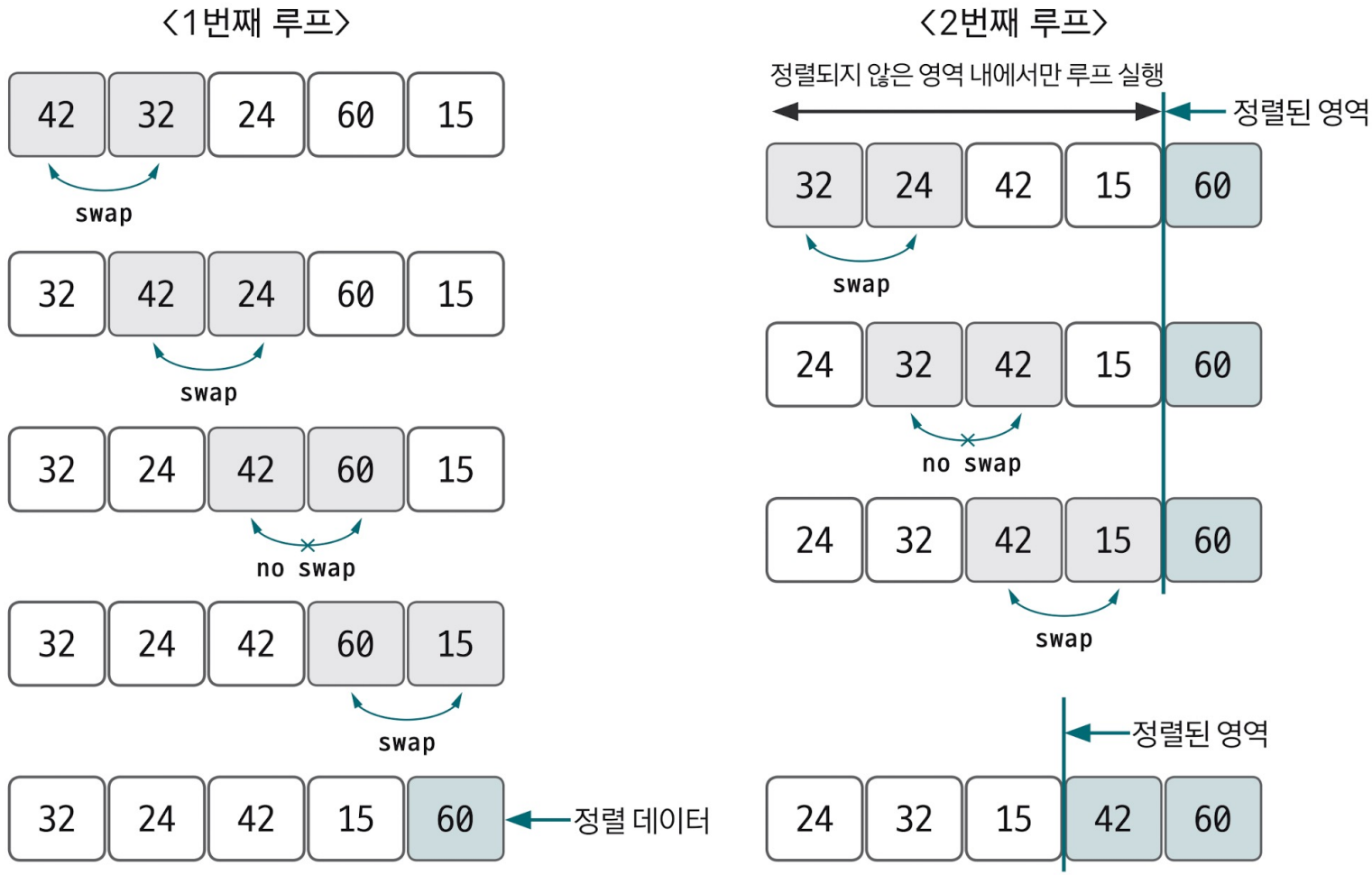
Algorithm

01

Bubble Sort

01. CONTENTS

버블정렬



버블 정렬 수행 방식

시간복잡도 : O(n^2)

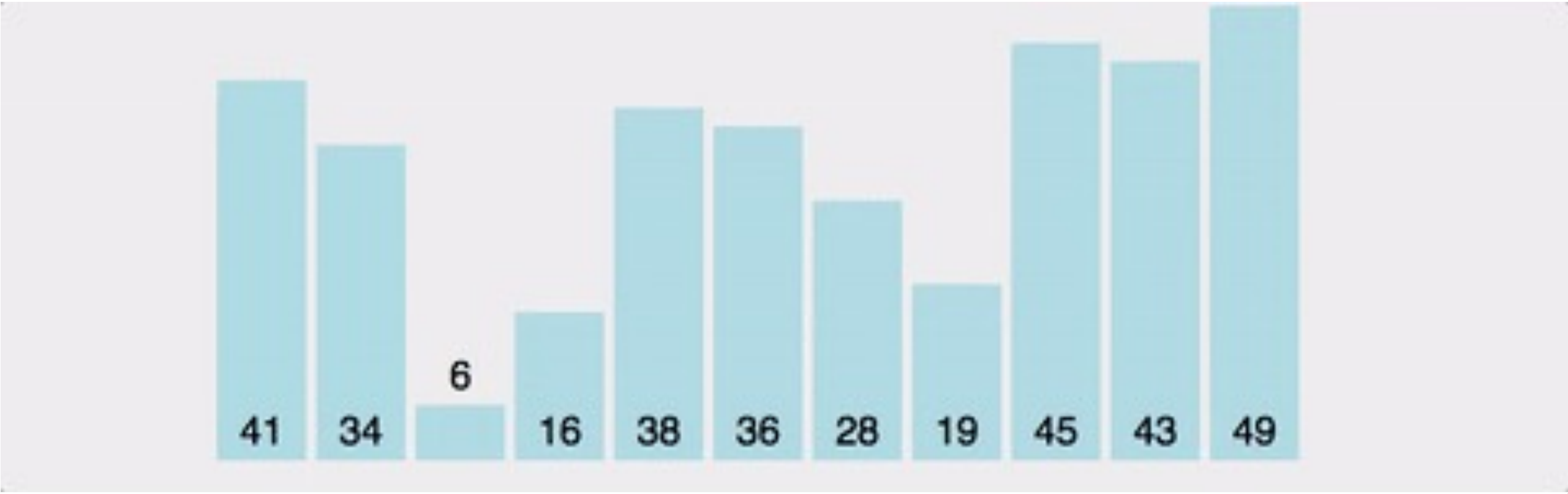
01. CONTENTS

버블정렬

```python
def bubbleSort(arr):
    N = len(arr)

    for i in range(N):
        for k in range(N-i-1):
            if arr[k] > arr[k+1]:
                arr[k],arr[k+1] = arr[k+1],arr[k]

    return arr


if __name__ == '__main__':
    arr = [43,32,24,60,15]
    print(bubbleSort(arr))
```

01. CONTENTS

버블정렬

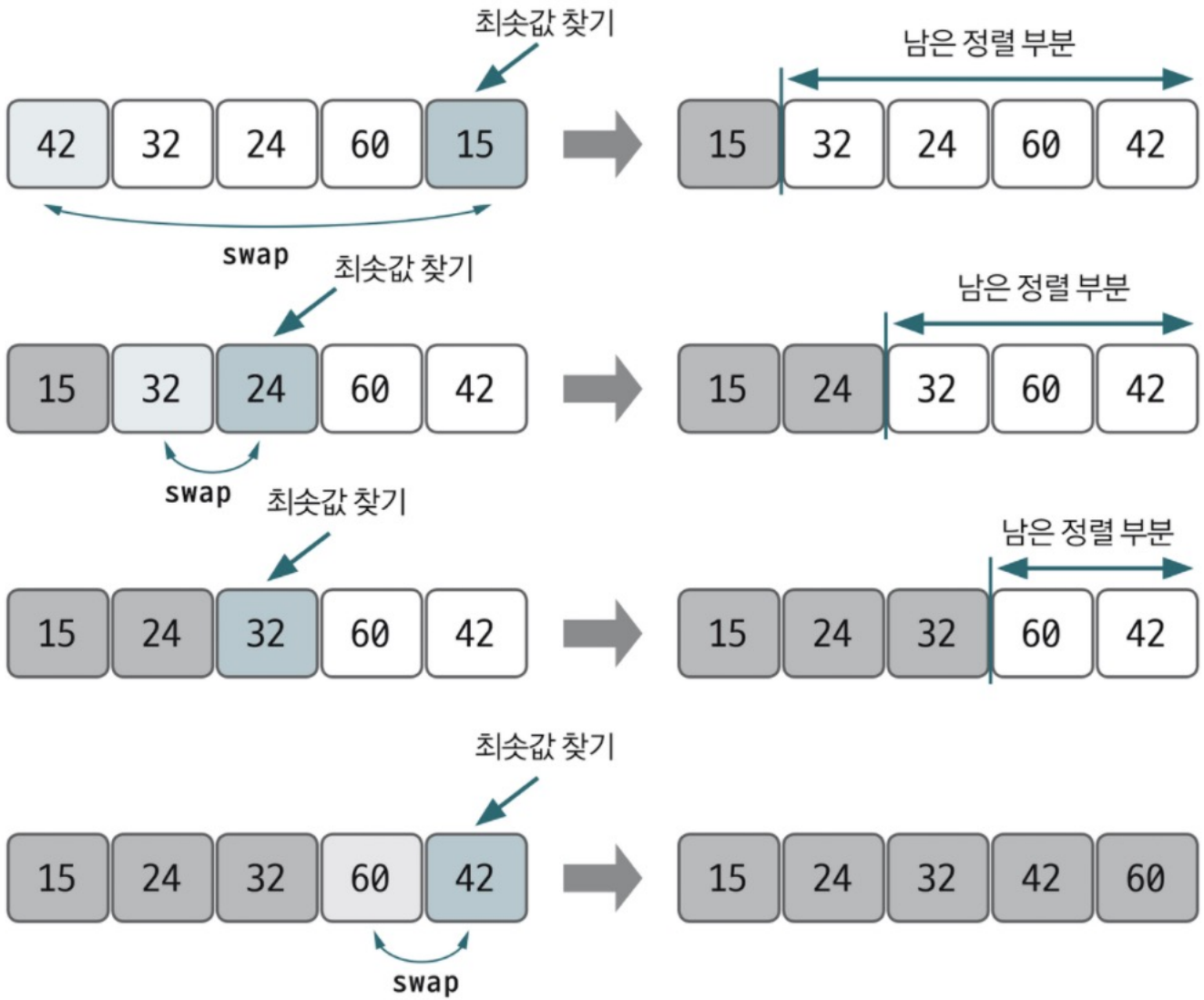


02

Selection Sort

02. CONTENTS

선택정렬



선택 정렬 수행 방식

시간복잡도 : O(n^2)

02. CONTENTS

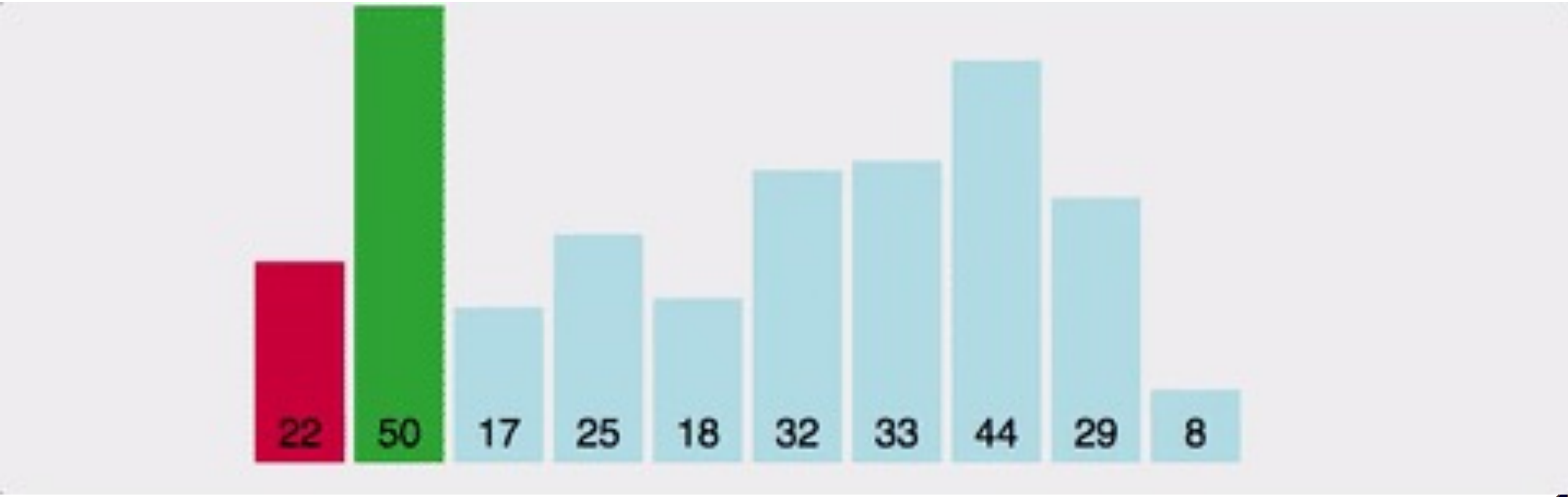
선택정렬

```
def selectedSort(arr):
    N = len(arr)
    for i in range(N-1):
        min_idx = i
        for j in range(i+1,N):
            if arr[i] > arr[j]:
                min_idx  = j
        if arr[min_idx] < arr[i]:
          arr[i],arr[min_idx] = arr[min_idx],arr[i]

    return arr

if __name__ == '__main__':
    arr = [42,32,24,60,15]
    print(selectedSort(arr))
```

시간복잡도 : O(n^2)

02. CONTENTS

선택정렬



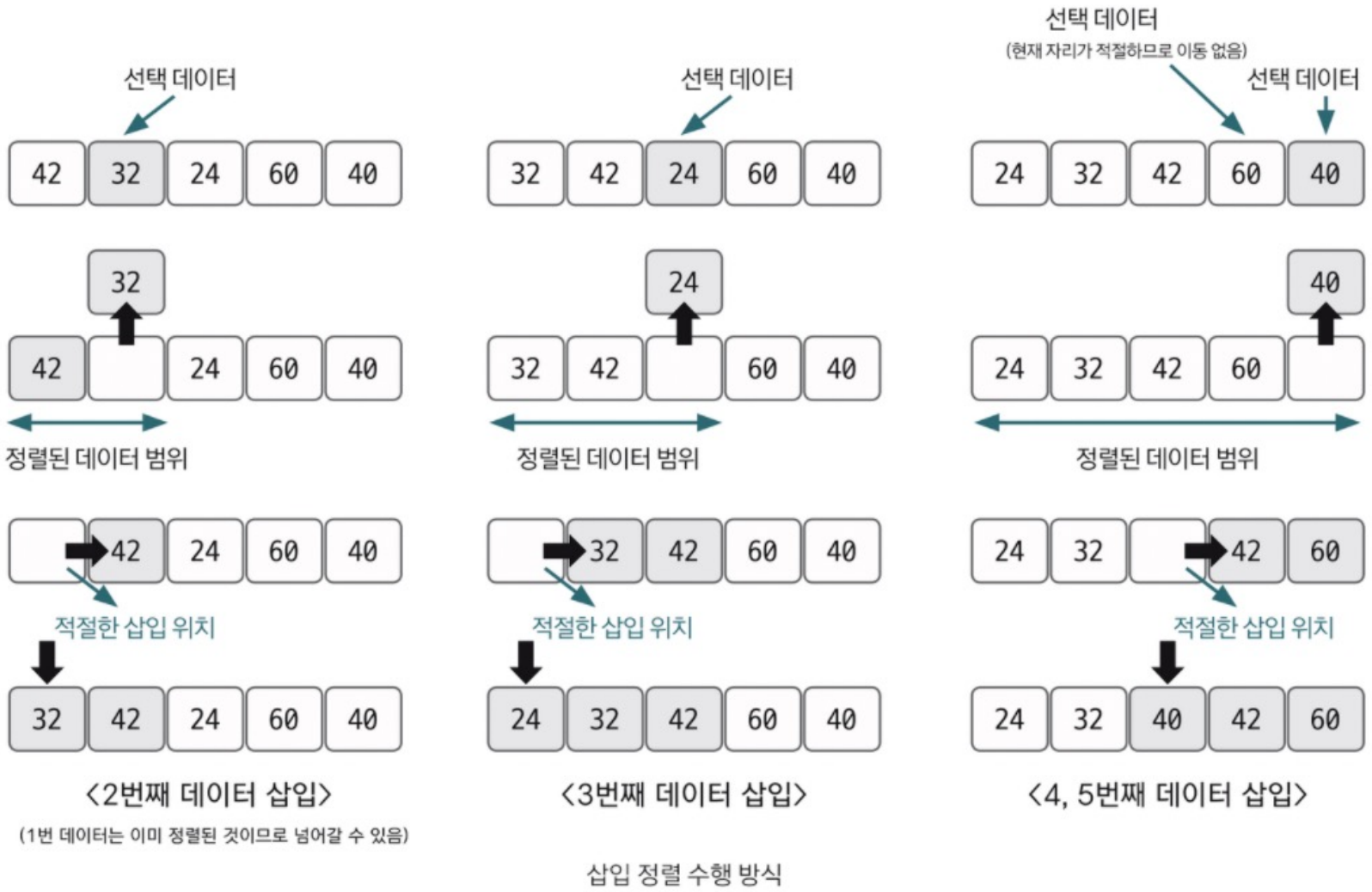
시간복잡도 : O(n^2)

03

Insertion Sort

03. CONTENTS

삽입정렬



삽입 정렬 수행 방식

시간복잡도 : O(n^2)

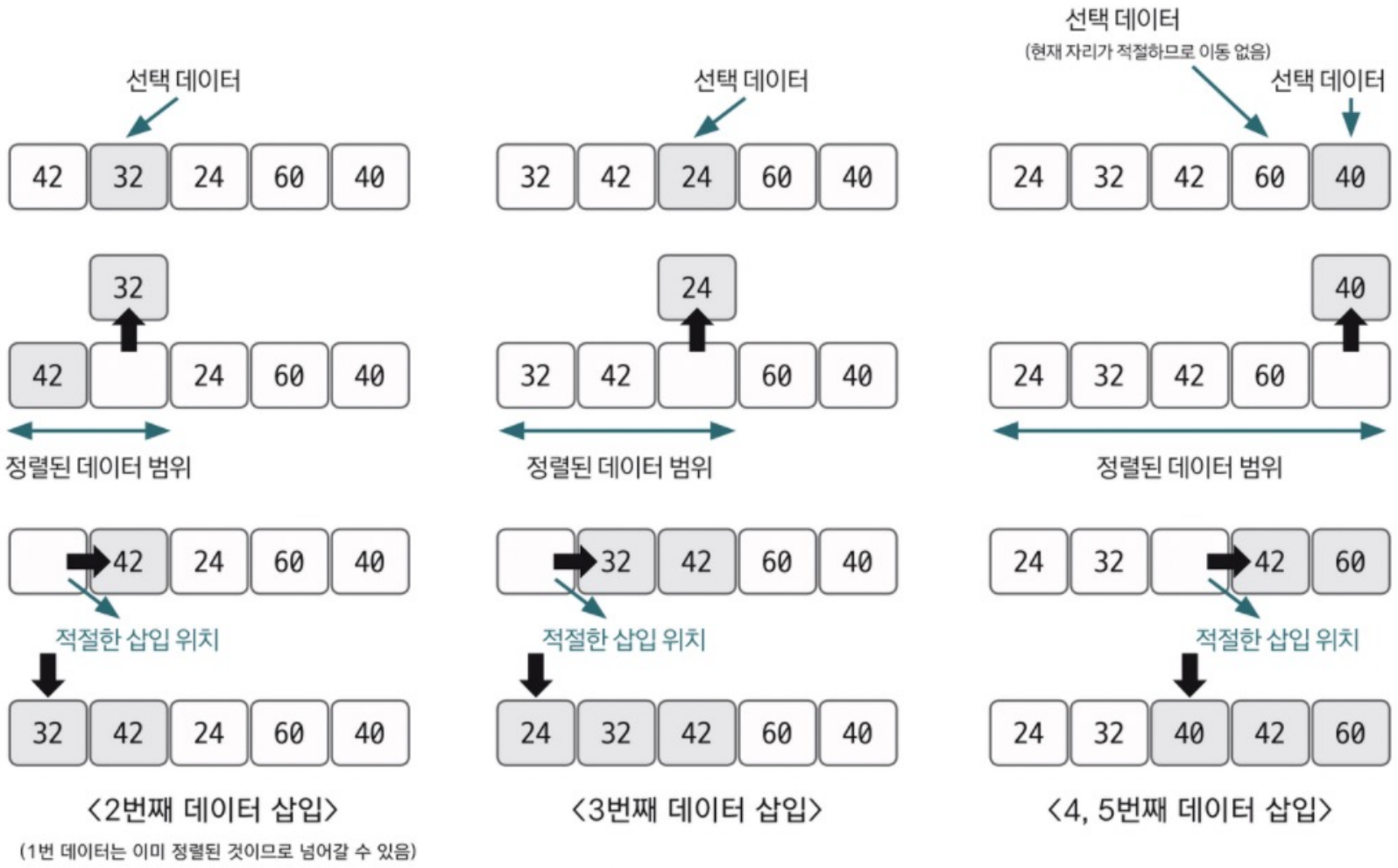
03. CONTENTS

삽입정렬

```
def insertSort(arr):
    N = len(arr)
    for i in range(1,N):
        for j in range(i,0,-1):
            if arr[j] < arr[j-1]:
                arr[j-1],arr[j] = arr[j],arr[j-1]

    return arr
```

03. CONTENTS

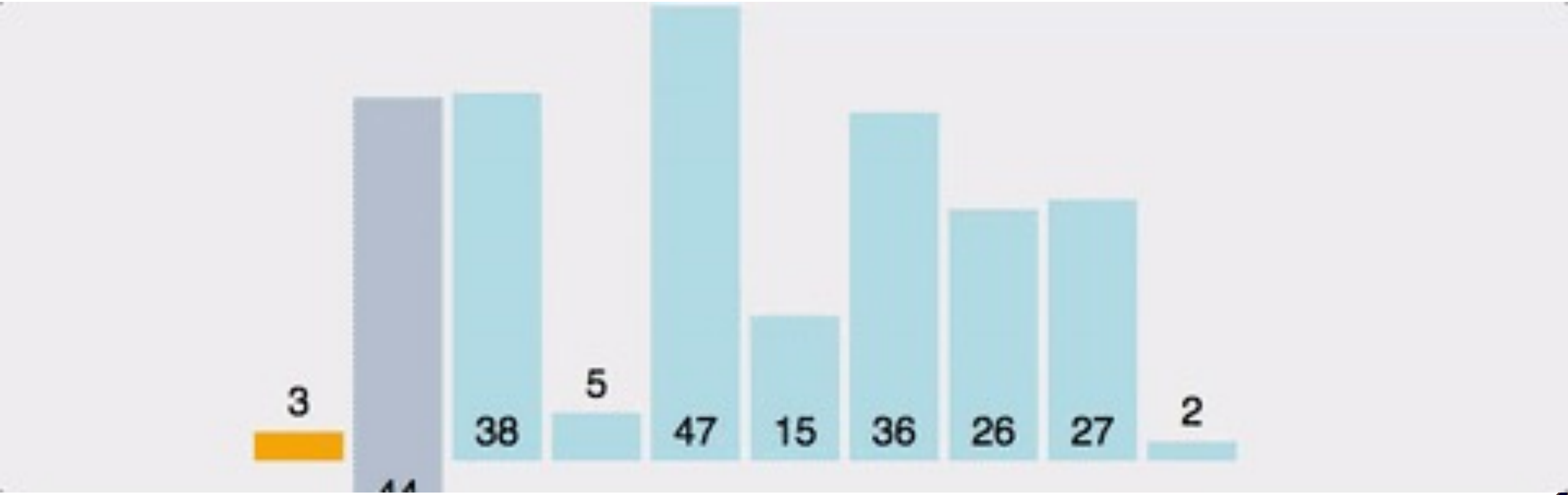
삽입정렬



삽입 정렬 수행 방식

03. CONTENTS

삽입정렬



03. CONTENTS

이진탐색

이분탐색이란 무엇인가 ?

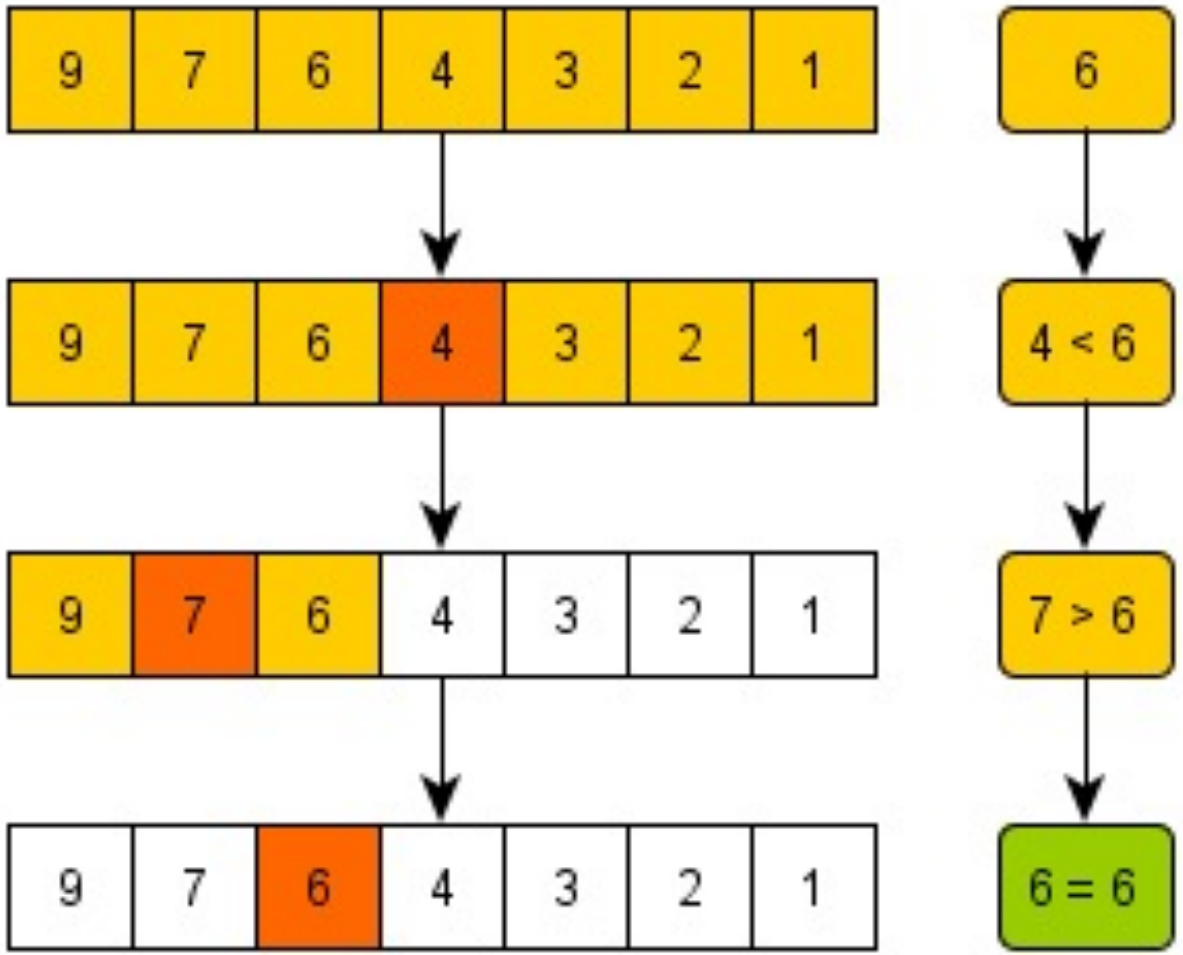
정렬 된 상태의 구간 내에서 중간에 위치한 원소와 비교하여 중간원소보다 작다면 왼쪽 구간으로, 크다면 오른쪽 구간으로 나누어 탐색하는 과정을 말한다.

어떻게 활용하는가 ?

현재의 수를 넣을 위치를 찾는 과정을 이진탐색으로 찾아내면 시간복잡도가 낮아진다.
현재의 수를 key 로 잡아서 이전까지 정렬해두었던 앞의 배열에서 key가 들어갈 자리를 찾는 것이다.

03. CONTENTS

이진탐색



Thank you

Alogorithm