C7. way back home

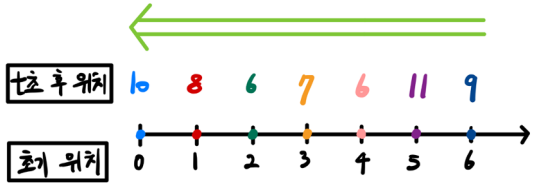
이원빈, 박민지

Problem Analyze

t초 동안의 모든 위치를 계산하여 그룹화를 시키는 것은 최악의 경우 n = 10^5에 대해 (1 ≤ n ≤ 100,000) t = 10^9 (1 ≤ t ≤1,000,000,000)만큼 비교를 하기 때문에 비효율적이고, 계산 시간이 오래걸릴 수 있다. 따라서 t초 후의 각 사람의 최종 위치를 기반으로 솔루션을 전개한다.

처음 위치 기준으로 t초 후 최종 위치로 이동했을 때, 특정 사람 a가 b를 따라잡았는지 따라잡지 않았는지 확인할 수 있다면 그룹화를 시킬 수 있다.
여기서 '따라잡다'의 정의는, 주어진 입력 (position, speed)에 대해 -> (p1, s1), (p2, s2) p1<p2 인 경우 p1 + t * s1 ≥ p2 + t * s2 를 만족할 때 따라잡았다고 할 수 있다.

더불어 사람들이 앞 사람을 추월하면 그룹을 이룰 수 밖에 없으므로,
초기에 가장 멀리있던 사람부터 확인하며 누가 앞 사람을 따라잡을 수 있는지 확인하는 것이 합리적이다.
이는 곧 앞 사람을 따라잡으며 생겨나는 그룹을 그 뒤에 누가 또 따라잡을 수 있는지 확인하는 것으로 확장할 수 있다.



이를 위해 처음 위치 기준으로 사람들이 내림차순으로 정렬이 되어있어야하며, 그 상태에서 각자의 t초 후 위치를 구한 후,
첫 번째 사람을 한 그룹으로 두고, 그 그룹이 그 뒷 사람에게 따라 잡힐 수 있는지 확인해야한다. 따라잡히지 않는다면 새로운 그룹이 [illegible]

Solution.

1. 사람의 수 n과 시간 t를 입력 받는다. 더불어 각 사람에 대한 position과 Speed를 입력받아 1차원 배열 people에 저장하고, position을 기준으로 내림차순 정렬한다.
2. people 배열을 이용하여, 각 사람에 대해 t초 후에 위치한 position을 (초기 position + 그 사람의 speed * t)로 계산하고, 1차원 배열 afterTPos에 저장한다.
3. n ≥ 1이므로 적어도 한 그룹은 생성된다. 따라서 특정 그룹을 구분짓는 대푯값을 나타내는 groupPos을 우선 afterTPos[0]로 하고, 그룹의 수를 나타내는 groupNum을 1로 초기화한다.
4. afterTPos[i] (1 ≤ i < n)에 저장된 afterTPos배열의 두 번째 사람의 위치부터 마지막 사람의 위치까지 다음의 과정을 반복한다.
    a. 현재 groupPos가 확인하고자 하는 사람의 위치보다 큰 지 확인한다. (groupPos > afterTPos[i])
        i. groupPos > afterTPos[i] 라면 , 대표값이 저장된 그룹을 따라잡지 못한 것이고, 새로운 그룹을 필요로 하므로 groupNum을 하나 늘린다.
        더불어 groupPos도 afterTPos[i]로 업데이트한다.
        ii. groupPos <= afterTPos[i] 라면, 해당 그룹을 따라잡은 것이므로 같은 그룹이라 판단하여 다음 사람으로 넘어간다.
5. 4번의 과정으로 구해진 groupNum을 출력한다.

Time complexity

(1) n명의 사람에 대해 position을 기준으로한 내림차순 정렬 ⇒ O(nlogn)
(2) n명의 사람에 대해 t초 후의 각 position 구하기 ⇒ O(n)
(3) (2)로부터 구해진 n명의 사람의 t초후 position을 기반으로, 현재 그룹의 위치와 현재 조회하는 사람의 위치 비교 ⇒ O(n-1) = O(n)
(4) 따라서 O(nlogn)