Chapter. 02

알고리즘

다양한 정렬 응용
Sort Application - 응용편

FAST CAMPUS
ONLINE
알고리즘 공채 대비반I

강사. 류호석

Life Changing Education

Chapter. 02

알고리즘
Sort Application - 응용편

FAST CAMPUS
ONLINE
류호석 강사.

I Key Points

조건

- 정렬 조건이 필요하다.
- 시간 복잡도는 **약 O(N log N)** 이다.
  엄청 복잡한 얘기들이 있다...
- In-place / Stable 여부를 알아야 한다.

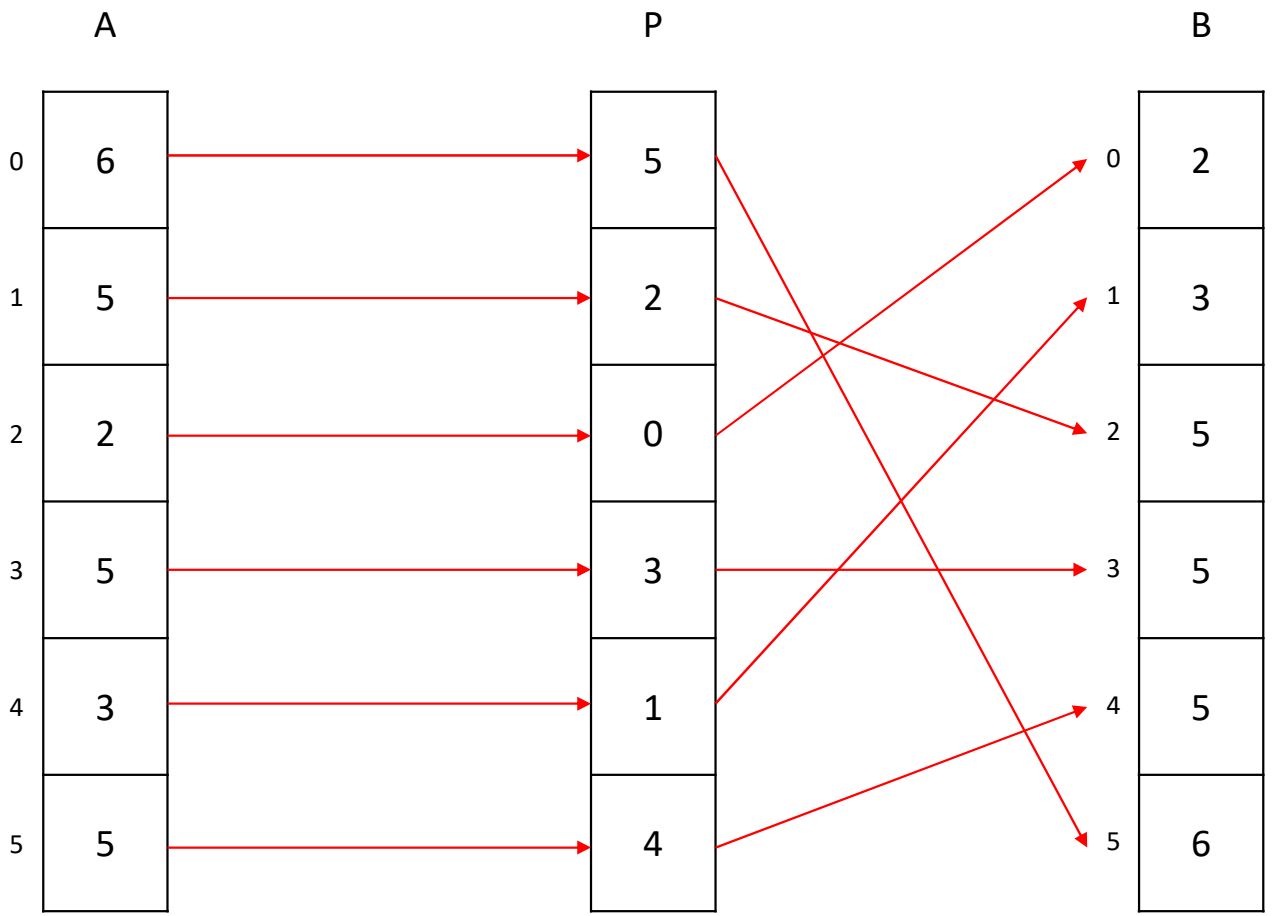
특성

- 같은 정보들은 인접해 있을 것이다.
- 각 원소마다, 가장 가까운 원소는 자신의 양 옆 중에 있다.
- 정렬만 해도 문제가 쉽게 풀리는 경우가 상당히 많다!

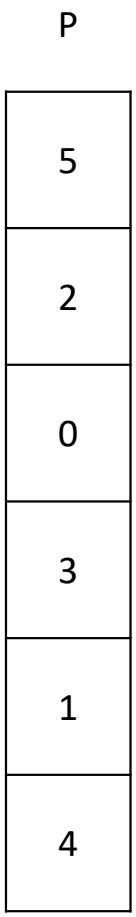
I BOJ 1015 – 수열 정렬

난이도: 2

$N \leq 50$



I 문제 파악하기 – 정답의 최대치

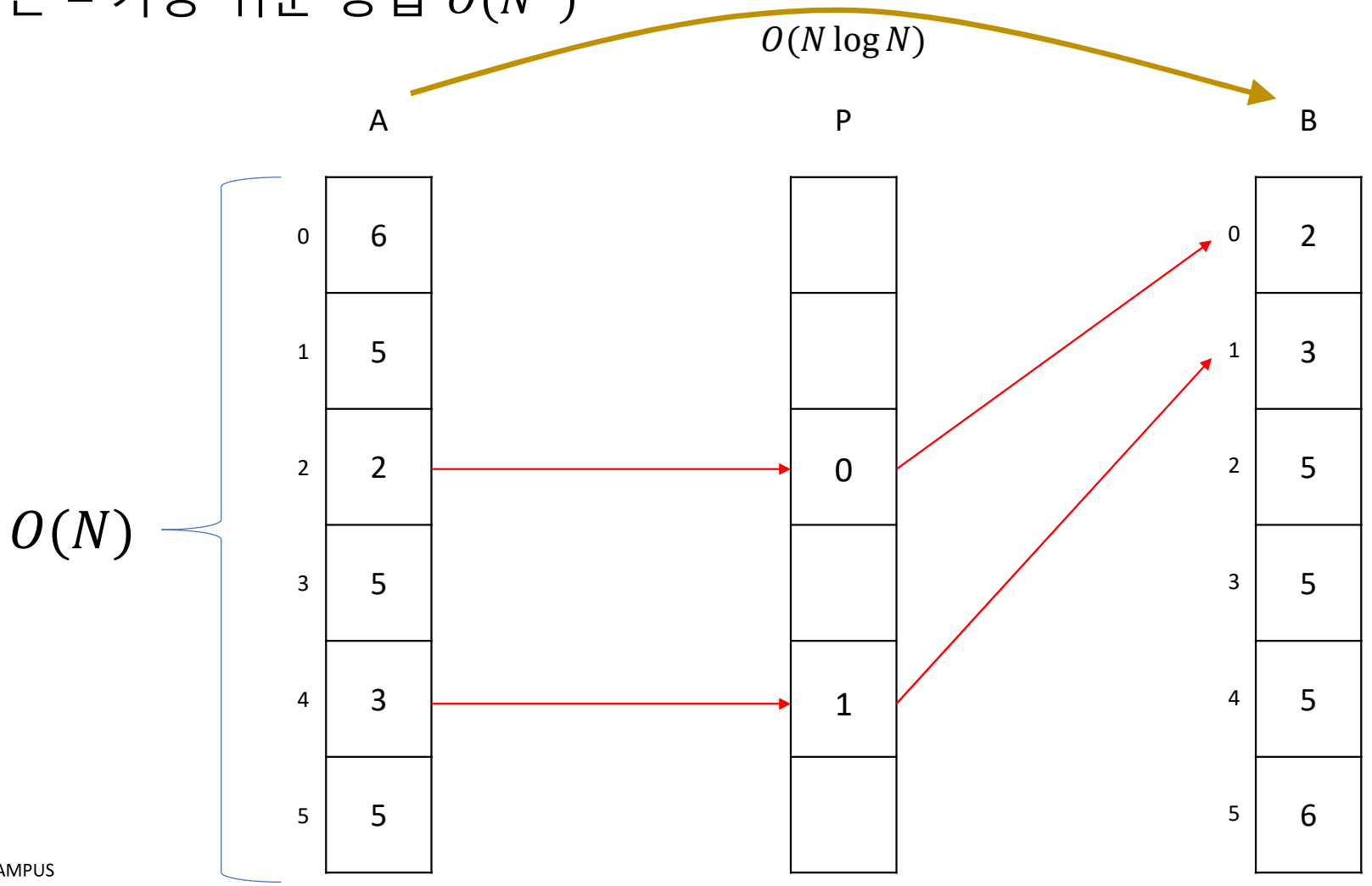


출력: $0 \sim N-1$

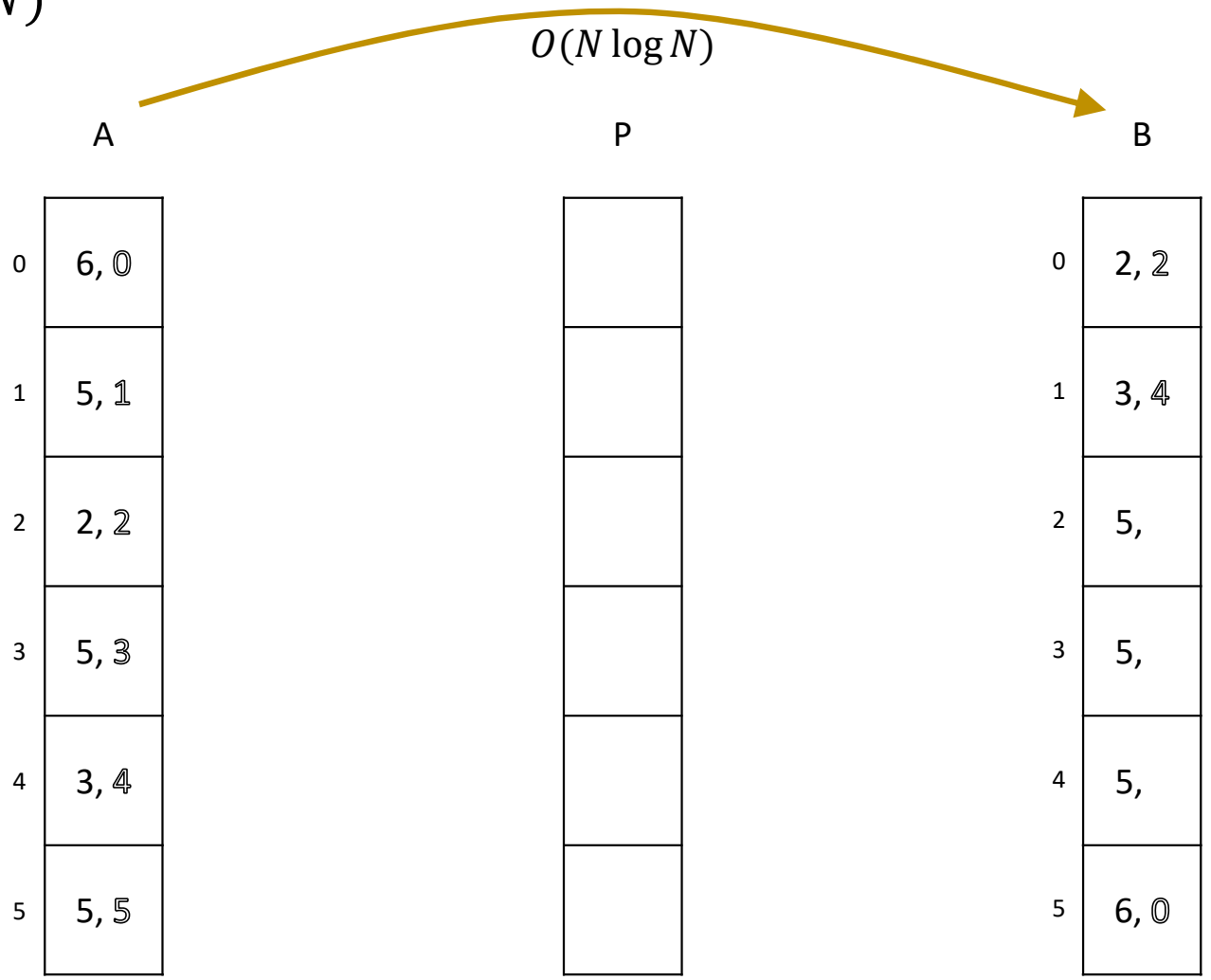
$N \leq 50$

Int 범위면 충분

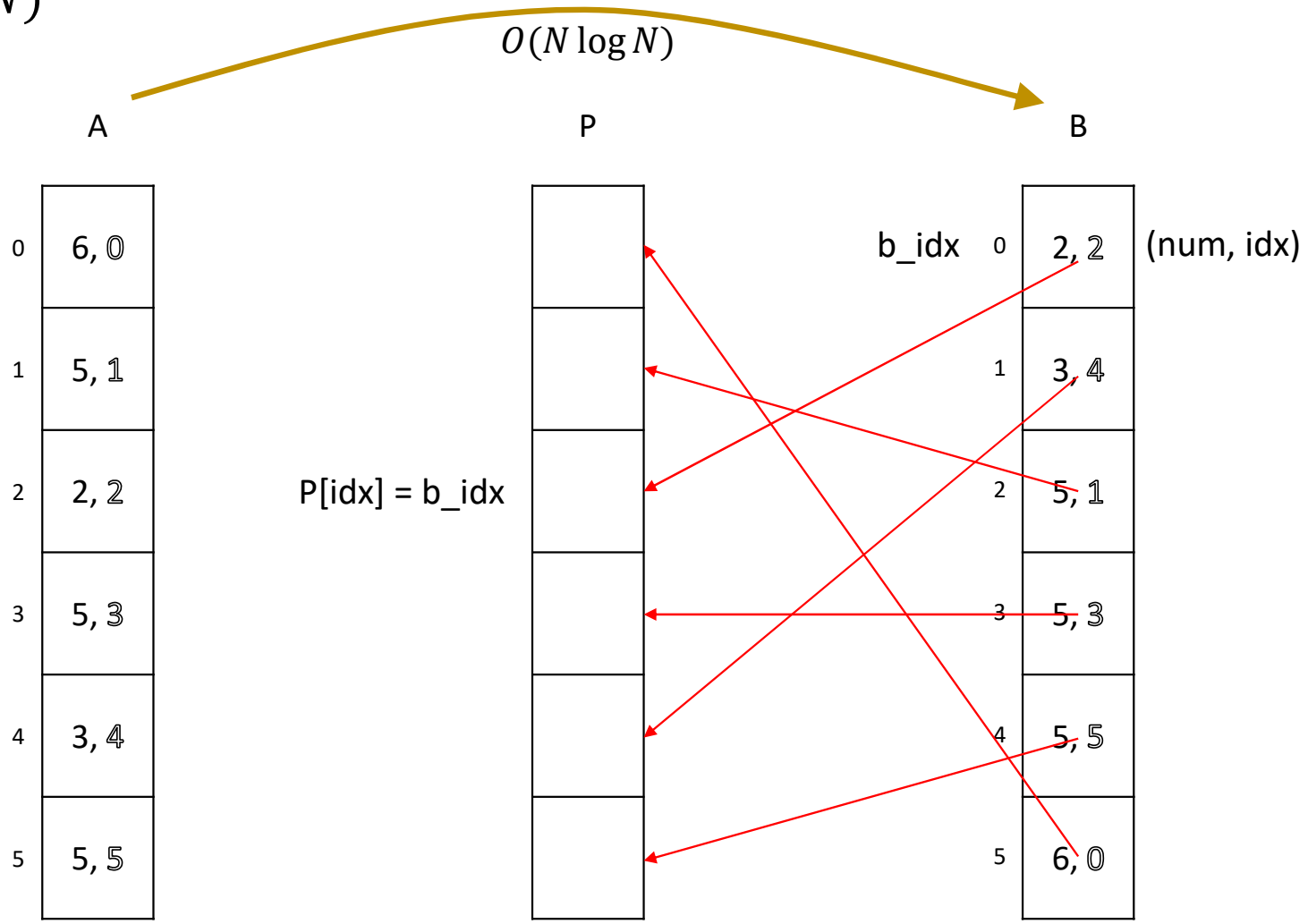
I 접근 – 가장 쉬운 방법 $O(N^2)$



I 접근 - $O(N \log N)$

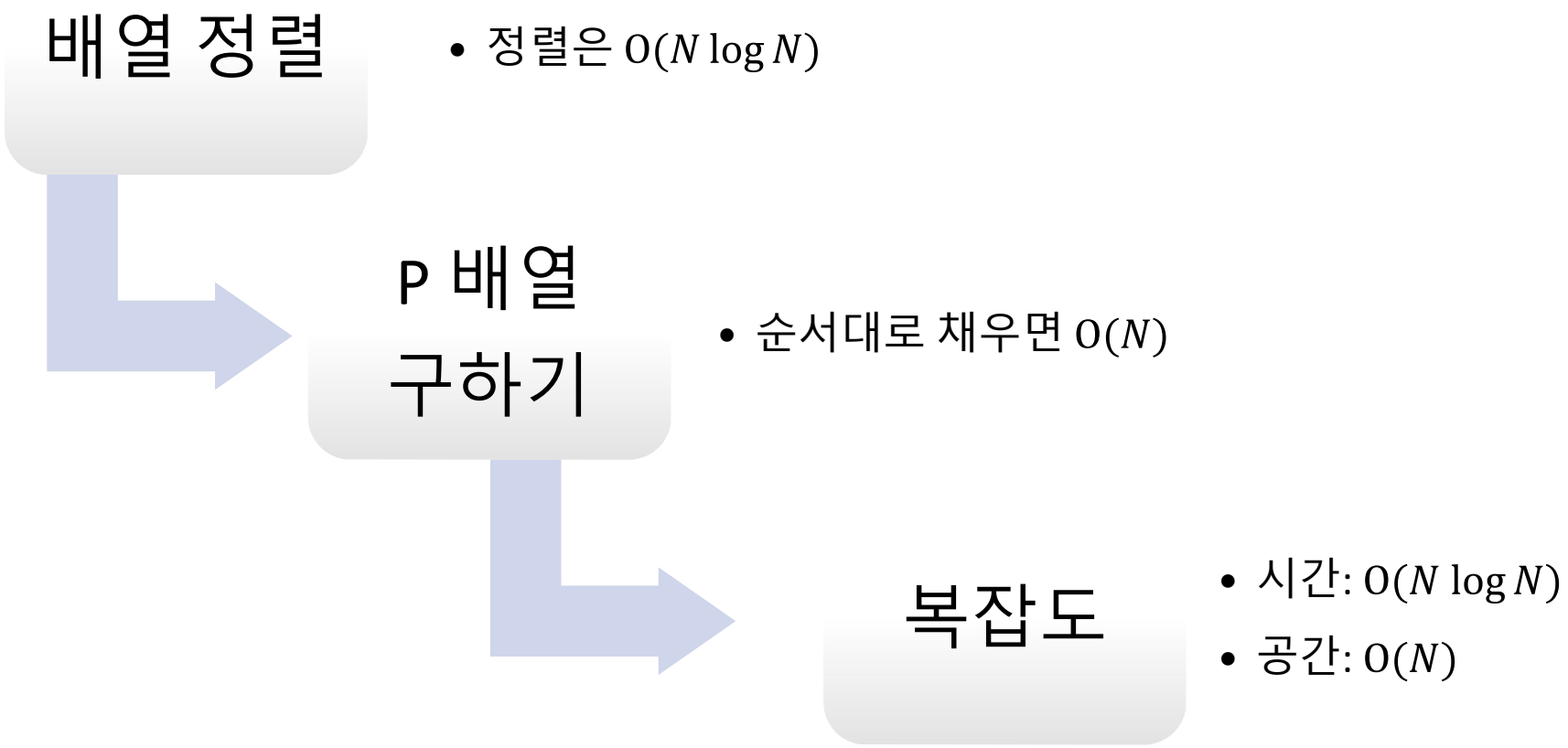


I 접근 - $O(N \log N)$



FAST CAMPUS
ONLINE
류호석 강사.

I 시간, 공간 복잡도 계산하기



I 구현

```
static class Elem implements Comparable<Elem> {

    /**
     * @param idx A 배열의 idx 위치를 기억하는 변수
     * @param num A[idx]의 원래 값
     */
    public int num, idx;

    @Override
    public int compareTo(Elem other) {
        // TODO
        // 정렬 조건에 맞게 정렬하기
        // 1. num 의 비내림차순
        // 2. num이 같으면 idx 오름차순
    }
}
static void pro() {
    // TODO: B 배열 정렬하기

    // TODO: B 배열의 값을 이용해서 P 배열 채우기

    // TODO: P 배열 출력하기
}
```

I 연습 문제

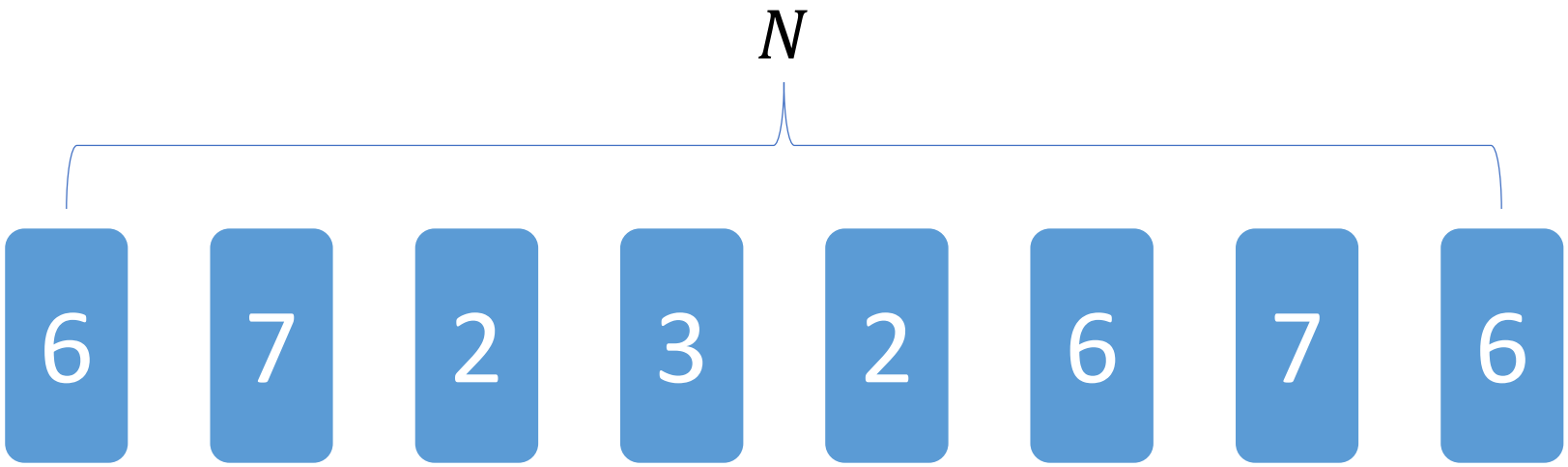
- BOJ 1181 – 단어 정렬

이외의 추천 문제가 추가되면 Github 자료에 코드 업로드

I BOJ 11652 – 카드

난이도: 1.5

- $N \leq 100{,}000$
- $-2^{62} \leq$ 카드 숫자 $\leq 2^{62}$



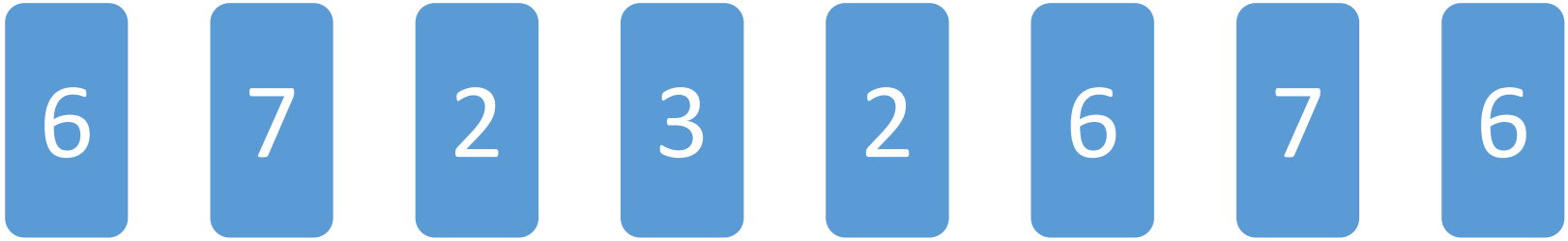
정답: 6

I 문제 파악하기 – 정답의 최대치

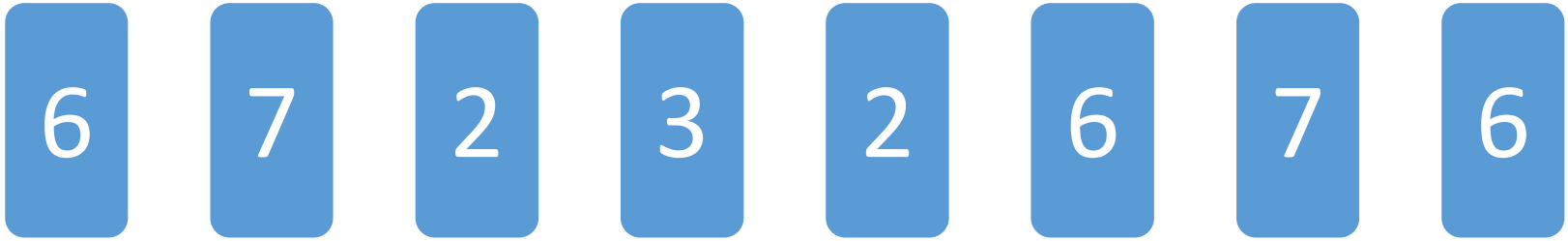
입출력: $-2^{62} \sim 2^{62}$

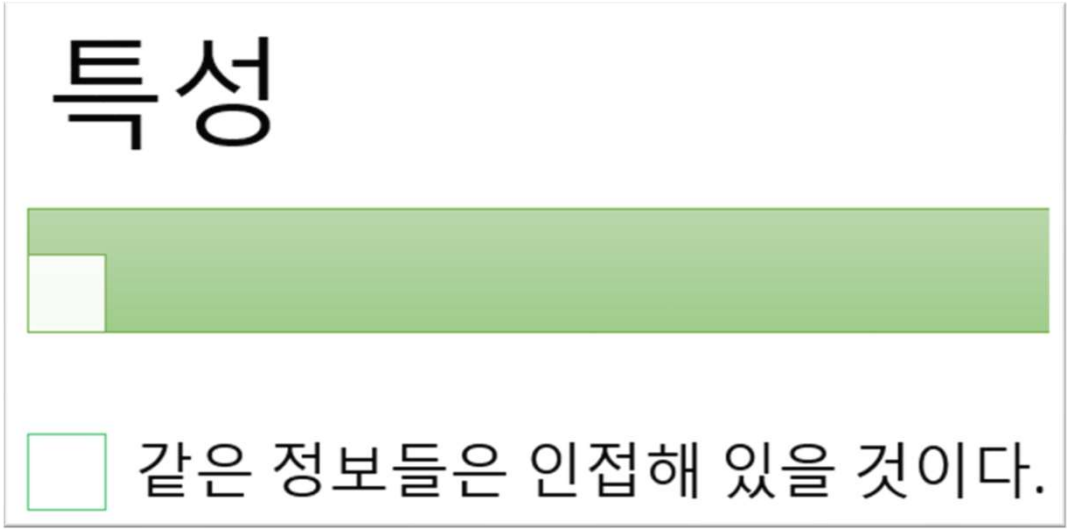
Int 범위로는 감당이 안되니까 long 을 쓰자!

I 접근 – 가장 쉬운 방법 $O(N^2)$

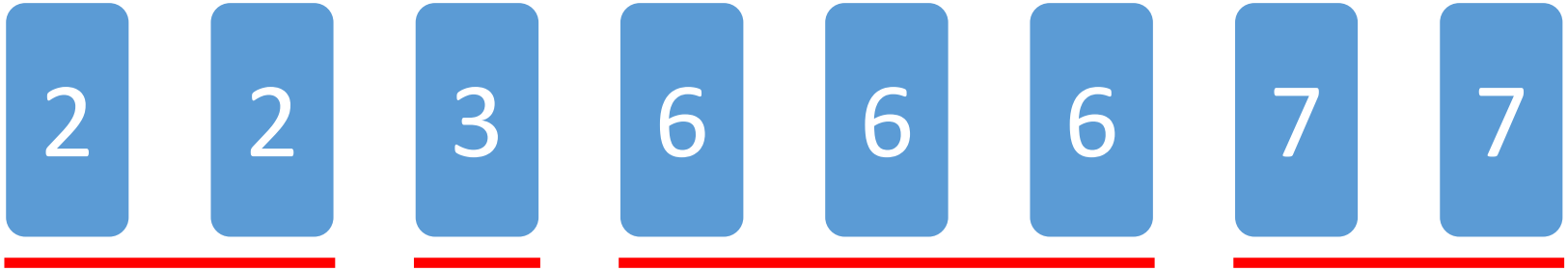


I 접근 – 같은 숫자를 빨리 보는 방법 $O(N \log N)$





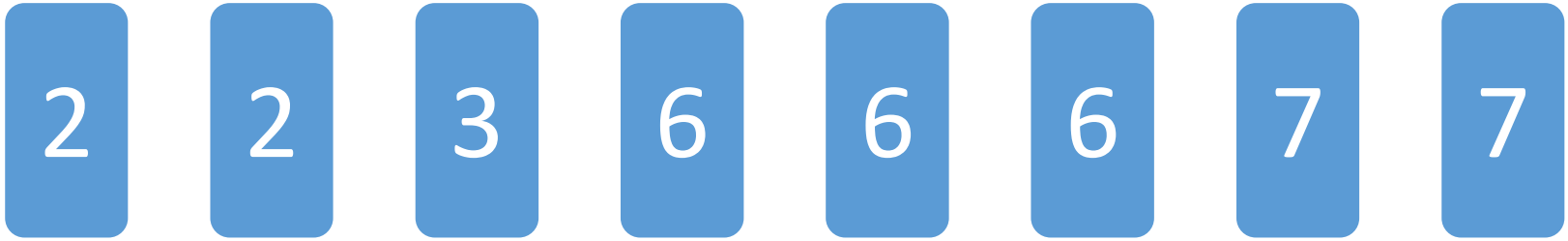
I 접근 – 같은 숫자를 빨리 보는 방법 $O(N \log N)$



Current Count								
Mode Count								
Mode								

Current Count: 지금 보고 있는 숫자가 등장한 횟수
Mode Count : 지금까지의 **최빈값**의 등장 횟수
Mode : 지금까지의 **최빈값**

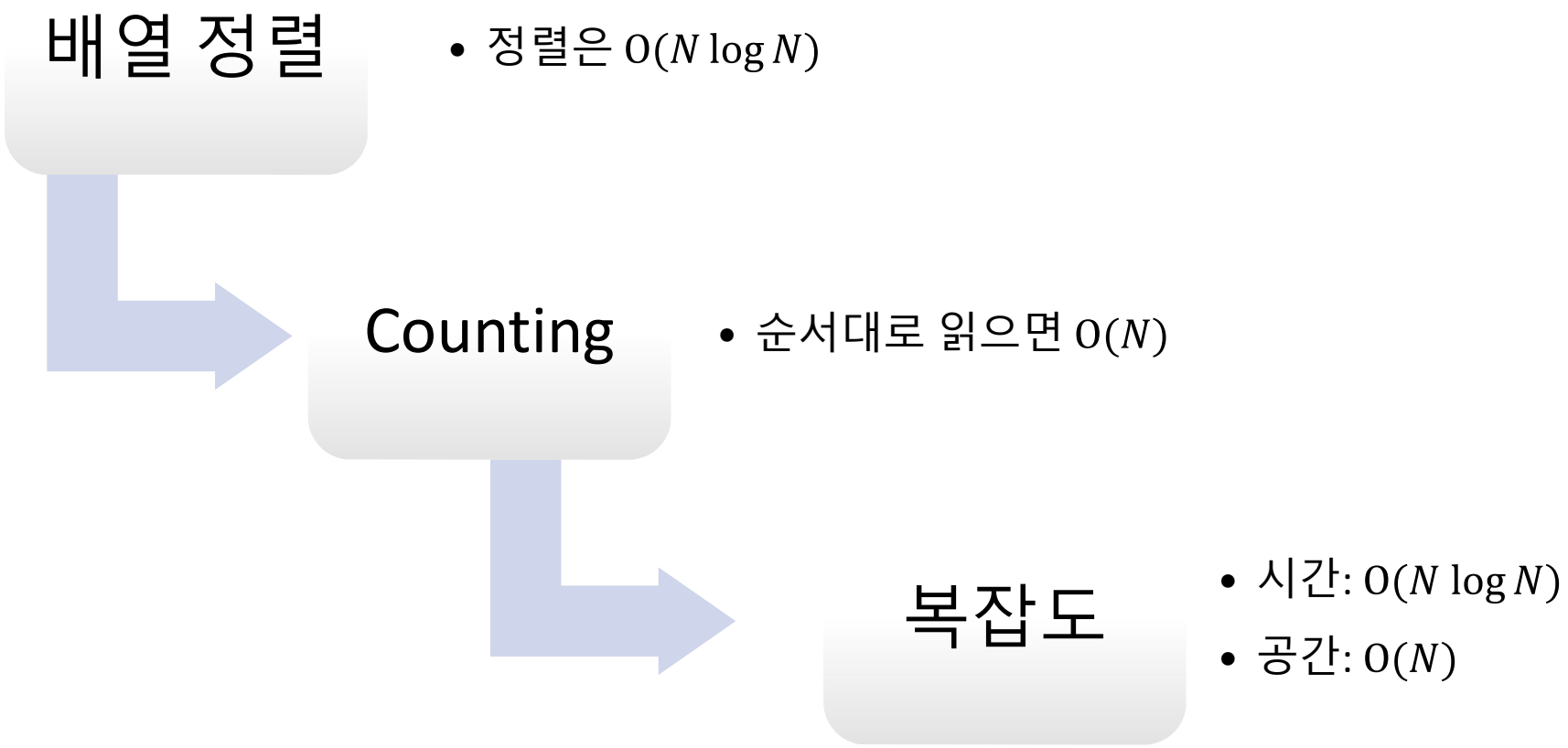
I 접근 – 같은 숫자를 빨리 보는 방법 $O(N \log N)$



Current Count	1	2	1	1	2	3	1	2
Mode Count	1	2	2	2	2	3	3	3
Mode	**2**	**2**	**2**	**2**	**6**	**6**	**6**	**6**

Current Count: 지금 보고 있는 숫자가 등장한 횟수
Mode Count : 지금까지의 **최빈값**의 등장 횟수
Mode : 지금까지의 **최빈값**

I 시간, 공간 복잡도 계산하기



I 구현

```
static void pro() {
    // Sort 정렬하기

    // mode: 최빈값, modeCnt: 최빈값의 등장 횟수, curCnt: 현재 값(a[1])의 등장 횟수
    long mode = a[1];
    int modeCnt = 1, curCnt = 1;



    // TODO
    // 2번 원소부터 차례대로 보면서, 같은 숫자가 이어서 나오고 있는 지, 새로운 숫자가 나왔는 지를 판단하여
    // curCnt를 갱신해주고, 최빈값을 갱신하는 작업.

    // 정답 출력하기
}
```

I 연습 문제

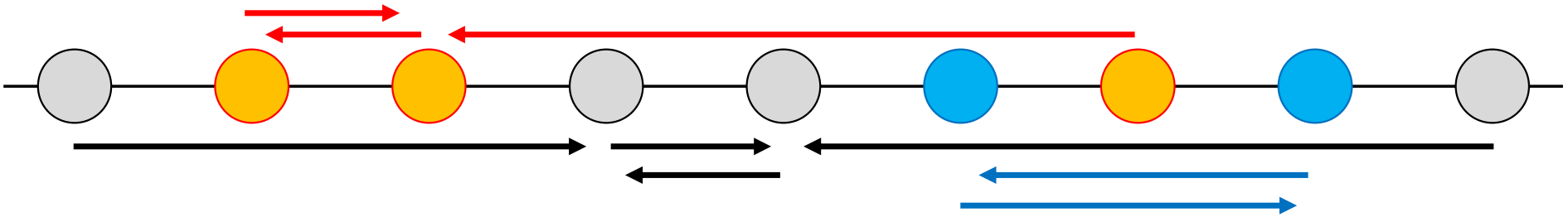
- BOJ 20291 – 파일 정리

이외의 추천 문제가 추가되면 Github 자료에 코드 업로드

BOJ 15970 – 화살표 그리기

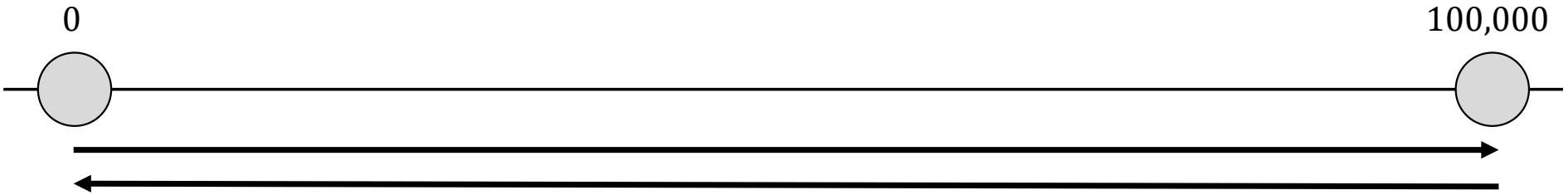
난이도: 2

- 점의 개수 $N \leq 5{,}000$
- $0 \leq$ 점의 위치 $\leq 10^5 = 100{,}000$
- $1 \leq$ 점의 색깔 $\leq N$



I 문제 파악하기 – 정답의 최대치

$N = 5{,}000$



점 두 개 ⇒ $2 * 10^5$ 만큼의 화살표 길이

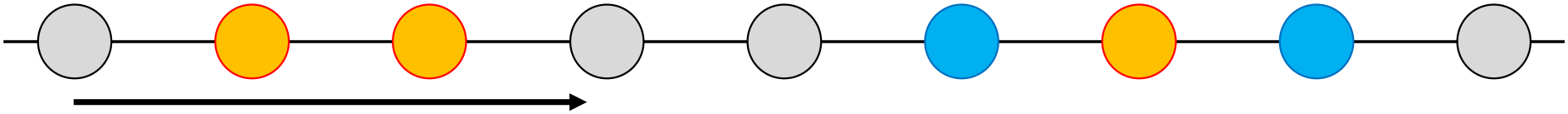
색깔마다 이런 점들이 있다면? 총 $5{,}000/2$ 쌍 만큼 만들 수 있다.

즉, 모든 점마다 10만 만큼의 길이를 갖는 화살표를 그리는 경우 이므로

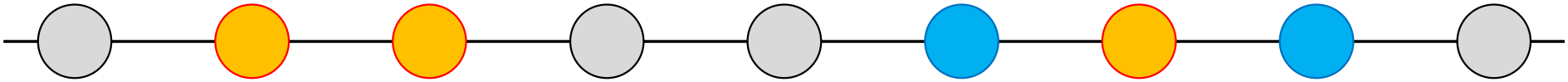
정답의 최대치: $5{,}000 * 100{,}000 = 5 * 10^8$

=> Integer 로 계산해도 충분하겠구나!

I 접근 – 가장 쉬운 방법 $O(N^2)$



I 접근 – 각 점마다, 자신과 가장 가까운 점을 빨리 찾기 $O(N \log N)$

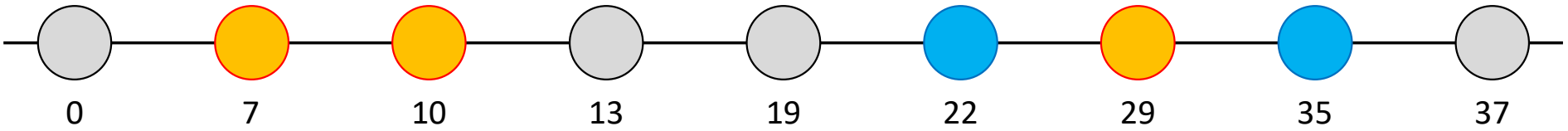


특성

- 각 원소마다, 가장 가까운 원소는 자신의 양 옆 중에 있다.

I 접근 – 각 점마다, 자신과 가장 가까운 점을 빨리 찾기 $O(N \log N)$

1. 같은 색깔의 점들만 모아서 보자.

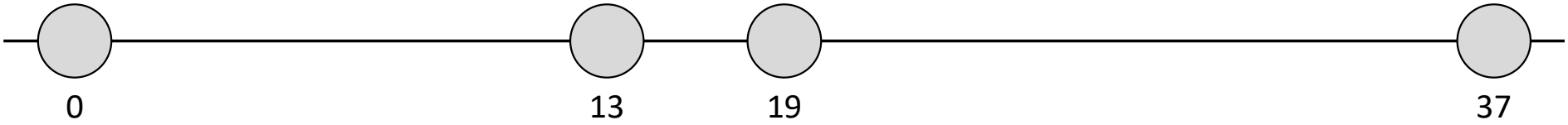


회색 점들	37	13	0	19
파란색 점들	35	22		
노란색 점들	10	29	7	

색깔마다 ArrayList를 만들어주면,
총 배열의 크기는 $O(N)$ 이다.

I 접근 – 각 점마다, 자신과 가장 가까운 점을 빨리 찾기 $O(N \log N)$

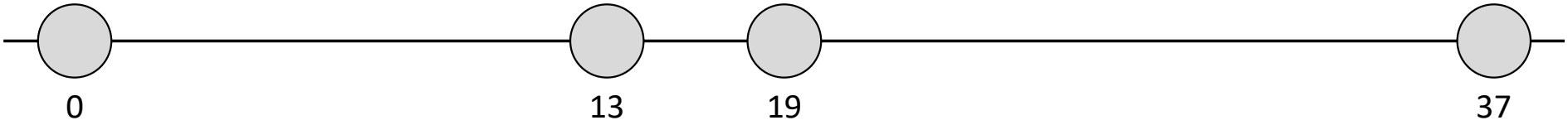
2. 모은 뒤에, 각 점마다 자신과 가장 가까운 것을 찾아야 한다.



회색 점들	33	37	0	19

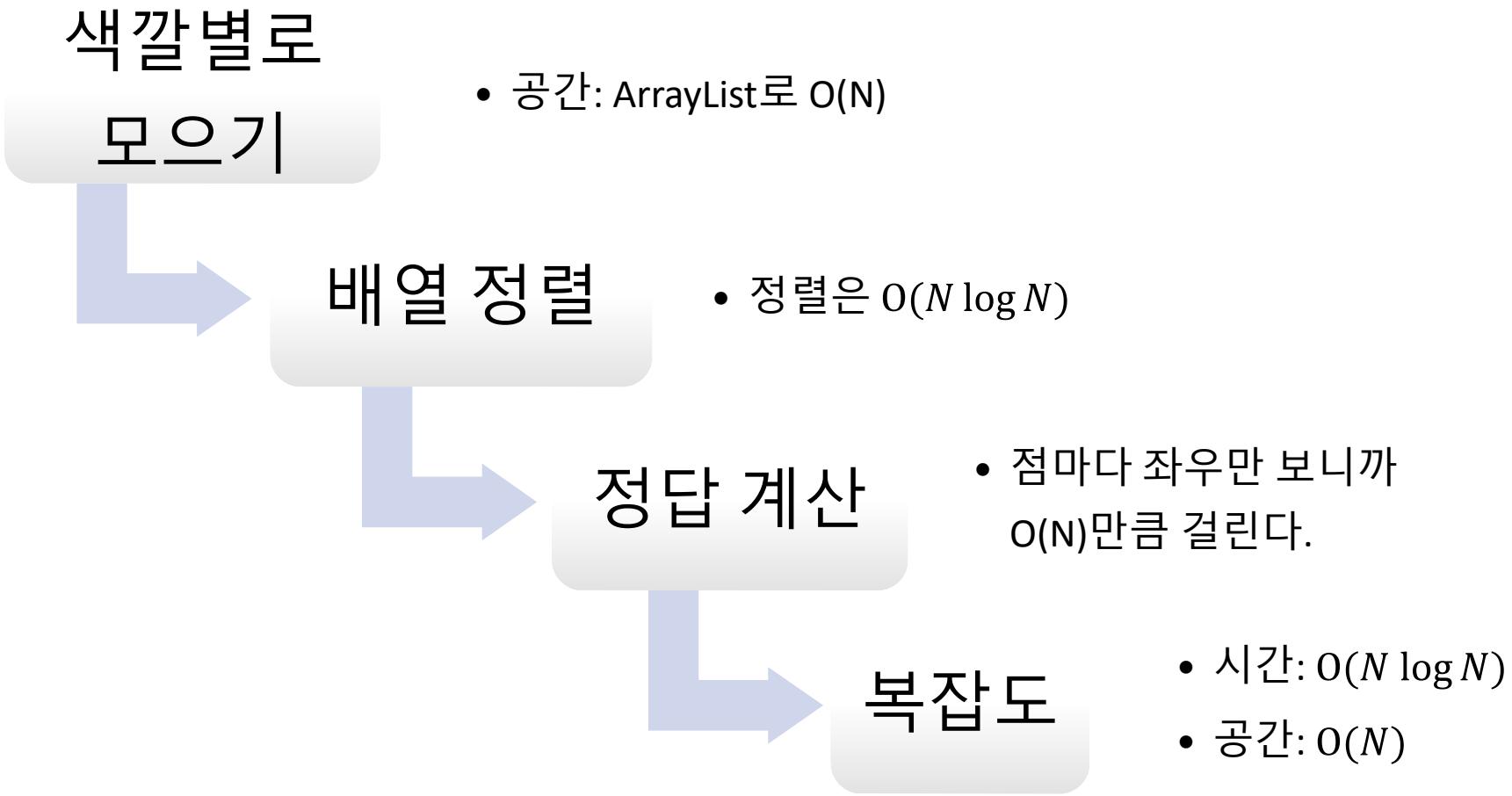
I 접근 – 각 점마다, 자신과 가장 가까운 점을 빨리 찾기 $O(N \log N)$

3. 정렬의 특성을 이용하기 위해 점들의 위치를 오름차순 정렬한다. $O(N \log N)$



회색 점들	0	13	19	37

I 시간, 공간 복잡도 계산하기



I 구현

```java
static int N;
static ArrayList<Integer>[] a;

static void input() {
    N = scan.nextInt();
    a = new ArrayList[N + 1];
    for (int color = 1; color <= N; color++) {
        a[color] = new ArrayList<Integer>();
    }
    for (int i = 1; i <= N; i++) {
        int coord, color;
        coord = scan.nextInt();
        color = scan.nextInt();
        // TODO: color 인 색깔의 점이 coord 에 놓여 있음
    }
}
```

```java
static void pro() {
    // TODO: 색깔별로 정렬하기

    int ans = 0;
    for (int color = 1; color <= N; color++) {
        // TODO: 색깔 별로, 각 점마다 가장 가까운 점 찾아주기
    }

    // 정답 출력하기
}
```