

# 정렬

최백준 [choi@startlink.io](mailto:choi@startlink.io)

---

# 정렬

---

# 정렬

sort

3

- 많은 정렬 알고리즘이 있다.

- 선택 정렬, 버블 정렬, 삽입 정렬, 퀵 정렬, 힙 정렬, 병합 정렬, ...

- 정렬은  $O(N \lg N)$ 이 걸리는 정렬을 사용하는 것으로 한다.

- 정렬을 직접 구현하는 것 보다는 STL에 있는 sort를 사용하는 것이 좋다.

- sort(begin, end)

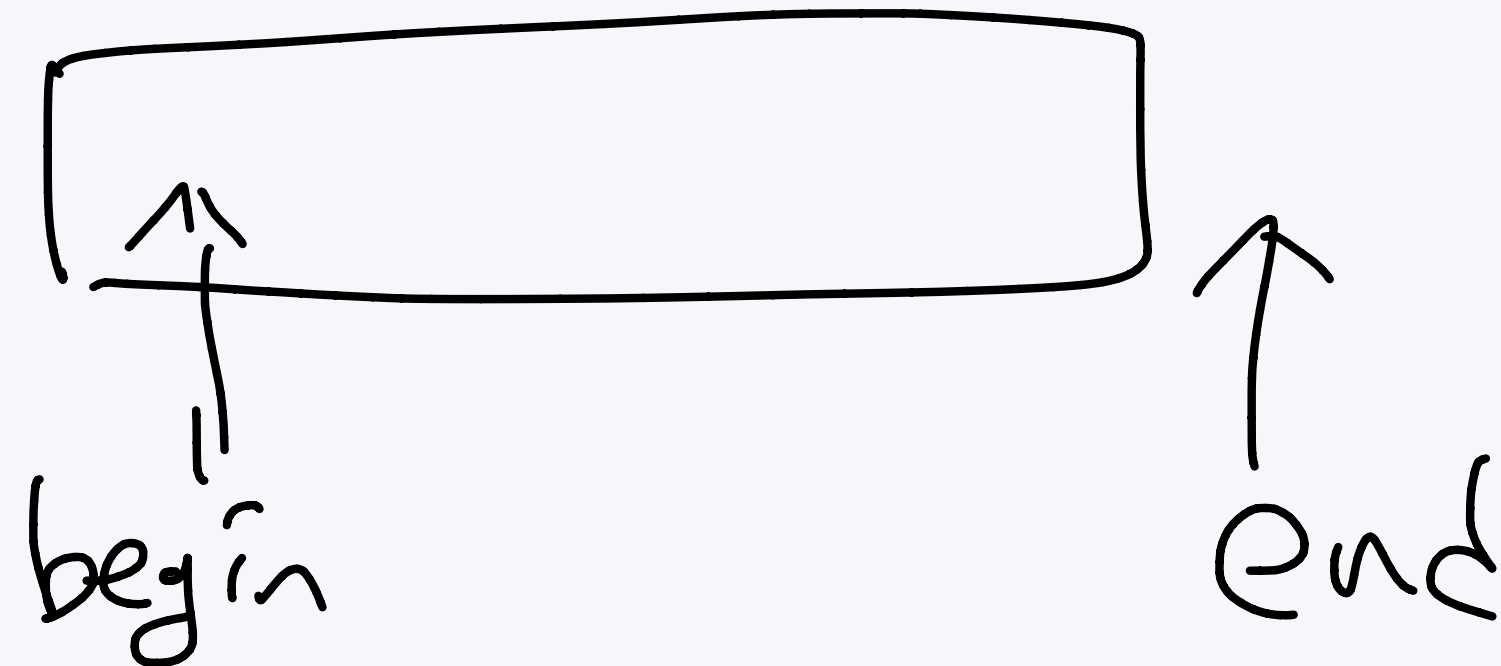
- begin부터 end 바로 전까지를 정렬하는 함수이다.

- [begin, end)를 정렬하는 함수

$O(N^2)$   
→  $O(N \lg N)$

C++

`#include <algorithm>`



# 정렬

sort

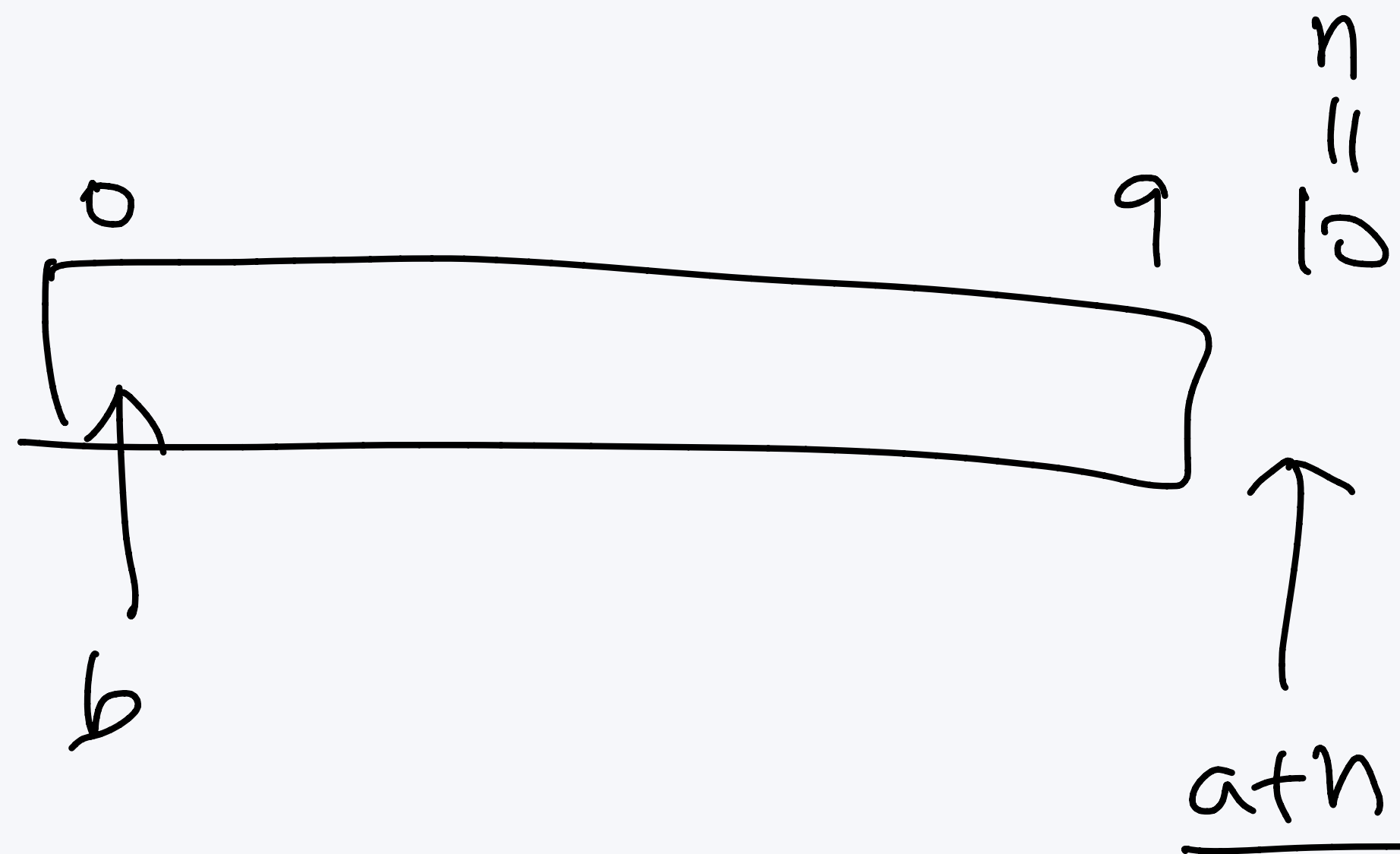
```
int n = 10;  
int a[10] = {};  
(sort(a, a+n));
```

- a[0]부터 a[n-1]까지를 정렬하는 소스

```
vector<int> a;  
sort(a.begin(), a.end());
```

- vector를 정렬하는 소스

10



# 수 정렬하기 2

<https://www.acmicpc.net/problem/2751>

N

5

- N개의 수를 정렬하는 문제
- C/C++ (Quick Sort 구현): <https://gist.github.com/Baekjoon/e9664b3b9da3d3e1a556>
- C/C++ (Merge Sort 구현): <https://gist.github.com/Baekjoon/797adaa952295573398c>
- C++ (STL Sort): <https://gist.github.com/Baekjoon/13d4a237d9e18d2fb1cf>

# 좌표 정렬하기

<https://www.acmicpc.net/problem/11650>



- $(x, y)$ 가 여러 개 있을 때,  $x$ 가 증가하는 순으로, 같으면  $y$ 가 증가하는 순서로 정렬하는 문제
- `pair`를 사용하면 편한다.

```
int n;
scanf("%d", &n);
vector<pair<int, int>> a(n);
for (int i=0; i<n; i++) {
    scanf("%d %d", &a[i].first, &a[i].second);
}
sort(a.begin(), a.end());
for (int i=0; i<a.size(); i++) {
    printf("%d %d\n", a[i].first, a[i].second);
}
```

# 좌표 정렬하기

<https://www.acmicpc.net/problem/11650>

- 직접 struct를 구현하는 경우에는 비교 함수를 만들어 줘야 한다.

```
struct Point {  
    int x, y;  
};  
bool cmp(const Point &u, const Point &v) {  
    if (u.x < v.x) {  
        return true;  
    } else if (u.x == v.x) {  
        return u.y < v.y;  
    } else {  
        return false;  
    }  
}
```

# 좌표 정렬하기

u

v

8

<https://www.acmicpc.net/problem/11650>

- cmp 함수는 u가 v의 앞에 오는 것이면 true, 아니면 false 이다.
- const와 &는 붙여야 한다.

true  
false;

```
bool cmp(const Point &u, const Point &v) {  
    if (u.x < v.x) {  
        return true;  
    } else if (u.x == v.x) {  
        return u.y < v.y;  
    } else {  
        return false;  
    }  
}
```

$u.x > v.x$

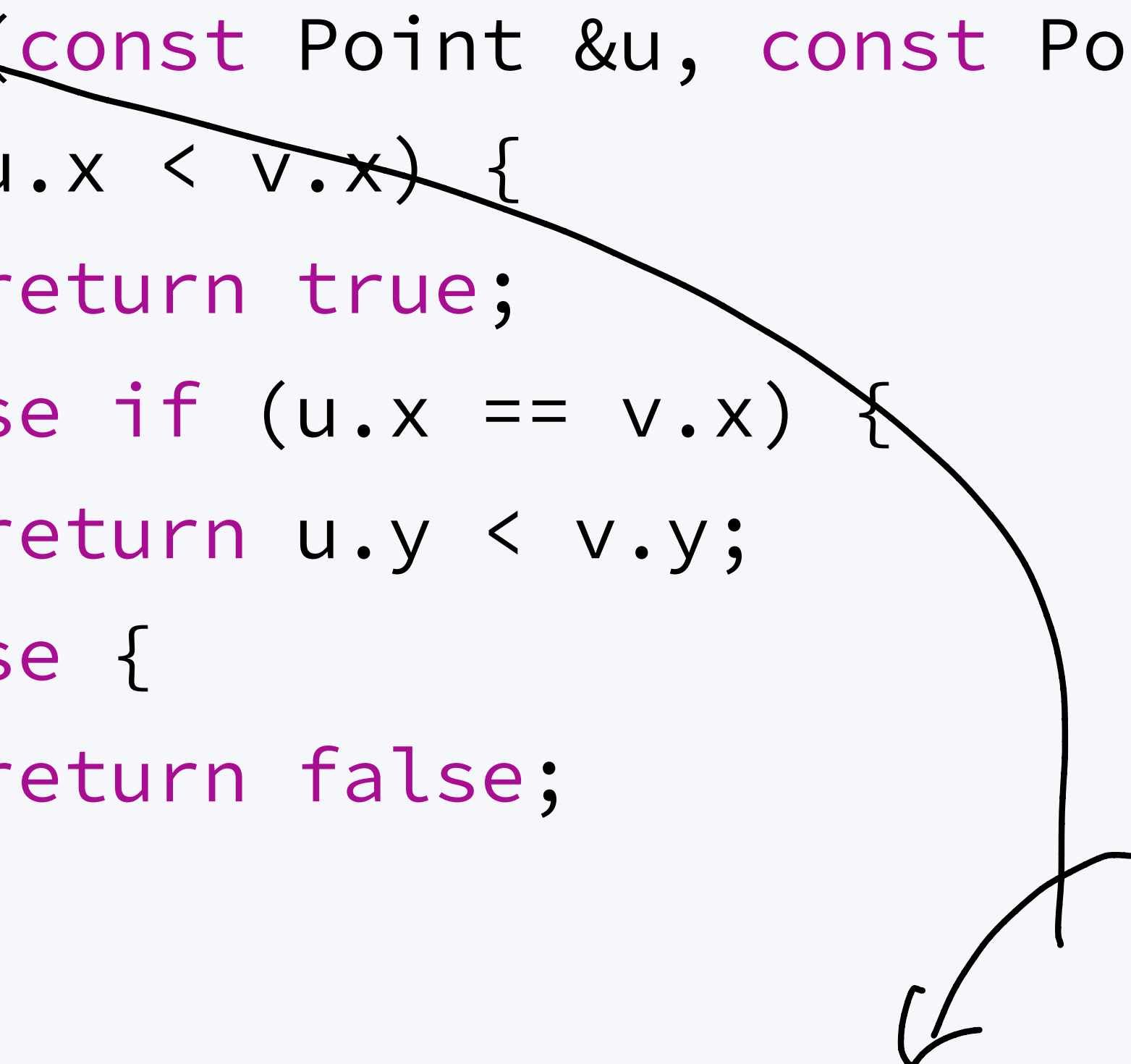


# 좌표 정렬하기

<https://www.acmicpc.net/problem/11650>

- 비교 함수를 만드는 경우에는 3번째 인자로 함수 이름을 넘겨줘야 한다.

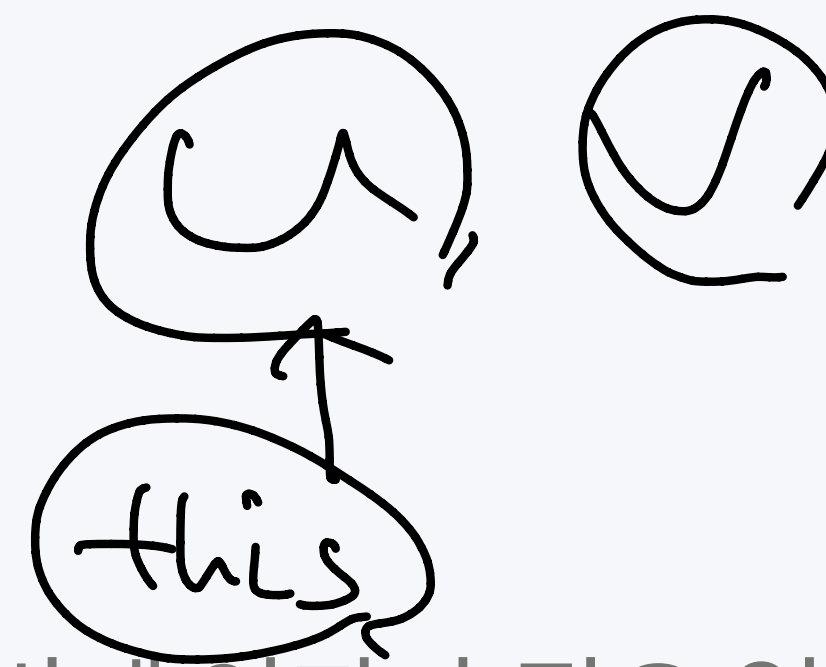
```
bool cmp(const Point &u, const Point &v) {  
    if (u.x < v.x) {  
        return true;  
    } else if (u.x == v.x) {  
        return u.y < v.y;  
    } else {  
        return false;  
    }  
}  
  
sort(a.begin(), a.end(), cmp);
```



# 좌표 정렬하기

<https://www.acmicpc.net/problem/11650>

10



- < 연산자를 over loading 할 수도 있다. 이 경우에는 3번째 인자가 필요 없다.

```
struct Point {  
    int x, y;  
    bool operator < (const Point (&v) const {  
        if (x < v.x) {  
            return true;  
        } else if (x == v.x) {  
            return y < v.y;  
        } else {  
            return false;  
        }  
    }  
}
```

# 좌표 정렬하기

<https://www.acmicpc.net/problem/11650>

- C++ (pair): <https://gist.github.com/Baekjoon/c6c920ca4fd31eb9960e>
- C++ (struct, cmp): <https://gist.github.com/Baekjoon/f3d180e438e2087a5a97>
- C++ (struct, 연산자 오버로딩): <https://gist.github.com/Baekjoon/180de19581f0b55c433c>

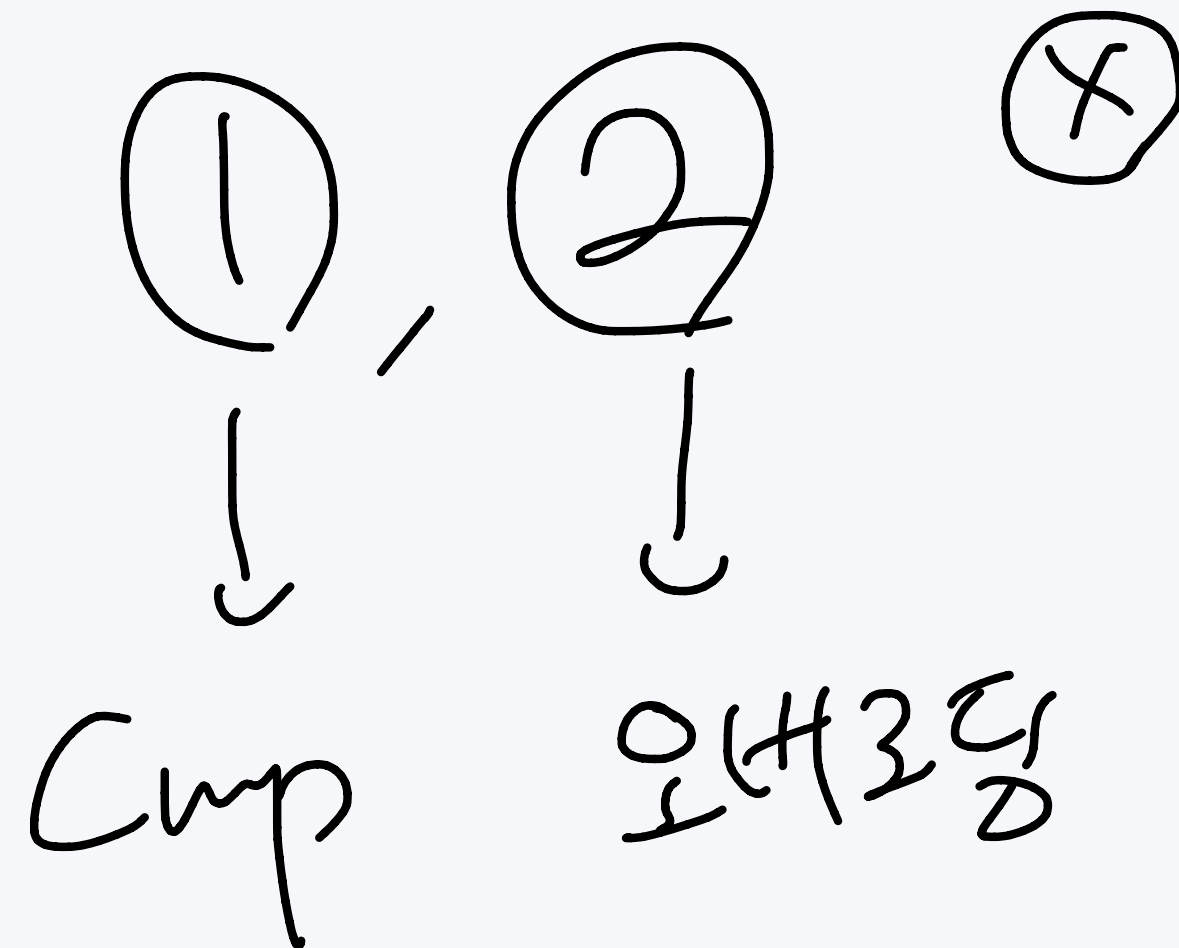
# 좌표 정렬하기 2

12

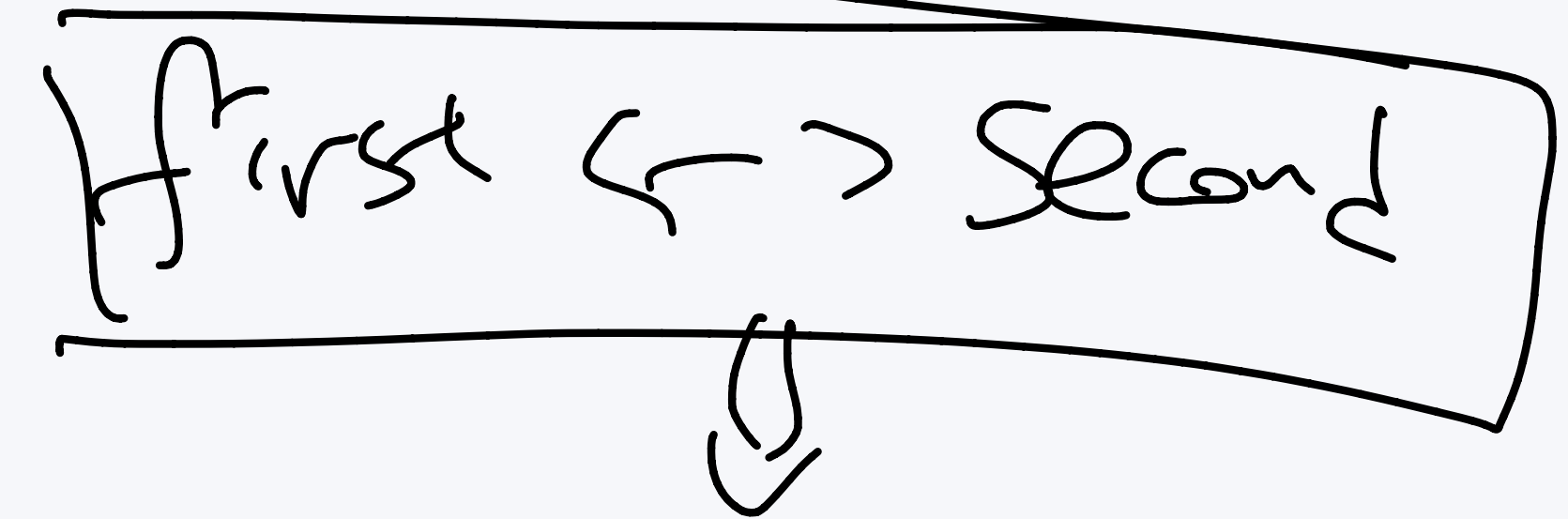
<https://www.acmicpc.net/problem/11651>

y

- $(x, y)$ 가 여러 개 있을 때,  $y$ 가 증가하는 순으로, 같으면  $x$ 가 증가하는 순서로 정렬하는 문제



비교함 순



# 좌표 정렬하기 2

<https://www.acmicpc.net/problem/11651>

- C++ (pair, cmp): <https://gist.github.com/Baekjoon/df02385467e2c6e36d59>
- C++ (pair, 뒤집어서 저장): <https://gist.github.com/Baekjoon/7cee7721722d6789769c>

# Stable Sorting

## Stable Sorting

오름차순 정렬

14

- 예를 들어 다음과 같이 카드가 있는 경우를 생각해보자.

• 7♠, 5♥, 2♥, 5♠

중간상 위치

- 위의 카드를 번호가 증가하는 순서로 정렬했을 때, 5♥와 5♠의 순서에 대해 생각해본다.

• 2♥, 5♥, 5♠, 7♠

- 와 같이 정렬이 될 수도 있고

위에서 뒤늦게, 안 뒤늦게 ...

• 2♥, 5♠, 5♥, 7♠

- 와 같이 정렬이 될 수도 있다.

- 같은 것이 있는 경우에 정렬하기 전의 순서가 유지되는 정렬 알고리즘을 Stable Sorting 알고리즘이라고 한다.

# Stable Sorting

15

## Stable Sorting

- 시간복잡도가  $N \lg N$  인 정렬 알고리즘 중에는 병합 정렬(Merge Sort)가 있다.
- STL에는 `stable_sort` 알고리즘을 사용하면 된다.

$N^2 \Rightarrow$  Bubble Sort  
거품 정렬

# 나이순 정렬

<https://www.acmicpc.net/problem/10814>

- 온라인 저지에 가입한 사람들의 나이와 이름이 가입한 순서대로 주어진다.
- 회원들을 나이가 증가하는 순으로, 나이가 같으면 먼저 가입한 사람이 앞에 오는 순서로 정렬하는 문제
- 가입한 순서는 입력으로 들어오지 않기 때문에, 따로 저장해줘야 한다.

```
struct Person {  
    int age;  
    string name;  
    int join;  
};
```



# 나이순 정렬

<https://www.acmicpc.net/problem/10814>

- C++ (sort): <https://gist.github.com/Baekjoon/633c9eaf24cd4606ef01>
- C++ (stable\_sort): <https://gist.github.com/Baekjoon/e183e261cb91790a5618>

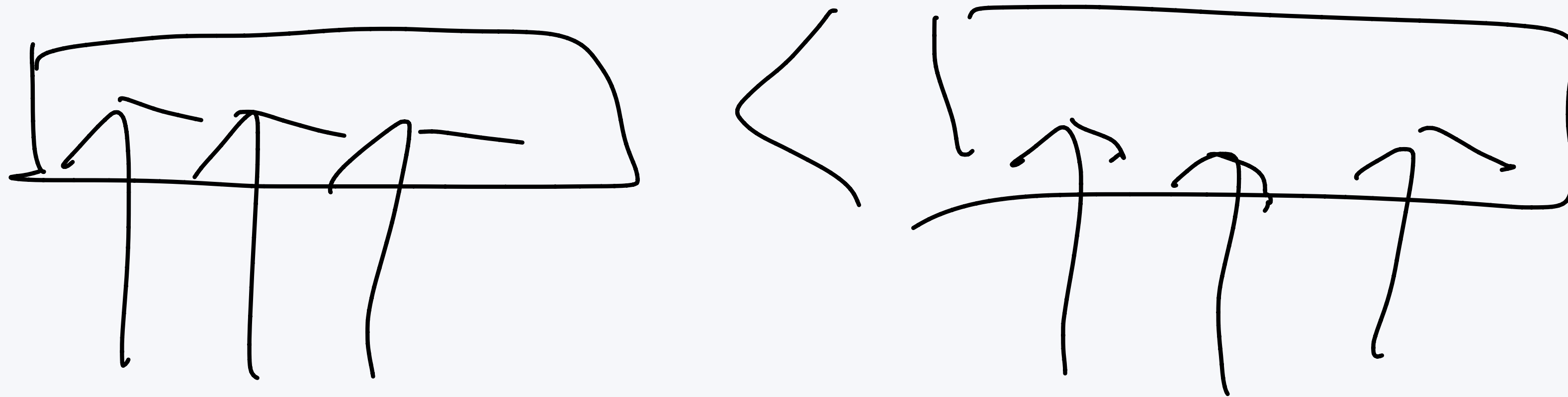
# 국영수

<https://www.acmicpc.net/problem/10825>

- 도현이네 반 학생 N명의 이름과 국어, 영어, 수학 점수가 주어진다.
  - 다음과 같은 조건으로 학생의 성적을 정렬하는 문제
1. 국어 점수가 감소하는 순서로
  2. 국어 점수가 같으면 영어 점수가 증가하는 순서로
  3. 국어 점수와 영어 점수가 같으면 수학 점수가 감소하는 순서로
  4. 모든 점수가 같으면 이름이 사전 순으로 증가하는 순서로

<https://www.acmicpc.net/problem/10825>

- C++: <https://gist.github.com/Baekjoon/112bdd3b8d53d14d0f35>
- C++ (tuple 사용): <https://gist.github.com/Baekjoon/cfbc58f340f2043213d0>



# 수 정렬하기 3

<https://www.acmicpc.net/problem/10989>

- N개의 수를 정렬하는 문제 ( $1 \leq N \leq 10,000,000$ )
- 입력으로 주어지는 수는 10,000보다 작거나 같은 자연수이다.
- $O(N + 10,000)$  만에 풀 수 있다.
- $\text{cnt}[i] =$  입력으로 들어온  $i$ 의 개수로 풀 수 있다.

$\text{cnt}[i] =$  ①

1 ① 100000

$\text{cnt}[i]$

# 수 정렬하기 3

<https://www.acmicpc.net/problem/10989>

- C++: <https://gist.github.com/Baekjoon/5413a3566a5de64fffa9>

# 정렬 응용하기

---

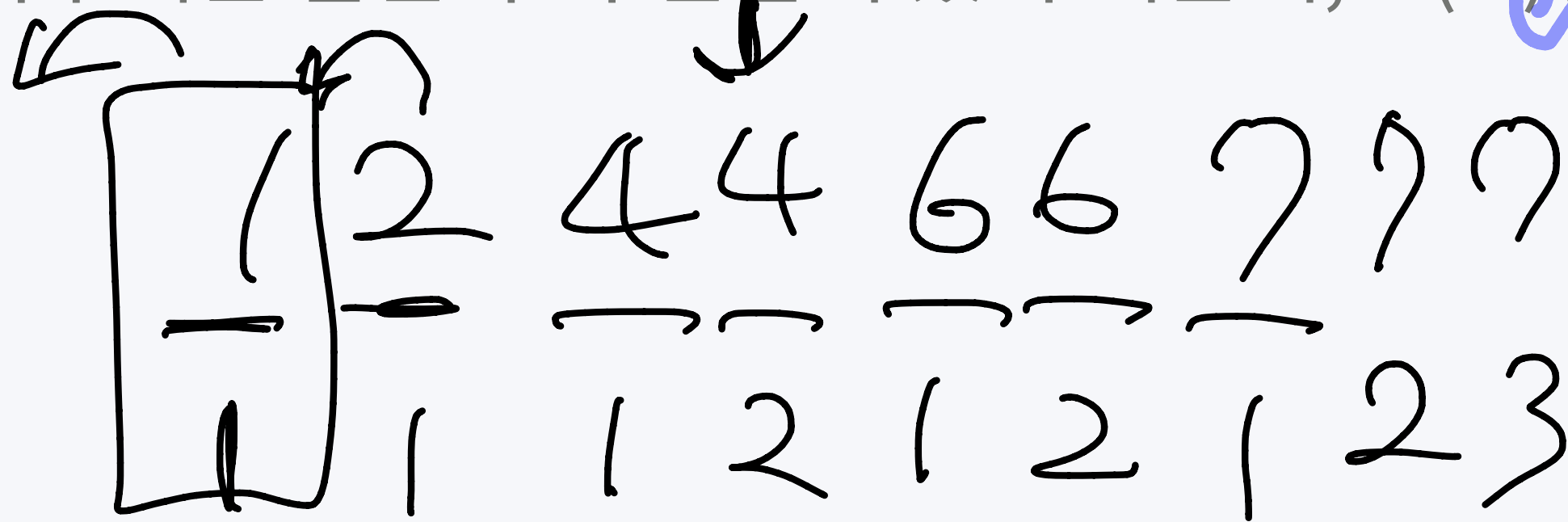
# 카드

<https://www.acmicpc.net/problem/11652>

cnt = 1

- 준규가 가지고 있는 카드가 주어졌을 때, 가장 많이 가지고 있는 정수를 구하는 문제
- 정렬한 이후에 문제를 풀면 된다.
- 정렬을 한 이후에는 같은 수가 인접해 있기 때문에,  $O(N)$ 만에 문제를 풀 수 있다.

ans



cnt = 3

```
for (i = 1; i < n; i++) {  
    if (a[i-1] != a[i]) {  
        if (ans < cnt)  
            ans = cnt;  
        cnt = 1;  
    } else {  
        cnt++;  
    }  
}
```

# 카드

<https://www.acmicpc.net/problem/11652>

- C++(sort): <https://gist.github.com/Baekjoon/fc6f6971496883a7096f>
- C++(map): <https://gist.github.com/Baekjoon/c3ab7b3d1c2f9429c914>

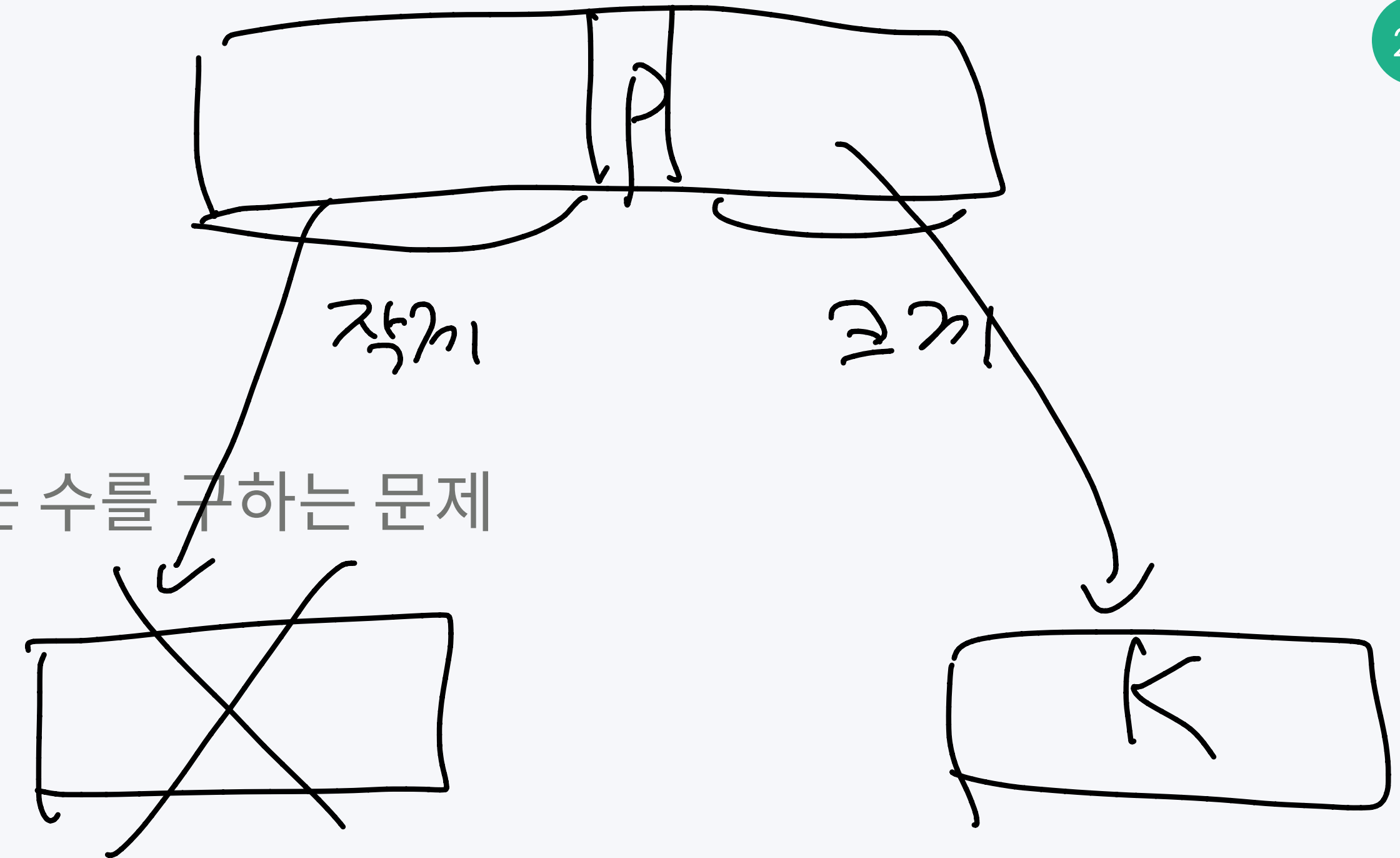


# K번째 수

<https://www.acmicpc.net/problem/11004>

- 수  $N$ 개  $A_1, A_2, \dots, A_N$ 이 주어진다.
- $A$ 를 오름차순 정렬했을 때, 앞에서부터  $K$ 번째 있는 수를 구하는 문제

(K)



# K번째 수

<https://www.acmicpc.net/problem/11004>

- 퀵 소트를 응용해 앞에서 부터 K만 정렬하는 방식으로 풀 수 있다.

# K번째 수

STL

27

<https://www.acmicpc.net/problem/11004>

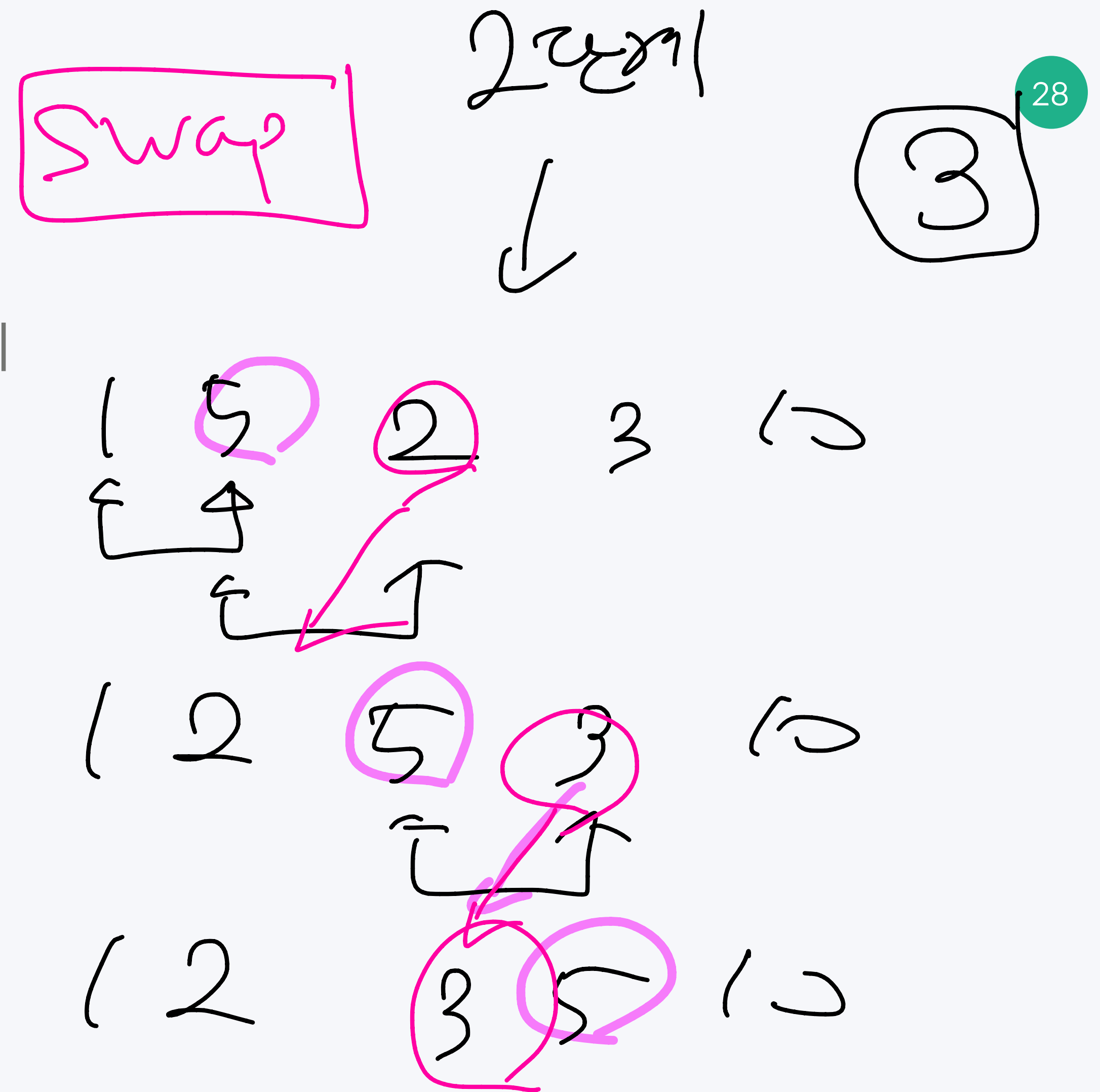
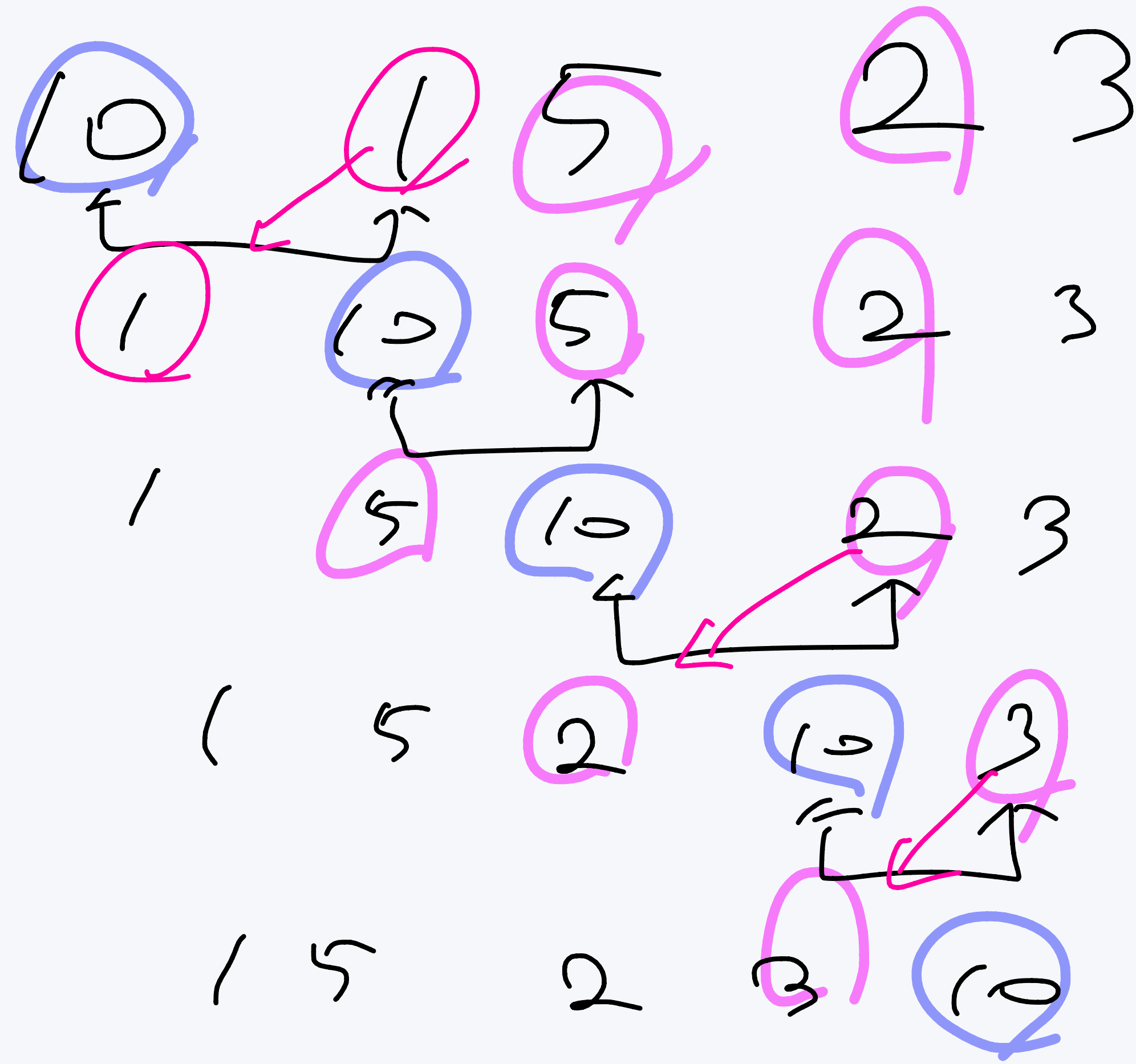
- C++: <https://gist.github.com/Baekjoon/fd3b8be64b4dfa723da6>

nth\_element

# 버블 소트

<https://www.acmicpc.net/problem/1377>

- 버블 소트가 총 몇 단계로 이루어져 있는지 구하는 문제



# 버블 소트

<https://www.acmicpc.net/problem/1377>

1 단계  
2 단계

3 단계

29

- C++: <https://gist.github.com/Baekjoon/d9abf98a1152de2546af7c2c92aaa0f9>

