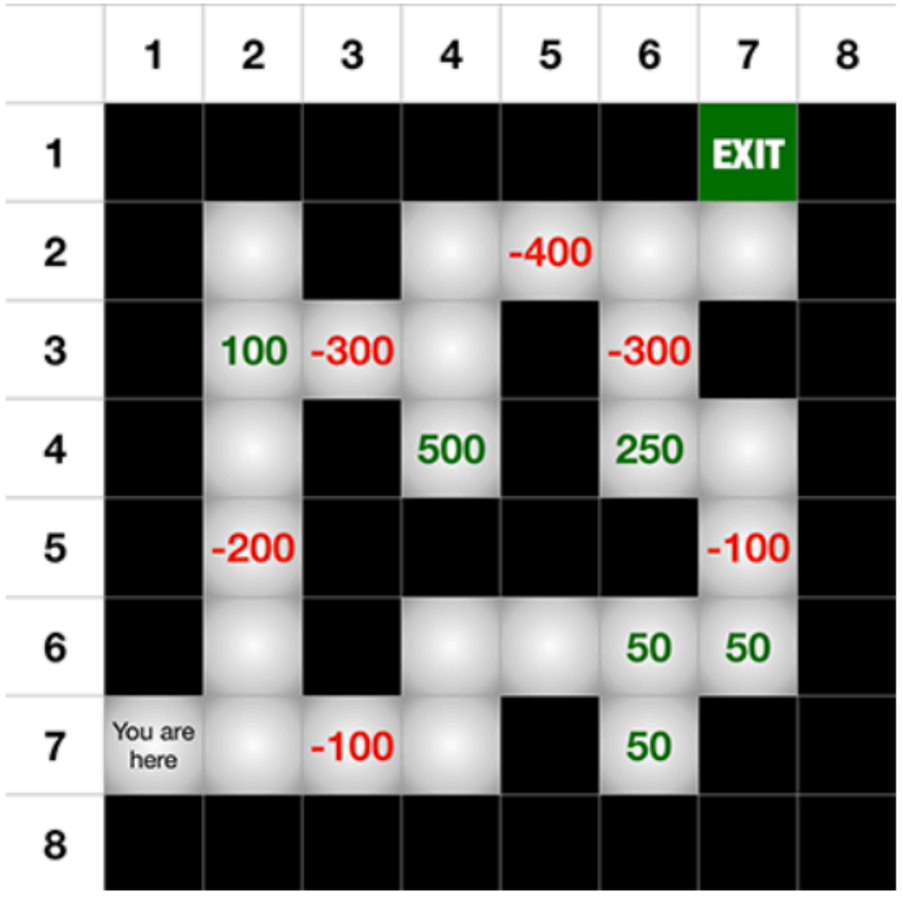
KICKSTART 2018 Round G

CaveEscape

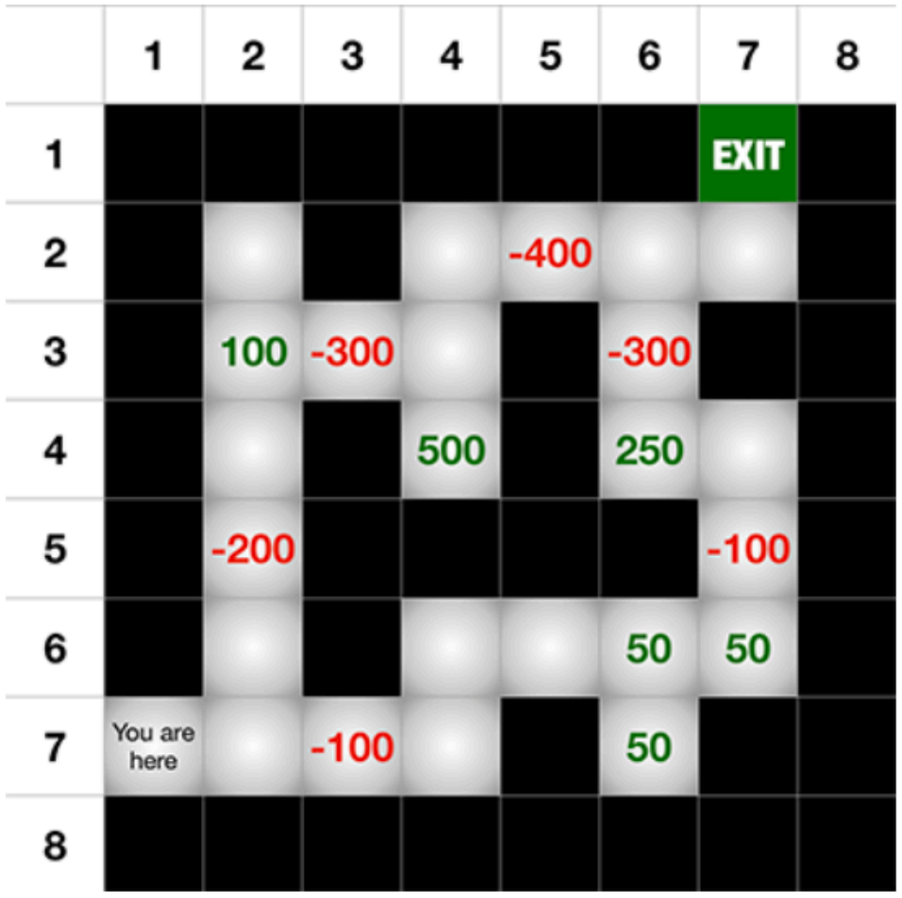
발표자 : 망고

Problem



NxM 의 동굴에서 탈출해야 한다.
시작점 (S_R, S_C)와 출구 (E_R, E_C)가 주어진다.
장애물인 칸은 지날 수 없으며, trap이 있는 칸을 처음으로 들어갈 때는 그 칸의 에너지만큼을 소비해야 한다.
또한 포션이 있는 칸에 처음 들어갈 때는 포션만큼의 에너지를 획득할 수 있다.
출구를 나갈 때 에너지를 최대한 많이 가지도록 움직일 때, 가능한 에너지의 값을 출력하여라.

Problem



제한사항

1<= N, M <= 100
0<= E <= 10^5
obstacle v_ij = 100000
trap -99999 <= v_ij <= -1
potion 1<= v_ij <= 99999
empty v_ij = 0
number of traps <= 15

Small Case
포션이 있는 칸이 없음

Large Case
포션이 있는 칸이 있음

Solution for small case

! Insight !

포션이 있는 칸이 없기 때문에, 출발할 이후 에너지가 증가할 수 있는 방법이 없다.
우리가 이동할 때 에너지는 그대로거나 감소하게 된다.
따라서 최단 경로 알고리즘을 이용해서 풀 수 있다.
(거리가 기준이 아니라 현재 가지고 있는 에너지 기준으로 heap을 사용한다

=>시간복잡도 O(NM*log(NM))

Solution for large case

포션이 있는 칸이 존재하기 때문에,
trap을 지나고 포션을 먹는 경우가 최종적으로 더 많은 에너지를 가지고 있을 수 있다.

이 경우 단순히 최단경로 알고리즘으로는 풀 수가 없다.

전탐색은 경우의 수가 너무 많다.

Solution for large case

1. **trap을 절대 지나지 않는다고 가정해보자.**
우리는 갈 수 있는 범위에서 최대한 많은 potion을 먹고, exit로 나가면(나갈 수 있다면) 된다.

Solution for large case

2. 우리가 지나갈 trap의 집합을 선택했고, 그 경우가 제한 사항에 위배되지않는 가능한 방법이라고 가정해보자.
우리는 갈 수 있는 범위에서 최대한 많은 potion을 먹을 것이다.
따라서 우리가 지나갈 칸들은 정해진다. (지나갈 트랩을 이미 선택했으므로 그 트랩의 차는 빼야하며, 갈 수 있는 potion 칸들은 모두 가야한다.)
마지막에 출구에서 가지고 있는 에너지는 **(우리가 갈 수 있는 범위에 있는 칸들의 potion의 에너지합 - trap의 에너지합)** 이며 **순서를 상관하지 않아도 된다.**

Solution for large case

이와 같은 가정을 토대로 문제를 풀어보자.

우리가 해야하는 것.

1. 지나갈 trap의 집합을 정하기
2. 지나갈 trap집합이 주어지면, cave를 탐색하여 exit에서 가지고 있을 에너지 값을 구하기

Solution for large case

이와 같은 가정을 토대로 문제를 풀어보자.

우리가 해야하는 것.

1. 지나갈 trap의 집합을 정하기
2. 지나갈 trap집합이 주어지면, cave를 탐색하여 exit에서 가지고 있을 에너지 값을 구하기

=> 순서는 상관없으므로 갈 수 있는 모든 칸을 다 방문하여 에너지를 합하거나 빼면 된다. 즉 BFS를 사용하면 된다.

Solution for large case

이와 같은 가정을 토대로 문제를 풀어보자.

우리가 해야하는 것.

1. 지나갈 trap의 집합을 정하기

우리가 지나갈 trap의 집합 T {trap_1, trap_2, … , trap_k}를 정했다고 가정하자. trap집합을 방문하는게 가능하려면, trap_i가 마지막에 방문하는 trap이라고 가정했을 때, trap_i를 제외한 집합 {trap_1, trap_2,…trap_i-1,trap_i+1, …, trap_k}에서 갈 수있는 구간과 인접하게 trap_i가 존재해야하고, 이 집합에서 구해지는 에너지합이 trap_i칸의 에너지보다 커야한다.

2. 지나갈 trap집합이 주어지면, cave를 탐색하여 exit에서 가지고 있을 에너지 값을 구하기

Solution for large case

이와 같은 가정을 토대로 문제를 풀어보자.

우리가 해야하는 것.

1. 지나갈 trap의 집합을 정하기

제한 사항에서 트랩의 개수가 15개 이하였으므로 우리는 bitmask를 이용해 집합을 표기할 수 있다!!!
즉 우리가 계산해야하는 집합의 개수는 2^(trap수) <= 2^15개이다.

2. 지나갈 trap집합이 주어지면, cave를 탐색하여 exit에서 가지고 있을 에너지 값을 구하기

Solution for large case

따라서 최종시간복잡도는

O(num_trap * NM) = O(2^15 * 100 * 100)이 된다.

Solution for large case

코드 보러가기