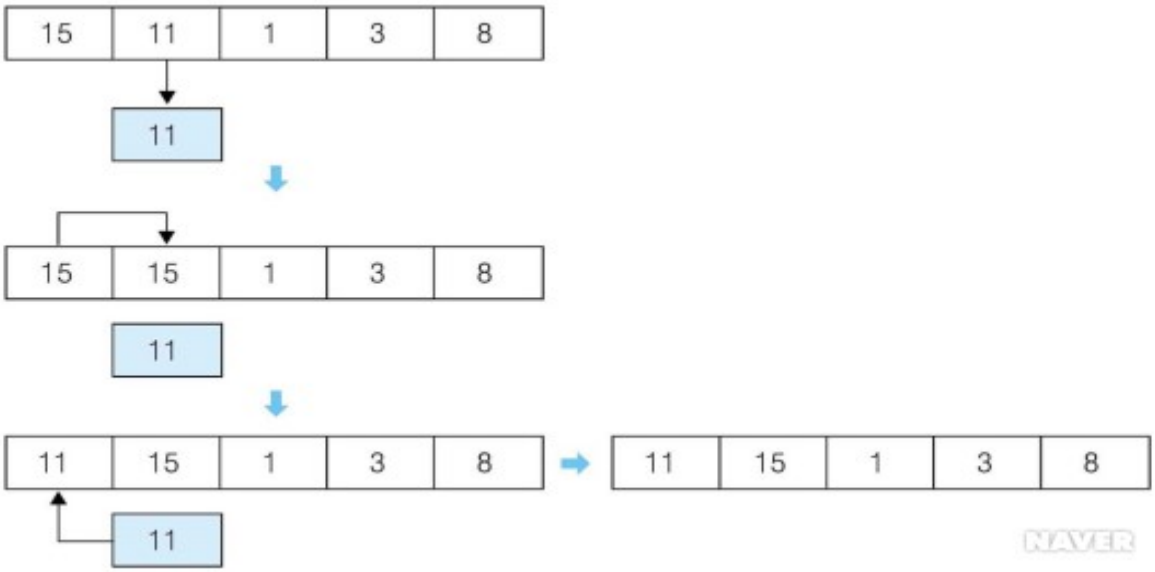
버블 정렬(Bubble sort)

서로 이웃한 데이터들을 비교하며 가장 큰 데이터를 가장 뒤로 보내며 정렬하는 방식



배열의 앞 인덱스부터 순차적으로 15와 11을 비교하여 더 작은 수인 11을 15보다 앞에 위치시키기 위해 11을 잠시 다른 변수에 담아두었다가 15를 뒤 인덱스에 저장한 후 11을 원래 15가 있던 인덱스에 저장하고 이중 for문을 이용하여 모든 수를 정렬시킨다. 하지만 정렬이 다 된 후에도 for문이 작동하기 때문에 효율성의 문제가 있다. 시간 복잡도는 n의 정수를 가진 배열을 이중 for문을 이용하여 정렬하기 때문에 O(n^2)이 될 것이다.

출처 : https://terms.naver.com/entry.naver?docId=2270437&cid=51173&categoryId=51173

```
//오름차순 정렬
    for (int i = 0; i < COUNT - 1; i++)
    {
        for (int j = 0; j < COUNT - 1 - i; j++)
        {
            if (data[j] > data[j + 1])
            {
                temp        = data[j];
                data[j]     = data[j + 1];
                data[j + 1] = temp;
            }
        }
    }
```

Q1. 버블 정렬을 내림차순으로 정렬해봅시다. (검색 지양)

도전문제.

1) [2,5,1,6,3]의 데이터는 버블 정렬을 이용할 때 몇번의 숫자 이동이 일어날까요? 예를들어 1은 맨 앞으로 가기 위해 2번을 움직일 것입니다.

2) 그리고 n가지(어느 숫자든 상관없음)의 정수 배열을 버블 정렬로 정렬할 때 정렬 횟수(반복 횟수가 아닌 숫자의 이동 횟수)를 카운트해주는 프로그램을 작성해봅시다. 실제 카운트 횟수와 for문의 반복 횟수를 비교해보면서 효율성도 고려해봅시다.

3) 제일 처음 위치했던 인덱스에 있던 숫자가 몇 번을 이동했는지(1이란 숫자가 몇번 이동했고 3이란 숫자가 몇번 이동했고..) 각각을 구해주는 프로그램을 작성해봅시다. 배열을 이용하여 각각 카운트하는게 제일 간단하겠죠.