<2021-2. 알고리즘 및 실습> - 과제2(OJ)

※ 입출력에 대한 안내

- 특별한 언급이 없으면 문제의 조건에 맞지 않는 입력은 입력되지 않는다고 가정하라.
- 입출력 예시에서 → 이 후는 각 입력과 출력에 대한 설명이다.
- 소스 코드 체크 후 감점이 될 수 있음. (정해진 규칙을 지키지 않으면 감점)

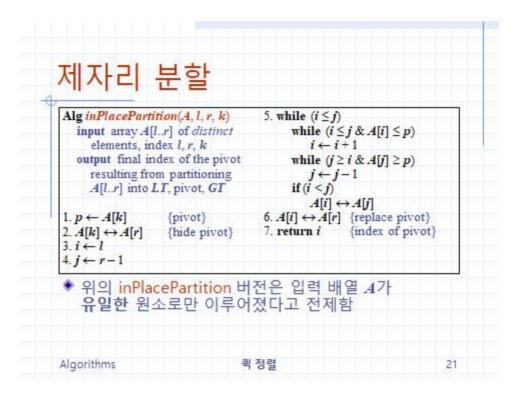
[문제 1] (퀵 정렬) n개의 양의 정수(중복 가능)를 입력받아 정렬하는 프로그램을 작성하시오. 정렬은 아래에 명시된 퀵 정렬을 구현하여 사용한다.

- 구현해야 할 퀵 정렬 알고리즘:
 - 크기가 n인 배열을 **동적 할당**하여, 입력된 양의 정수 저장(입력 정수는 중복 가능).
 - 기준값(pivot)을 정할 때, 다음 방법을 이용한다:
 - (1) 부리스트의 수들 중 **1**개의 위치를 무작위로 선택(즉, 입력배열의 l과 r 사이의 **1**개 위치를 랜덤하게 선택)하여 pivot의 위치로 정한다.
 - (2) 아래 힌트처럼 pivot의 위치를 정하는 findPivot 함수를 별도 작성해서 이 함수의 반환값을 inPlacePartition 함수가 사용하는 방식을 추천한다.
 - inPlacePartition의 반환값은 두 인덱스 (a, b)로, 그 의미는 분할 결과, 배열의 I번째 수부터 a 1번째 수는 pivot보다 작은 값을, 배열의 a번째부터 b번째 수는 pivot과 같은 값을, b + 1번째부터 r번째 수는 pivot보다 큰 값을 가진다는 것이다(즉, 이후 호출되는 재귀함수는 I부터 a 1까지 부배열에 대해서와 b + 1부터 r까지의 부배열에 대해서 다루고, pivot과 같은 값들인 a부터 b번째 값들은 재귀에서 제외된다).

입력 예시 1		출력 예시 1	
3	→ n	□1 4 9	→ 정렬 결과
4 9 1			
입력 예시 2		출력 예시 2	
8	→ n	□3 8 20 29 31 48 65 73	→ 정렬 결과
73 65 48 31 29 20 8 3			

힌트: 다음은 제자리 퀵 정렬 알고리즘이다. 아래 내용을 참고하여 구현하되 위의 명시된 조건에 따라 pivot을 정한다.

1) 유일한 원소인 경우의 제자리 분할을 사용하면 실습 OK



2) 과제 2는 (a, b)를 가지는 아래 inPlaceQuickSort버젼으로 inPlacePartition을 만들고 이를 활용한 inPlaceQUickSort를 진행하여라.

Accept가 되는 게 중요한 게 아니라 방법을 제대로 했는지가 중요.

```
Alg inPlaceQuickSort(L, l, r)
   input list L, rank l, r
   output list L with elements of rank from l to r rearranged in increasing order

1. if (l ≥ r)
        return
2. k ← findPivot(L, l, r)
3. (a, b) ← inPlacePartition(L, l, r, k) {elements from a to b are same to pivot}
4. inPlaceQuickSort(L, l, a - 1) {elements from l to a - 1 are smaller than pivot}
5. inPlaceQuickSort(L, b + 1, r) {elements from b + 1 to r are bigger than pivot}
```