<알고리즘 실습> - 우선순위큐(선택 & 삽입 정렬)

※ 입출력에 대한 안내

- 특별한 언급이 없으면 문제의 조건에 맞지 않는 입력은 입력되지 않는다고 가정하라.
- 특별한 언급이 없으면, 각 줄의 맨 앞과 맨 뒤에는 공백을 출력하지 않는다.
- 출력 예시에서 □는 각 줄의 맨 앞과 맨 뒤에 출력되는 공백을 의미한다.
- 입출력 예시에서 ↦ 이 후는 각 입력과 출력에 대한 설명이다.

※ 참고: 이번 주의 **주요 실습 내용은 두 알고리즘의 성능 분석에 대한 [문제 3]** 이다.

[문제 1] (선택 정렬) N개의 양의 정수를 입력(중복 가능)받아, 아래에서 설명하는 선택 정렬을 이용하여 정렬하는 프로그램을 작성하시오.

○ 구현해야할 선택 정렬 알고리즘 (가장 큰 값을 찾는 버전):
- 크기가 N인 **배열을 동적 할당**하여, 입력된 양의 정수 저장 (입력 정수는 중복 허용)
- **제자리(in place) 정렬** 사용.
  즉, 입력 값 저장을 위한 배열 이외에 O(1)의 추가 공간만 사용
- 배열의 **뒷 부분을 정렬 상태로 유지**하고, 매 반복마다 최대 한 번의 교환 연산만 사용
  (매 반복마다 가장 큰 값을 찾아, 오른쪽부터 채우는 방식으로 정렬)
- 가능하면 교재의 의사코드를 보지 말고 구현해볼 것을 권장

○ 알고리즘 동작 과정 예시 (N=8)

최초 상태:	8	31	48	73	3	65	20	29	
1번째 반복 후:	8	31	48	29	3	65	20	**73**	(73과 29 교환)
2번째 반복 후:	8	31	48	29	3	20	**65**	73	(65와 20 교환)
3번째 반복 후:	8	31	20	29	3	**48**	65	73	(48과 20 교환)
4번째 반복 후:	8	3	20	29	**31**	48	65	73	(31과 3 교환)
5번째 반복 후:	8	3	20	**29**	31	48	65	73	(교환 X)
6번째 반복 후:	8	3	**20**	29	31	48	65	73	(교환 X)
7번째 반복 후:	3	**8**	20	29	31	48	65	73	(8과 3 교환)

입력 예시 1	출력 예시 1
8 ↦ N 8 31 48 73 3 65 20 29	□3 8 20 29 31 48 65 73 ↦ 정렬 결과

입력 예시 2

```
8          ↦ N
73 65 48 31 29 20 8 3
```

출력 예시 2

```
□3 8 20 29 31 48 65 73   ↦ 정렬 결과
```

[문제 2] (삽입 정렬) N개의 양의 정수를 입력(중복 가능)받아, 아래에서 설명하는 삽입 정렬을 이용하여 정렬하는 프로그램을 작성하시오.

○ 구현해야할 삽입 정렬 알고리즘:
- 크기가 N인 **배열을 동적 할당**하여, 입력된 양의 정수 저장 (입력 정수는 중복 허용)
- **제자리(in place) 정렬** 사용.
  즉, 입력 값 저장을 위한 배열 이외에 O(1)의 추가 공간만 사용
- 배열의 **앞부분을 정렬 상태로 유지**
- 가능하면 교재의 의사코드를 보지 말고 구현해볼 것을 권장

○ 알고리즘 동작 과정 예시 (N=7)

최초 상태 :	3	73	48	31	8	11	20	
1번째 반복 후:	3	**73**	48	31	8	11	20	73 삽입
2번째 반복 후:	3	**48**	73	31	8	11	20	48 삽입
3번째 반복 후:	3	**31**	48	73	8	11	20	31 삽입
4번째 반복 후:	3	**8**	31	48	73	11	20	8 삽입
5번째 반복 후:	3	8	**11**	31	48	73	20	11 삽입
6번째 반복 후:	3	8	11	**20**	31	48	73	20 삽입

입력 예시 1

```
7        ↦ N
3 73 48 31 8 11 20
```

출력 예시 1

```
□3 8 11 20 31 48 73   ↦ 정렬 결과
```

입력 예시 2

```
8          ↦ N
73 65 48 31 29 20 8 3
```

출력 예시 2

```
□3 8 20 29 31 48 65 73   ↦ 정렬 결과
```

[문제 3] **(주 실습 내용-OJ 문제 아님)** 아래 절차로 여러 가지 다양한 입력에 대해 선택 정렬과 삽입 정렬의 **실행 시간을 측정 비교**하라.

작성해야할 프로그램:

① 표준 입력으로 정렬할 원소의 개수 N을 입력 받고, 크기가 N인 정수 배열 A와 B를 동적할당 받는다.

② 난수발생 함수(srand, rand 등)를 사용하여 N개의 정수 난수로 배열 A와 B를 동일하게 초기화 한다.

③ 배열 A에 대해서는 '선택 정렬'을, 배열 B에 대해서는 '삽입 정렬'을 수행하고, 시간 측정 함수(clock 등)를 이용하여 각 정렬에 수행된 시간을 표준 출력으로 출력한다.

입력 예시 1	출력 예시 1
100000 ↦ N	0.051289721ms ↦ 선택 정렬 수행 시간 0.054142322ms ↦ 삽입 정렬 수행 시간

실행 시간 비교 분석:

다음과 같이 다양한 입력 데이터에 대해 두 정렬 알고리즘의 시간을 측정하여, 두 정렬의 성능을 비교 분석해보자.

A. 각 정렬의 입력으로 **정렬이 안 된 데이터**가 들어오는 경우

a) 동일한 N으로 여러 번 실험을 하여, 어느 정렬이 더 빠른 지 비교해보자.

b) N이 증가됨에 따라 두 정렬의 실행 시간이 어떤 비율로 증가하는 지 확인해보자.

B. 각 정렬의 입력으로 **정렬된 데이터**가 들어오는 경우.

(참고) 정렬된 데이터로 실험하기 위해서는 ②번과 ③번 사이에, A와 B를 아무 정렬 알고리즘이나 사용해서 정렬시키는 과정을 추가하면 된다. (시간은 ③에서 측정)

a) 동일한 N으로 여러 번 실험을 하여, 어느 정렬이 더 빠른 지 비교해보자.

b) N이 증가됨에 따라 두 정렬의 실행 시간이 어떤 비율로 증가하는 지 확인해보자.

C. 각 정렬의 입력으로 **역순으로 정렬된 데이터**가 들어오는 경우.

(참고) 정렬된 데이터로 실험하기 위해서는 ②번과 ③번 사이에, A와 B를 아무 정렬 알고리즘이나 사용해서 역순으로 정렬시키는 과정을 추가하면 된다. (시간은 ③에서 측정)

a) 동일한 N으로 여러 번 실험을 하여, 어느 정렬이 더 빠른 지 비교해보자.

b) N이 증가됨에 따라 두 정렬의 실행 시간이 어떤 비율로 증가하는 지 확인해보자.