정렬은 가장 기본적인 자료구조 알고리즘이다.

같이 묶은 이유는 시간복잡도가 O(N2)이기 떄문이다.

1. 선택정렬은 가장 작은 수를 찾고 맨 앞으로 보내는 정렬방법으로

가장 작은 값과 그 위치를 바꾸며, 가장 작은 것을 찾고 앞으로 보낸다

1 10 5 8 7 6 4 3 2 9

1이 정렬이 완료됐으니 10부터 9중 가장 작은수를 탐색한다. 2가 발견된다.

1다음의 숫자(10)과 2를 바꾼다.

1 2 5 8 7 6 4 3 10 9

5와 3을 바꾼다

1 2 3 8 7 6 4 5 10 9

이를 반복하면 정렬이 완료된다.

하지만 시행의 횟수가 첫 번쨰 (10번 탐색) 이후 두 번째도 탐색(9번 탐색) 등등 시행하면

10 + 9 + 8 +7 + \*\*\*\*+ 1 즉 N(N-1)/2 횟수가 나올것이며 이를 빅오표기법으로 나타내면 O(N2)이 나옴

시간복잡도는 O(N2)으로 좋지 않다.

굉장히 오래걸려 효율성이 좋지않다.

2. 버블정렬

버블정렬은 옆에 있는 값과 비교하여 작은 값을 앞으로 보냄, 선택정렬보다 좋지 않으며 가장 느리고 비효율적이라고 할 수 있다.

1 10 5 8 7 6 4 3 2 9

1과 10을 비교하여 1이 작으니 1을 앞으로 보냄

이후 10과 5를 비교하여 작은 5를 앞으로 보냄

1 5 10 8 7 6 4 3 2 9

이제 또 (1,5), (5,10), (10,8)을 비교 하닥 8이 더 작으니 8을 보냄

1 5 8 10 7 6 4 3 2 9

또 서로서로 찾다 보면 정렬이 완료된다.

복잡도 또한 10 번 하고 정렬되면 9번 하고 쭉쭉하다보니 O(N2)이 나온다.

하지만 스와핑이 굉장히 많아 O(N2) 중 가장 느리다고 할 수 있다.

3. 삽입정렬

삽입정렬도 O(N2)의 시간복잡도를 가지며 각 숫잘ㄹ 적절한 위치에 삽입하는 방법으로 필요할 때만 위치를 바꾼다.

1 10 5 8 7 6 4 3 2 9

1은 그대로 두고 10이 들어갈 위치를 찾는다 \_ 1 \_ 10 이니 10은 그대로 들어간다.

이후 5가 들어갈 위치를 찾는다 \_ 1 \_ 10 \_ 5 라고 보면 5는 1과 10 사이숫자이니 1과 10 사이로 들어간다.

1 5 10 8 7 6 4 3 2 9

8의 위치를 찾아보자, 8은 5와 10 사이로 들어간다

1 5 8 10 7 6 4 3 2 9

쭉쭉 반복하다보면 정렬이 완료된다.

이 또한 첫 번쨰 10번 하고 두 번째 9번 쭉쭉 하다보니 시간복잡도는 O(N2)이 나온다.

하지만 만약 정렬이 거의 완료된 경우에는 엄청나게 빠르게 진행이 된다. 자리를 찾았는데 스와핑이 일어나지 않는 것이다. 이럴경우에는 O(N2)이지만 엄청나게 빠르게 진행이 된다.

반면에 정렬이 개판인 경우에는 엄청나게 느려진다

BEST – 정렬이 거의 완료된 경우

Worst – 정렬이 개판이여서 스와핑이 많이 일어나는 경우