정렬(1)

광운대학교
소프트웨어학부
김석희

2023.10.18

정렬

어떤 데이터들이 주어졌을 때,
이를 정해진 순서대로 나열하는 문제

정렬을 하는 이유

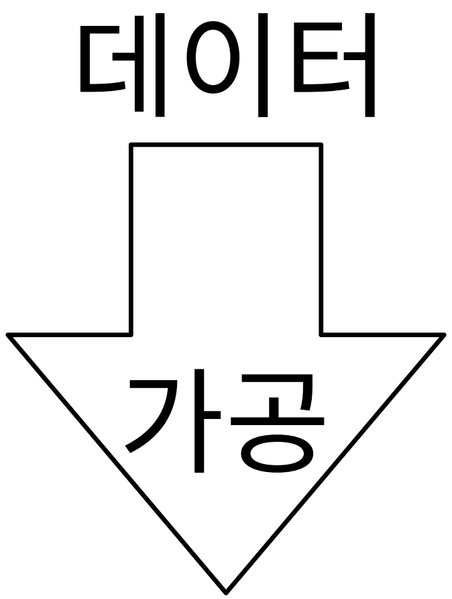
데이터

정렬을 하는 이유

데이터

특정 분야에서 **관측된 아직 가공되는 않은 것**
사실인 것처럼 관측되지만 **오류**나 **잡음**을 포함 가능

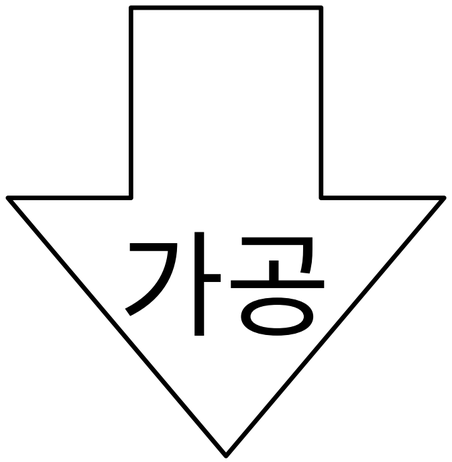
정렬을 하는 이유



데이터를 **가공**하여 어떤 **목적**이나 **의미**를
갖도록 한 것

정렬을 하는 이유

데이터



정보

정렬을 하는 이유

가공

탐색

데이터들 속에서 내가 원하는 값을 찾아내는 것

정렬을 하는 이유

탐색

데이터들 속에서 내가 원하는 값을 찾아내는 것

정렬을 하는 이유

탐색

데이터들 속에서 내가 원하는 값을 찾아내는 것

전제조건

데이터가 정렬이 되어 있어야 함

정렬을 하는 이유

데이터가 정렬이 되어 있지 않을 때 순차 탐색

$$O(N)$$

데이터가 정렬이 되어 있을 때 이분 탐색

$$O(\log N)$$

정렬을 하는 이유

100억개의 데이터

데이터가 정렬이 되어 있지 않을 때 순차 탐색

O(N)

최대 100억번 탐색

데이터가 정렬이 되어 있을 때 이분 탐색

O(log N)

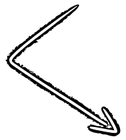
최대 34번 탐색

2^34 > 10000000000

정렬을 하는 이유

https://youtu.be/kPRA0W1kECg?si=0rOm7FsgAmUamdaO

- 정렬의 큰 개념 확립
- 정렬을 배움으로써 알고리즘의 본질을 '더' 깨우침
- '더' 어려운 문제를 풀기위한 기초



코드가 적당히 어려운데다,
결과를 알아 코드의 방향성을 알고,
시각화가 가능하기 때문

O(N^2)인 정렬

버블 정렬

삽입 정렬

선택 정렬

판별 및 비교

교환

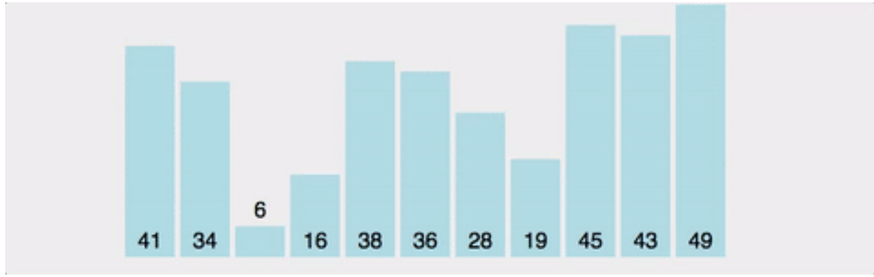
버블 정렬

우측 값이 자기보다 작으면 교환하는 정렬

버블 정렬

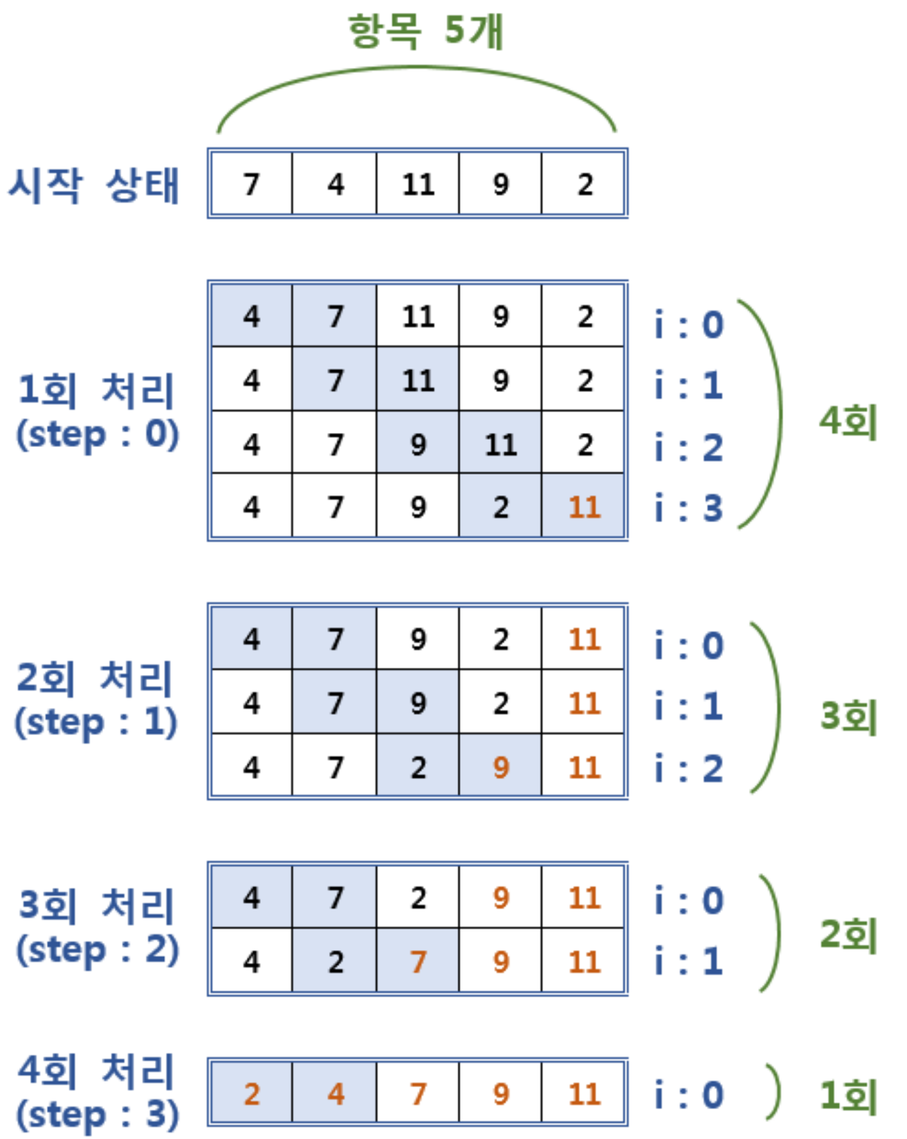
https://youtu.be/Cq7SMsQBEUw?si=42QUcuTWKv1-9ZkW

1. 좌측 값이 자기 보다 크면 교환한다.
2. (반복)한다.
3. 끝에 도달하면 다시 처음부터



버블 정렬

1. 좌측 값이 자기 보다 크면 교환한다.
2. (반복)한다.
3. 끝에 도달하면 다시 처음부터



1. 좌측 값이 자기 보다 크면 교환한다

```
if (value[j] > value[j + 1]) {
    int temp = value[j];
    value[j] = value[j + 1];
    value[j + 1] = temp;
}
```

2. (반복)한다.

```
for (int j = 0; j < i; j++) {
    if (value[j] > value[j + 1]) {
        int temp = value[j];
        value[j] = value[j + 1];
        value[j + 1] = temp;
    }
}
```

3. 끝에 도달하면 다시 처음부터

```
for (int i = value_size - 1; i > 0; i--) {
    for (int j = 0; j < i; j++) {
        if (value[j] > value[j + 1]) {
            int temp = value[j];
            value[j] = value[j + 1];
            value[j + 1] = temp;
        }
    }
}
```

버블 정렬

```cpp
#include <iostream>
using namespace std;

int value[8] = {5, 6, 4, 3, 2, 7, 1, 8};

int main() {
    int value_size = 8;

    for (int i = value_size - 1; i > 0; i--) {
        for (int j = 0; j < i; j++) {
            if (value[j] > value[j + 1]) {
                int temp = value[j];
                value[j] = value[j + 1];
                value[j + 1] = temp;
            }
        }
    }

    for(int i = 0 ; i < 8; i++){
        cout << value[i] << " ";
    }
    return 0;
}
```

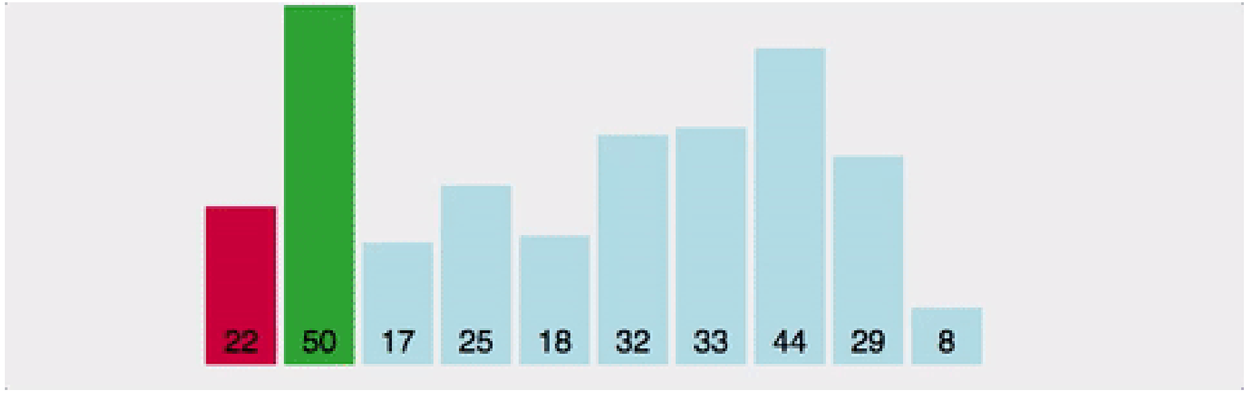
문제

```cpp
#include <iostream>
using namespace std;

int value[8] = {5, 6, 4, 3, 2, 7, 1, 8};

void bubble_sort(int *arr, int arr_size) {}

int main() {
    int value_size = 8;

    bubble_sort(value, value_size);

    for (int i = 0; i < 8; i++) {
        cout << value[i] << " ";
    }
    return 0;
}
```

선택 정렬

https://youtu.be/92BfuxHn2XE?si=KsM6cvqM4BWxuXHX

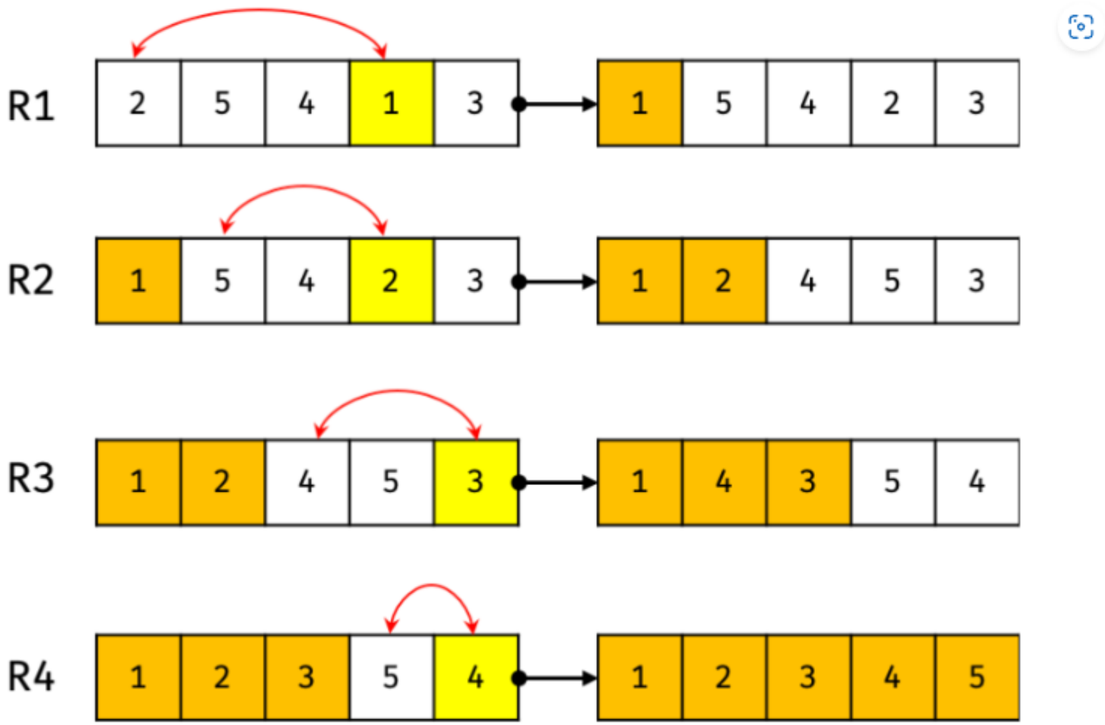
1. 주어진 배열에서 최솟값을 찾는다.
2. 그 최솟값과 맨 앞에 위치한 값을 교체한다.(swap)
3. 맨 앞에 위치한 값을 제외한 나머지 배열에 대해서 같은 방법으로 교체한다.



선택 정렬

https://youtu.be/92BfuxHn2XE?si=KsM6cvqM4BWxuXHX

1. 주어진 배열에서 최솟값을 찾는다.
2. 그 최솟값과 맨 앞에 위치한 값을 교체한다.(swap)
3. 맨 앞에 위치한 값을 제외한 나머지 배열에 대해서 같은 방법으로 교체한다.



선택 정렬

https://youtu.be/92BfuxHn2XE?si=KsM6cvqM4BWxuXHX

1. 주어진 배열에서 최솟값을 찾는다.

```
int minIdx = i;
for (int j = i + 1; j < value_size; j++) {
    if (value[j] < value[minIdx])
        minIdx = j;
}
```

2. 그 최솟값과 맨 앞에 위치한 값을 교체한다. (swap)

```
int temp = value[i];
value[i] = value[minIdx];
value[minIdx] = temp;
```

3. 맨 앞에 위치한 값을 제외한 나머지 배열에 대해서 같은 방법으로 교체한다.

```
for (int i = 0; i < value_size; i++) {
    int minIdx = i;
    for (int j = i + 1; j < value_size; j++) {
        if (value[j] < value[minIdx])
            minIdx = j;
    }
    int temp = value[i];
    value[i] = value[minIdx];
    value[minIdx] = temp;
}
```

선택 정렬

```cpp
#include <iostream>
using namespace std;

int value[8] = {5, 6, 4, 3, 2, 7, 1, 8};

void selection_sort(int *arr, int arr_size) {}

int main() {
    int value_size = 8;

    for (int i = 0; i < value_size; i++) {
        int minIdx = i;
        for (int j = i + 1; j < value_size; j++) {
            if (value[j] < value[minIdx])
                minIdx = j;
        }
        int temp = value[i];
        value[i] = value[minIdx];
        value[minIdx] = temp;
    }

    for (int i = 0; i < 8; i++) {
        cout << value[i] << " ";
    }
    return 0;
}
```

선택 정렬

```cpp
#include <iostream>
using namespace std;

int value[8] = {5, 6, 4, 3, 2, 7, 1, 8};

void selection_sort(int *arr, int arr_size) {}

int main() {
    int value_size = 8;

    selection_sort(value, value_size);

    for (int i = 0; i < 8; i++) {
        cout << value[i] << " ";
    }
    return 0;
}
```

삽입 정렬

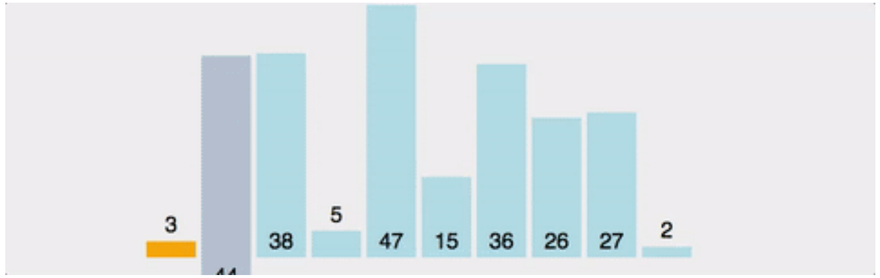
https://youtu.be/8oJS1BMKE64?si=_QrzWbLZb0CGR97R

0. 두번째 수부터 시작한다

1. 정렬 리스트의 오른쪽에 있는 정렬되지 않은 수(인덱스 1번 위치의 수)는 자신의 앞에 있는 수와 크기를 비교하여 앞쪽 수가 더 크면 자리를 바꾼다.

2. 인덱스 2번 위치의 수도 왼쪽으로 전진 하면서 자기 자리를 찾는다.

3. 더 이상 정렬된 대상 원소가 없을 때까지 반복한다.



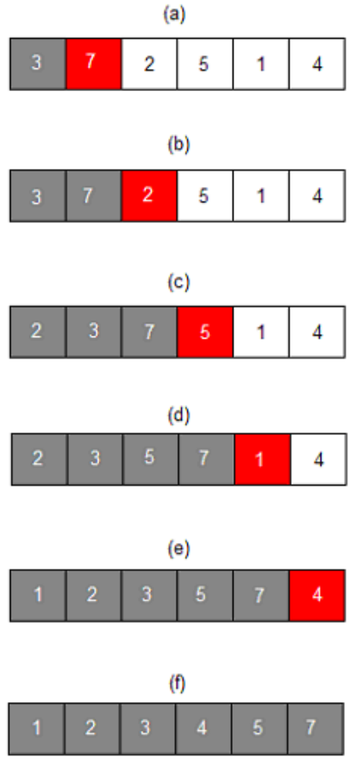
삽입 정렬

0. 두번째 수부터 시작한다

1. 정렬 리스트의 오른쪽에 있는 정렬되지 않은 수(인덱스 1번 위치의 수)는 자신의 앞에 있는 수와 크기를 비교하여 앞쪽 수가 더 크면 자리를 바꾼다.

2. 인덱스 2번 위치의 수도 왼쪽으로 전진 하면서 자기 자리를 찾는다.

3. 더 이상 정렬된 대상 원소가 없을 때까지 반복한다.



두번째 수부터 시작한다

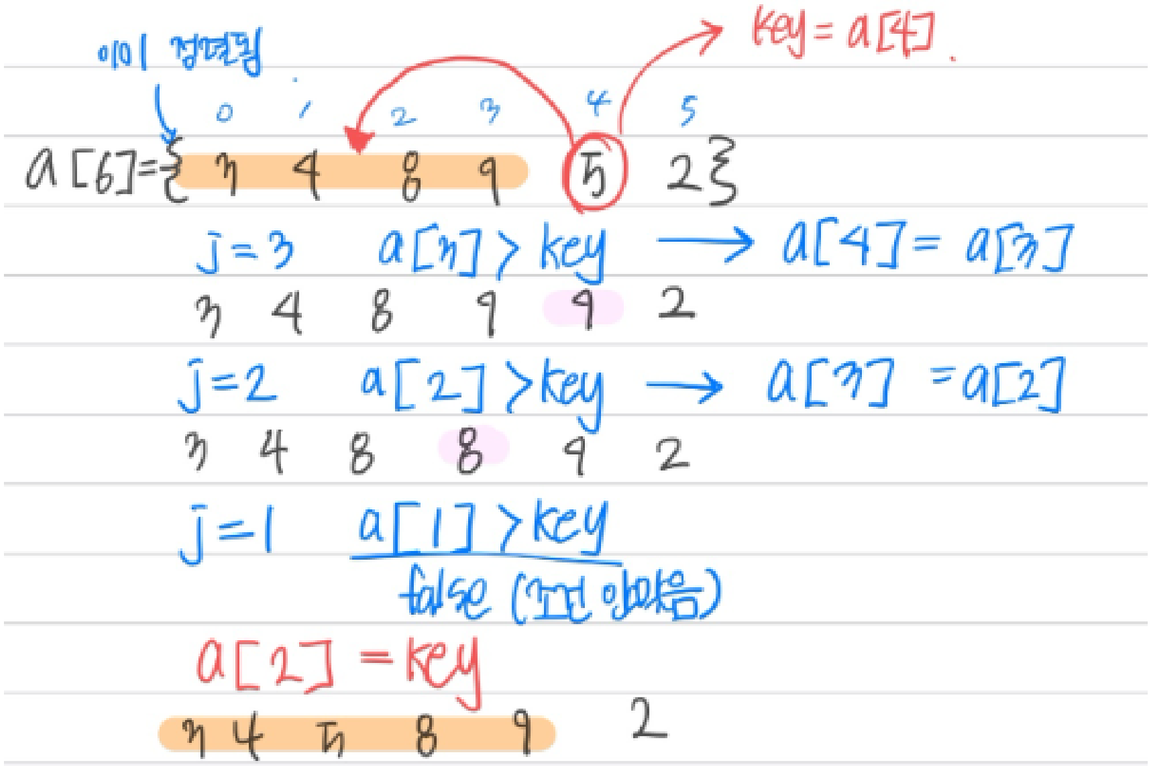
```
for (int i = 1; i < value_size; i++) {
```

삽입 정렬

정렬 리스트의 오른쪽에 있는 정렬되지 않은 수는 자신의 앞에 있는 수와 크기를 비교하여 앞쪽 수가 더 크면 자리를 바꾼다.
인덱스 2번 위치의 수도 왼쪽으로 전진 하면서 자기 자리를 찾는다.

```
    for (j = i - 1; j >= 0; j--) {
        if (value[j] > temp)
            value[j + 1] = value[j];
        else
            break;
    }
    value[j + 1] = temp;
}
```

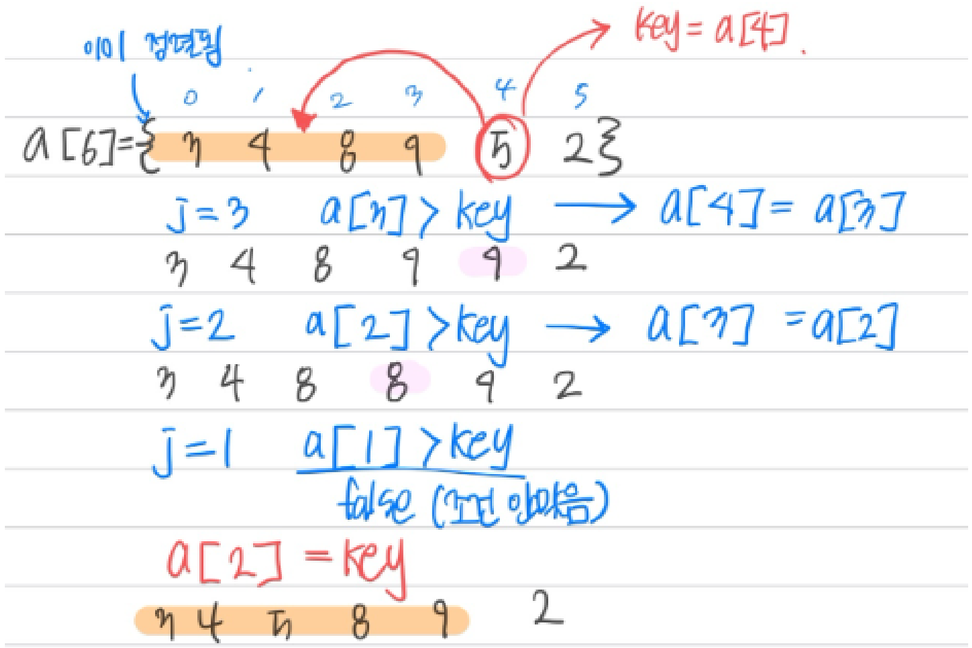
삽입 정렬



5를 정렬하려고 할때 설명

삽입 정렬

```cpp
#include <iostream>
using namespace std;

int value[8] = {5, 6, 4, 3, 2, 7, 1, 8};

void insert_sort(int *arr, int arr_size) {}

int main() {
    int value_size = 8;
    int j;
    for (int i = 1; i < value_size; i++) {
        int temp = value[i];
        for (j = i - 1; j >= 0; j--) {
            if (value[j] > temp)
                value[j + 1] = value[j];
            else
                break;
        }
        value[j + 1] = temp;
    }

    for (int i = 0; i < 8; i++) {
        cout << value[i] << " ";
    }
    return 0;
}
```



5를 정렬하려고 할때 설명

문제

```cpp
#include <iostream>
using namespace std;

int value[8] = {5, 6, 4, 3, 2, 7, 1, 8};

void insert_sort(int *arr, int arr_size) {}

int main() {
    int value_size = 8;
    int j;

    insert_sort(value, value_size);

    for (int i = 0; i < 8; i++) {
        cout << value[i] << " ";
    }
    return 0;
}
```

문제

boj.kr/9946,
boj.kr/1427,
boj.kr/2822,
boj.kr/5648.