커넥티드 카

baekjoon 25395

202021018 오윤주

문제

커넥티드 카 실험 성공

★

골드 IV

시간 제한	메모리 제한	제출	정답	맞힌 사람	정답 비율
2 초 (추가 시간 없음)	1024 MB (추가 메모리 없음)	207	108	95	58.282%

문제

정보통신기술(ICT)의 발달에 힘입어, 미래형 자동차로 여겨졌던 인터넷 연결을 통해 운전자에게 다양한 서비스를 제공하는 커넥티드 카(connected car)가 현실로 다가왔다. 현대오토에버는 이에 발맞춰 클라우드와 사물 인터넷을 비롯한 최신 ICT를 적용한 차세대 커넥티드 카 서비스 플랫폼을 구축하고, 최고의 커넥티드 카를 완성하기 위한 핵심 소프트웨어 기술을 축적하고 있다.

현대오토에버의 엔지니어 현오는 새로운 서비스를 생각하던 중, 커넥티드 카의 핵심 기술인 사물 인터넷과 위치 기반 기술을 접목한 실험을 해 보기로 했다. 현오가 개발한 실험용 프로그램은 다음과 같은 기능을 가지고 있다.

- 현오는 사물 인터넷에 연결된 커넥티드 카를 원격 조종할 수 있다.
- 사물 인터넷에 연결된 커넥티드 카가 그렇지 않은 커넥티드 카와 같은 위치에 있게 되면, 그 커넥티드 카를 사물 인터넷에 연결시킬 수 있다. 이후 두 커넥티드 카가 다시 서로 멀어지더라도 연결은 그대로 유지된다.

현오는 실험을 위해 1부터 N까지 번호가 매겨진 N대의 커넥티드 카를 일렬로 배치했다. i번 커넥티드 카의 초기 위치는 x_i이고, 연료량은 h_i이다. 모든 커넥티드 카는 1만큼의 연료를 소비해서 1의 거리만큼 이동할 수 있으며, 연료를 모두 소비하면 더 이상 움직일 수 없다.

처음에 커넥티드 카들은 모두 사물 인터넷에 연결되지 않은 상태이다. 현오는 먼저 S번 커넥티드 카를 사물 인터넷에 연결시키고, 프로그램의 기능을 적절히 사용해 다른 커넥티드 카들에 사물 인터넷을 전파하려고 한다.

현오가 커넥티드 카들을 어떻게 다루느냐에 따라 실험에서 사물 인터넷에 연결되는 커넥티드 카들의 조합은 달라질 수 있다. 현오가 다양한 방법으로 여러 번 실험을 진행할 때, 사물 인터넷에 연결될 가능성이 있는 커넥티드 카를 전부 구해 보자.

문제

예제 입력 1 복사

```
5 3
1 2 4 5 8
2 1 2 2 3
```

1. 3번 차가 2번 차로 이동 -> {2, 3}
2. 3번 차가 2번 차로 이동, 2번 차가 1번 차로 이동 -> {1, 2, 3}
3. 3번 차가 4번 차로 이동 -> {3, 4}
4. 3번 차 혼자 -> {3}

= 1, 2, 3, 4번 차 연결 가능

문제

```python
import sys

input  = sys.stdin.readline
N, S = map(int, input().split())

car = list(map(int, input().split()))
fuel = list(map(int, input().split()))

S -= 1
ans = [S + 1]

left = car[S] - fuel[S]
right = car[S] + fuel[S]
first_llo = left
first_rlo = right

left_lo = S
right_lo = S
```

// 차, 움직일 수 있는 거리 연료 입력

// S는 위치

// 연결되어 있는 차의 왼쪽, 오른쪽 이동

// 연결된 차의 처음 위치

// 맨 처음 연결되어있는 차 위치

문제

```
while True:
    for i in range(right_lo + 1, N):
        if car[i] <= right:
            right = max(right, car[i] + fuel[i])
            left = min(left, car[i] - fuel[i])
            ans.append(i+1)
        else:
            right_lo = i - 1
            break
    else:
        right_lo = N
```

// 맨 처음 연결된 차 기준 오른쪽으로

// 현재 위치한 왼쪽, 오른쪽으로 갈 수 있는 거리와 맨 처음 연결된 차의 거리 비교

// 다 하면 왼쪽도 해야하기때문에 맨 처음 차 위치 바꾸기

// 못간다면 마지막 차가 연결되어있는 차

문제

```
    for i in range(left_lo - 1, -1, -1):
        if car[i] >= left:
            left = min(left, car[i] - fuel[i])
            right = max(right, car[i] + fuel[i])
            ans.append(i+1)
        else:
            left_lo = i+1
            break
    else:
        left_lo = 0

    if left == first_llo and right == first_rlo:
        break

    first_llo = left
    first_rlo = right

ans.sort()

print(*ans)
```

// 왼쪽도 같은 방식

// 같으면 연결되었다는 뜻

// 연결된 차 위치 업데이트

감사합니다.