"중등부 3번. 건초 더미" 문제 풀이

작성자: 오주원

부분문제 1

각 건초 더미를 설치할지 말지 결정하는 2^N가지 배치를 모두 시도해 보고, 그중 화살을 멈출 수 있는 배치들 가운데 설치한 건초 더미의 수가 가장 적은 것을 답으로 출력하면 된다.

부분문제 2

설치할 수 있는 건초 더미들 가운데 방어력이 큰 건초 더미부터 설치하는 것이 이득이다. 따라서 위치 X 이하의 건초 더미를 방어력 순으로 정렬하고, 제일 큰 방어력부터 합을 누적하다가 처음으로 합이 P 이상이 되는 순간까지 설치한 더미 개수를 반환하면 그것이 최소 개수이다. 쿼리마다 최대 N개의 원소를 정렬해야 하므로, 시간복잡도는 $O(QN \log N)$이다.

부분문제 3

방어력 별 누적합을 미리 구해두면, 위치 X 이하에 방어력이 D인 건초 더미의 개수를 $O(1)$에 얻을 수 있다. 쿼리마다 방어력이 큰 값부터 내려가면서 방어력의 합이 P 미만이도록 추가로 설치할 수 있는 건초 더미의 최대 개수를 수식을 통해 구한다. 그 값이 해당 실제 개수보다 크거나 같다면 전부 설치하고, 작으면 그 값에 1을 더한 개수만큼 설치한 방법이 바로 답이 된다. 이렇게 하면 쿼리당 방어력의 범위만큼 시간이 들어가고, 누적합을 전처리하는 과정을 추가하여 $O(ND+QD)$의 시간복잡도로 부분문제를 해결할 수 있다.

누적합을 사용하지 않고 해당 부분문제를 풀 수 있다. 쿼리를 X 오름차순으로 정렬해 두고 답을 구한 뒤, 입력받은 순서대로 답을 출력하는 방법으로 문제를 해결할 수 있다. 이렇게 하면 누적합을 사용하지 않고 방어력 별 개수를 저장하는 카운팅 배열을 관리하는 것만으로 앞선 풀이대로 풀 수 있다. 이 때, 위치를 카운팅 소트식으로 정렬하면 로그를 붙이지 않고 $O(N+QD)$에 풀 수 있다.

부분문제 4

건초 더미의 방어력이 오름차순으로 주어지므로, 위치 X이하에서 최소 개수로 힘 P를 넘기려면 제일 오른쪽 끝인 위치 X에 설치할 건초 더미부터 차례로 선택해 합이 P 이상이 되는 지점까지 설치하면 된다.

우선 방어력의 합이 P 미만인 경우 어떠한 배치로도 화살을 못 막으므로 -1을 출력하고, 그렇지 않다면 맨 오른쪽부터 K개의 합이 P를 넘기는 최소 K를 구해서 $K+1$을 출력하면 된다. K는 누적합과 이분 탐색을 이용해 $O(\log N)$에 구할 수 있으므로, $O(N+Q\log N)$에 부분문제를 해결할 수 있다.

부분문제 5

초등부 3번. 허수아비와 동일한 부분문제이다. P가 동일하므로 한 번 설치하지 않기로 결정한 건초 더미는 다시 설치하지 않게 되고, 이를 이용해 우선순위 큐로 최적화가 가능하다. 자세한 풀이는 **초등부 3번. 허수아비**의 해설을 참고하라.

부분문제 6

위치에 상관없이 모든 건초 더미를 사용할 수 있으므로 D를 정렬한 뒤 답을 구할 수 있고, 정렬한 뒤에는 부분문제 4와 동일한 문제가 된다.

부분문제 8

부분문제 3의 두 번째 풀이처럼 모든 쿼리를 X 오름차순으로 정렬해 한번에 처리하면 X가 증가하는 순서대로 답을 계산할 수 있다. 카운팅 배열 대신 방어력 합을 저장하는 세그먼트 트리를 사용하면 **설치할 수 있는 건초 더미 중 방어력이 K이상인 건초 더미의 방어력 합**을 $O(\log N)$에 구해줄 수 있다. 방어력의 값 자체는 10^9 수준으로 매우 크지만, 서로 다른 값은 최대 N개이므로 좌표 압축 후 세그먼트 트리를 만들면 메모리 부담 없이 관리할 수 있다.

앞서 만든 세그먼트 트리를 이용하면, 각 쿼리마다 **방어력 합이 P 이상이 되도록 선택해야 하는 건초 더미의 최소 개수**를 빠르게 구할 수 있다. 구체적으로는 **설치할 수 있는 건초 더미 중 방어력이 K이상인 건초 더미의 방어력 합**이 P를 넘지 않는 가장 작은 K를 찾은 뒤, 방어력이 $K-1$인 건초 더미를 일부 추가해 문제의 답을 만족하는 완성하는 것이고, 이는 쿼리 하나당 $O(\log^2 N)$의 시간이 소요되어 전체 문제를 $O(N \log N + Q \log^2 N)$에 해결할 수 있다.

더 나아가 세그먼트 트리에 (건초더미의 개수, 방어력 합)을 저장하면, 오른쪽 자식부터 내려가면서 누적 합이 P에 처음 도달하는 지점을 찾아 최소 개수를 바로 계산할 수 있다. 이 테크닉을 사용하면 이분 탐색을 사용하지 않아 $O(\log N)$으로 최적화되며, 문제의 특성상 Suffix의 값을 구하므로 상수가 비교적 작은 Binary Indexed Tree(Fenwick Tree)에서도 사용해 추가적으로 시간을 단축할 수 있다. BIT에서의 구현 방법은 같이 첨부된 예제 코드를 참고하라. 이 풀이대로 전체 문제를 $O(N \log N + Q \log N)$에 해결할 수도 있다.