

[PRG]프로세스

문제 파악

- 1) 우선순위가 높은 프로세스 먼저 출력.
- 2) location번 째 프로세스가 몇 번째로 출력되는지를 반환.

어려웠던 점

location이 처음 들어온 배열 priorities에서의 인덱스이기 때문에, 우선순위 큐에서 최대 힙으로 정렬하게 되면 이 인덱스를 어떻게 찾아야 할 지가 막막했음.

Hint

- 우선순위 큐에서 현재 출력하려는 값이 원래 배열인 priorities의 인덱스 location번째 값과 일치하는 지 확인한다.

⇒ `priorities[i] == pq.peek()`

```
import java.util.Collections;
import java.util.PriorityQueue;

class Solution {
    public int solution(int[] priorities, int location) {
        int answer = 0;
        PriorityQueue<Integer> pq = new PriorityQueue<>(Collections.reverseOrder()); //최대 힙 우선순위 큐 호출

        for (int i=0; i < priorities.length; i++){
            pq.add(priorities[i]); //우선순위가 담긴 배열을 최대 힙 우선순위 큐에 담는다. => pq의 루트노드는 가장 높은 우선순위
        } pq = [5,4,3,2,1]
        priorities = [3,2,1,4,5]
        while(!pq.isEmpty()){ //우선순위 큐가 빌 때까지 반복
            for (int i = 0; i < priorities.length; i++){ //현재 출력하려는 프로세스가 가장 높은 우선순위가 맞는지 확인하는 과정
                if(priorities[i] == pq.peek()){ //현재 출력하려는 프로세스의 우선순위가 pq의 루트노드인지 확인하고 맞다면
                    if (i == location){ // 실행 순번을 알고싶어하는 프로세스의 인덱스인지 확인하고 맞다면
                        answer++; // 실행 순서 증가
                        return answer;
                    }
                    pq.poll(); // 인덱스 불일치한다면 pq에서 삭제(= 프로세스 실행)
                    answer++; // 실행 순서 증가
                }
            }
        }
        return answer;
    }
}
```

디버깅으로 변수의 변화 알아보기

priorities = [2, 1, 3, 2] ⇒ 프로세스 A,B,C,D

location = 3

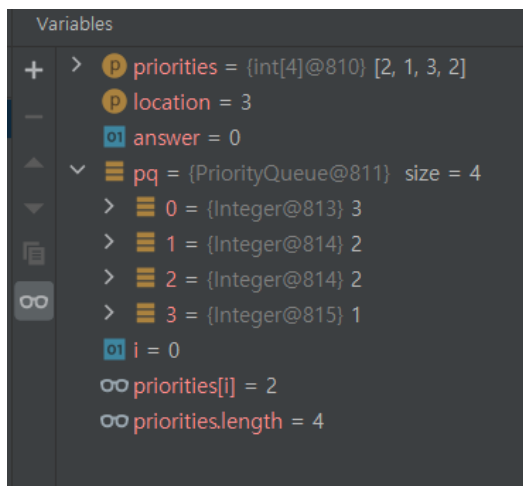
즉, priorities[location] == priorities[3] == 우선순위가 2인 프로세스D

`while(!pq.isEmpty())` pq가 빌 때까지 while문 안에서 for문이 실행

priorities를 기준으로 출력시도를 하므로 프로세스 A - B - C - D 순으로 출력시도를 한다.

1. `for (int i = 0; i < priorities.length; i++)` 총 4번 반복시도, i = 0 ~ 3

1) i = 0



```
Variables
+ > p priorities = {int[4]@810} [2, 1, 3, 2]
  p location = 3
  oi answer = 0
  v pq = {PriorityQueue@811} size = 4
    > 0 = {Integer@813} 3
    > 1 = {Integer@814} 2
    > 2 = {Integer@814} 2
    > 3 = {Integer@815} 1
  oi i = 0
  oo priorities[i] = 2
  oo priorities.length = 4
```

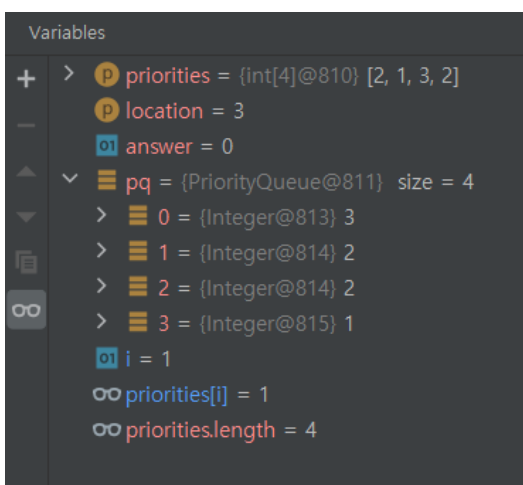
☀ priorities = [2, 1, 3, 2] ⇒ 프로세스 A,B,C,D
pq = [3, 2, 2, 1] ⇒ 프로세스 C, D, A, B

i = 0, location = 3 ⇒ priorities의 인덱스로 대입
하려는 값들
priorities[0] = 2 ⇒ 출력시도 프로세스 A(우선순
위: 2)
pq.peek() = 3

`if(priorities[i] == pq.peek())` priorities[0] = 2 ≠ pq.peek() = 3

즉, priorities에서 현재 출력하려는 프로세스가 A 인데 pq에 따르면 우선순위가 가장 큰 프로세스는 C 이므로 첫번째 반복문 종료

2) i = 1



```
Variables
+ > p priorities = {int[4]@810} [2, 1, 3, 2]
  p location = 3
  oi answer = 0
  v pq = {PriorityQueue@811} size = 4
    > 0 = {Integer@813} 3
    > 1 = {Integer@814} 2
    > 2 = {Integer@814} 2
    > 3 = {Integer@815} 1
  oi i = 1
  oo priorities[i] = 1
  oo priorities.length = 4
```

☀ priorities = [2, 1, 3, 2] ⇒ 프로세스 A,B,C,D
pq = [3, 2, 2, 1] ⇒ 프로세스 C, D, A, B

i = 1, location = 3
priorities[1] = 1 ⇒ 출력시도 프로세스 B(1)
pq.peek() = 3

`if(priorities[i] == pq.peek())` priorities[1] = 1 ≠ pq.peek() = 3

즉, priorities에서 현재 출력하려는 프로세스가 B 인데 pq에 따르면 우선순위가 가장 큰 프로세스는 C 이므로 두번째 반복문 종료

3) i = 2

```
Variables
+ > p priorities = {int[4]@810} [2, 1, 3, 2]
- p location = 3
  o1 answer = 0
  v pq = {PriorityQueue@811} size = 4
    > # 0 = {Integer@813} 3
    > # 1 = {Integer@814} 2
    > # 2 = {Integer@814} 2
    > # 3 = {Integer@815} 1
  o1 i = 2
  oo priorities[i] = 3
  oo priorities.length = 4
```

☀ priorities = [2, 1, 3, 2] ⇒ 프로세스 A,B,C,D
pq = [3, 2, 2, 1] ⇒ 프로세스 C, D, A, B

i = 2, location = 3
priorities[2] = 3 ⇒ 출력시도 프로세스 C(3)
pq.peek() = 3

`if(priorities[i] == pq.peek())` priorities[2] = 3 == pq.peek() = 3 일치!
⇒ `if (i == location)` i = 2 ≠ location = 3 출력 순서를 확인하려는 프로세스가 아님
⇒ `pq.poll();` 따라서 프로세스 실행하고 반복문 킵고잉
이때, pq = [2, 2, 1] ⇒ 프로세스 D, A, B 가 된다.

4) i = 3

```
Variables
+ > p priorities = {int[4]@810} [2, 1, 3, 2]
- p location = 3
  o1 answer = 1
  v pq = {PriorityQueue@811} size = 3
    > # 0 = {Integer@814} 2
    > # 1 = {Integer@815} 1
    > # 2 = {Integer@814} 2
  o1 i = 3
  oo priorities[i] = 2
  oo priorities.length = 4
```

☀ priorities = [2, 1, 3, 2] ⇒ 프로세스 A,B,C,D
pq = [2, 2, 1] ⇒ 프로세스 D, A, B

i = 3, location = 3
priorities[3] = 2 ⇒ 출력시도 프로세스 D(2)
pq.peek() = 2

`if(priorities[i] == pq.peek())` priorities[2] = 2 == pq.peek() = 2 일치!
⇒ `if (i == location)` i = 3 == location = 3 일치!
⇒ `return answer;` return문을 통해 solution함수 실행을 종료시킨다.

만약 location = 0 혹은 1 이었다면 for문이 종료된 후 while문을 통해 pq가 비어있지 않음을 확인하고 다시 for문이 실행되었을 것.