




학교 공부/알고리즘


[BOJ][Python] 백준 2075번 N번째 큰 수

그리버 | 2023. 9. 26. 08:57 | 수정 | 삭제

문제

<https://www.acmicpc.net/problem/2075>

 **2075번: N번째 큰 수**
www.acmicpc.net



주제	힙
시간 / 메모리 제한	1 초 / 12 MB
정답 비율	39.005%

$N \times N$ 의 표에 수 N^2 개가 채워져 있다.

채워진 수의 모든 수는 자신의 한 칸 위에 있는 수보다 크다.

- $N=5$ 일 때의 예

12	7	9	15	5
13	8	11	19	6
21	10	26	31	16
48	14	28	35	25

52	20	32	41	49
----	----	----	----	----

이런 표가 주어졌을 때, N번째 큰 수를 찾는 프로그램을 작성하시오.

(표에 채워진 수는 모두 다르다.)

입력

- 첫째 줄에 $N(1 \leq N \leq 1,500)$ 이 주어진다.
- 다음 N개 줄에는 각 줄마다 N개의 수가 주어진다.
 - 표에 적힌 수는 -10억 보다 크거나 같고, 10억 보다 작거나 같은 정수이다.

출력

- 첫째 줄에 N번째 큰 수를 출력한다.

- 예제 입력 1

```
5
12 7 9 15 5
13 8 11 19 6
21 10 26 31 16
48 14 28 35 25
52 20 32 41 49
```

- 예제 출력 1

```
35
```

풀이

위의 문제를 풀기 위해선 리스트에 대한 정렬이 필요하다.

단순한 정렬로 풀 수 있는 문제이지만, 주의깊게 봐야할 것은 이 문제의 메모리 제한이 12 MB라는 것이다.

처음에는 원초적으로 생각할 수 있는 힙 정렬 풀이로 아래와 같이 풀어보았다.

파이썬은 heapq 라이브러리로 힙 정렬을 구현할 수 있는데,

기본적으로 최소 힙이다.

따라서 아래와 같이 heappush와 heappop에 각각 - 부호를 붙여 최대 힙으로 만들어주었다.

```
from sys import stdin
from heapq import heappush, heappop

N = int(stdin.readline().rstrip())
L = []

for _ in range(N):
    line = list(map(int, stdin.readline().rstrip().split(' ')))
    L.extend(line)

heap = []
for l in L:
    heappush(heap, -l)

for _ in range(N):
    res = -heappop(heap)
    print(res)
```

이렇게 풀었는데 메모리 초과가 발생하였다.

리스트는 L, line, heap 이렇게 세 개가 사용되었다.

이에 쓸 데 없이 사용되는 리스트를 없애도록 수정하였다.

```
from sys import stdin
from heapq import heappush, heappop

N = int(stdin.readline().rstrip())
heap = []

for _ in range(N):
    line = []
    line = list(map(int, stdin.readline().rstrip().split(' ')))
    for l in line:
        heappush(heap, -l)

for _ in range(N):
    if _ == N - 1:
        print(-heappop(heap))
    heappop(heap)
```

이런 식으로 사용되는 리스트를 heap(NxN), line(Nx1) 두 개로 줄여보았지만

마찬가지로 메모리 초과가 발생하였다.

메모리 초과

메모리 초과를 개선할 수 있는 알고리즘이 필요했다.

$N \times N$ 사이즈의 리스트를 한 개만 사용했는데도 메모리 초과가 발생했기 때문에

$N \times N$ 사이즈 리스트는 사용하지 않고 구현해야 한다.

$N \times 1$ 리스트 두 개를 이용해서 문제를 풀어야 하는데,

그러기 위해서는 아직 고려하지 않은 문제의 조건을 하나 더 생각해야 한다.

바로 언제든 N 번째 큰 수를 찾으려면 된다는 것이다.

따라서 우리는 $N \times N$ 개의 수를 모두 고려할 필요가 없고

N 번째 큰 수보다 작은 수와 비교했을 때 작은 수는 저장하지 않고 버리면 된다.

이 방법은 힙 자료구조를 이용해 어렵지 않게 구현할 수 있다.

힙

힙은 모든 노드가 자식보다 작거나 같은 값을 갖는 이진 트리로,

최소 힙을 제공하는 파이썬에서는 가장 작은 요소가 항상 루트인 `heap[0]`이 된다.

그리고 `heap`에 요소를 추가하는 `heappush`을 하면 힙 불변성을 유지하면서 요소를 추가할 수 있다.

`heap`의 가장 작은 항목을 삭제하는 `heappop`을 하면 요소를 삭제할 수 있다.

힙을 이용한다면 아래와 같은 알고리즘으로 문제를 풀 수 있다.

알고리즘

한 줄씩 숫자를 받고 리스트로 저장한다.

그 숫자 리스트를 순회하면서 현재 `heap`의 길이가 N 보다 작다면 일단 `heappush`를 한다.

`heap`의 길이가 N 보다 커지면 비교를 수행한다.

만약 heap의 가장 작은 요소 heap[0]가 숫자보다 작으면,

heappop하여 힙의 가장 작은 요소인 heap[0]을 제거하고 받은 숫자를 heappush한다.

이렇게 heap의 길이를 N개로 유지하면서 heap에 가장 큰 N개의 요소를 남길 수 있다.

이를 계속 반복하면 힙에는 최종적으로 가장 큰 N개의 숫자가 남게된다.

그리고 heap은 N개의 리스트이므로 heappop[0]을 출력하면 N번째 큰 수를 얻을 수 있다.

정답 코드

```
from sys import stdin
from heapq import heappush, heappop

N = int(stdin.readline().rstrip())
heap = []

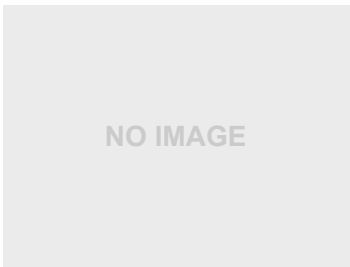
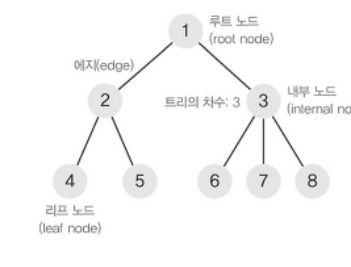
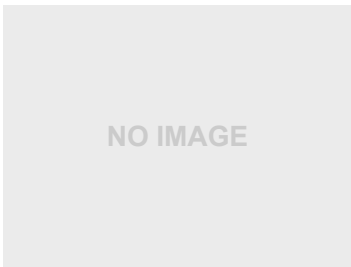
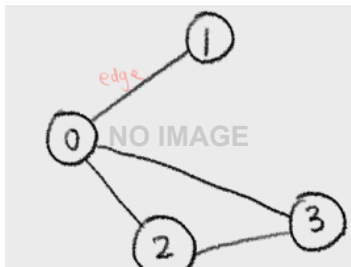
for _ in range(N):
    line = []
    line = list(map(int, stdin.readline().rstrip().split(' ')))
    for l in line:
        if len(heap) < N:
            heappush(heap, l)
        else:
            if heap[0] < l:
                heappop(heap)
                heappush(heap, l)

print(heap[0])
```

제출 번호	아이디	문제	결과	메모리	시간	언어	코드 길이	제출한 시간
67177843	aprilwldnjs	2075	맞았습니다!!	33324 KB	920 ms	Python 3 / 수정	399 B	1분 전

[알고리즘](Python) 그래프 개념과 파이썬 코드 구현 (3)	2023.09.19
[알고리즘](Python) 백준 2696번 중앙값 구하기 - 우선순위 큐 (0)	2023.09.19
[알고리즘](Python) 트리 (1)	2023.09.12
[알고리즘](Python) 백준 10828번 - 스택 (0)	2023.09.12
[알고리즘](Python) 백준 10845번 - 큐 (0)	2023.09.12

'학교 공부/알고리즘' Related Articles



[알고리즘](Python) 그래프 개념과 파이썬 코...

[알고리즘](Python) 백준 2696번 중앙값 구...

[알고리즘](Python) 트리

[알고리즘](Python) 백준 10828번 - 스택

☐ Secret

안녕하세요! 어떤 댓글이든 환영합니다! 📌

댓글달기