ILE3-018 정렬

임의의 순서로 배치된 데이터를 순서대로 만드는 작업

버블 정렬

서로 인접한 두개의 항목의 비교를 수행하는 정렬

선택 정렬

데이터 중 가장 작은 항목을 찾아서 현재 위치와 교환하는 정렬

최악 시간복잡도	$O(n^2)$ 비교, $O(n)$ 교환
최선 시간복잡도	$O(n^2)$ 비교, $O(n)$ 교환
평균 시간복잡도	$O(n^2)$ 비교, $O(n)$ 교환

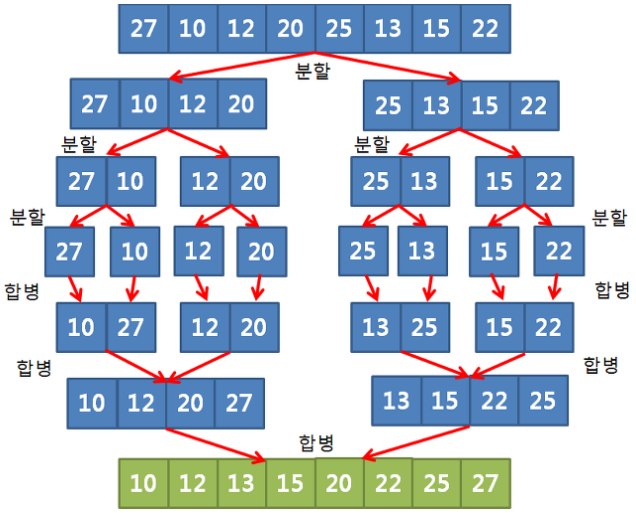
삽입 정렬

앞에서부터 차례대로 이미 비교를 진행하면서 자신의 위치를 찾아 삽입함으로써 정렬
정렬이 많이 되어 있을 수록 성능이 좋음
삽입이 자주 발생하므로 배열에서는 성능이 낮음

최악 시간복잡도	O(n^2) 비교 및 교환
최선 시간복잡도	O(n) 비교, O(1) 교환
평균 시간복잡도	O(n^2) 비교 및 교환

합병 정렬

데이터를 나누어 정렬한 이 후 병합을 수행하며 정렬
분할-정복 알고리즘(Divide and conquer algorithm)을 사용



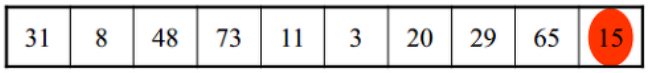
최악 시간복잡도	O(n log n)
최선 시간복잡도	O(n log n)
평균 시간복잡도	일반적으로, O(n log n)

퀵 정렬

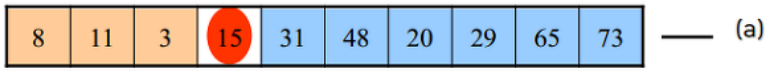
선택한 값을 기준으로 좌측으로는 선택 값보다 작은 값만, 우측으로는 선택 값보다 큰 값을 배치를 수행하는 정렬
이미 정렬이 되어 있는 데이터의 경우 성능이 나쁨
분할-정복 알고리즘(Divide and conquer algorithm)을 사용

최악 시간복잡도	$O(n^2)$
최선 시간복잡도	$O(n \log n)$
평균 시간복잡도	$O(n \log n)$

정렬할 배열이 주어짐. 마지막 수를 기준으로 삼는다.

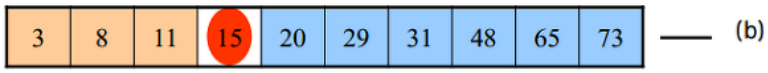


기준보다 작은 수는 기준의 왼쪽에 나머지는
기준의 오른쪽에 오도록 재배치한다



(a)

기준(31) 왼쪽과 오른쪽을 각각 독립적으로 정렬한다 (정렬완료)



(b)

정렬의 안정성

비교 값이 같은 값을 지니고 있을 때, 해당 값이 순서가 유지 여부

정렬 전 데이터

```
(20, '아자차카')
(14, '가나다라')
(5, '바사아자')
(14, '나다라마')
(15, '다라마바')
```

안정한 정렬

```
(5, '바사아자')
(14, '가나다라')
(14, '나다라마')
(15, '다라마바')
(20, '아자차카')
```

불안정한 정렬

```
(5, '바사아자')
(14, '나다라마')
(14, '가나다라')
(15, '다라마바')
(20, '아자차카')
```

과제

삽입 정렬 알고리즘을 구현해 보자

구현 방법

1. 정렬이 완료 된 Index를 지정, 처음은 0
2. 정렬을 수행 할 Index를 지정, 처음은 1
3. 정렬을 수행 할 Index가 배열의 길이가 될 때 까지 아래 작업을 반복
   a. 정렬이 완료 된 Index와 값을 비교한다
      i. 정렬을 수행 할 Index가 더 큰 값인 경우, 정렬을 수행 할 Index를 1 증가
      ii. 정렬을 수행 할 Index가 더 작은 경우 앞으로 탐색하면서 작거나 같은 값을 찾고, 해당 다음 Index에 비교 값을 삽입한다
      그리고 정렬이 완료된 Index, 정렬을 수행 할 Index를 각각 1씩 증가한다

테스트 방법

1. 임의의 100,000개의 값을 지니는 List or 연결리스트를 만들고 해당 값의 정렬 시간 측정
2. 1 ~ 100,000까지 값을 지니는 List or 연결리스트를 만들고 해당 값의 정렬 시간 측정