BP 4/25 연습문제 해설

sorohue@sasa.hs.kr

광고

2023년도 한국정보올림피아드 신청 마감까지 1주일 남았습니다!!

많은 신청 부탁드립니다.

~~2023년 4월 30일 오후 12:31 구데기컵 많관부~~

문제		난이도
A	OX퀴즈	Bronze II
B	먹을 것인가 먹힐 것인가	Silver III
C	잃어버린 괄호	Silver II
D	조삼모사	Gold III

BOJ 8958
OX퀴즈

#implementation #string

난이도 – Bronze II

BOJ 8958 OX퀴즈

- O가 연속되어 나올 수록 얻는 점수가 증가합니다.
- 이번 문제에서 얻은 점수를 p라고 하면,

  O가 나왔을 때 p는 이전보다 1 증가하고 X가 나왔을 때 p는 0이 됩니다.
- 이를 바탕으로 문자열의 첫 글자부터 훑으면서 p를 갱신하고,

  그 합을 구하면 정답을 구할 수 있습니다.
- 이때 시간 복잡도는 테스트 케이스 당 O(N) 입니다.

BOJ 7795
먹을 것인가 먹힐 것인가

#sorting

난이도 – Silver III

BOJ 7795 먹을 것인가 먹힐 것인가

- N, M의 크기가 각각 최대 20000이므로 가능한 쌍은 최대 4억 개입니다.
  단순 이중 for문으로 구현하기에는 시간이 너무 오래 걸립니다.
- 보다 빠르게 가능한 모든 경우를 탐색하는 방법이 있을까요?

- 세 수 a, b, c에 대해 a가 b보다 크고 c가 a보다 크다면
  c도 b보다 큽니다.

BOJ 7795 먹을 것인가 먹힐 것인가

- A와 B가 각각 크기 순으로 정렬되어 있다고 합시다.

  그러면 A를 크기 순서대로 B와 비교할 때, A에 속하는 생명체가

  B에 속하는 생명체보다 크다면 그 뒤에 비교할 A의 생명체들은

  모두 해당 B의 생명체보다 크기가 큽니다.
- 마찬가지로 B에 속하는 생명체가 A에 속하는 생명체보다 크다면

  그 뒤에 비교할 B의 생명체들은 모두 해당 A의 생명체보다 크기가 큽니다.

BOJ 7795 먹을 것인가 먹힐 것인가

- 따라서 비교했을 때 더 작은 쪽만 다음 생명체로 변경해 비교를 해주면서
  A의 생명체가 B의 생명체보다 작거나 같은 경우가 되었을 때마다
  현재까지 비교해주었던 B의 생명체의 수의 합을 구하면
  A의 크기가 B보다 큰 쌍의 개수를 구할 수 있습니다.
- 이 상황에서 시간 복잡도는 테스트 케이스 당 O(N+M) 입니다.

BOJ 7795 먹을 것인가 먹힐 것인가

- 하지만 문제에서 주어진 A와 B는 정렬되어 있지 않습니다.
- 때문에 A와 B를 각각 정렬해 주어야 합니다. 이때 std::sort 등을 이용하면
  O(N logN)의 시간 복잡도에 A를,
  O(M logM)의 시간 복잡도에 B를 정렬할 수 있습니다.

- 따라서 최종 시간 복잡도는 테스트 케이스당 O(N logN + M logM) 입니다.

BOJ 1541
잃어버린 괄호

#greedy #string

난이도 – Silver II

BOJ 1541 잃어버린 괄호

- 식의 값을 최소화하기 위해서는 빼는 값의 크기를 최대화해야 합니다.
- 따라서 최대한 많은 숫자를 더한 뒤 뺄셈을 하는 것이 유리합니다.
- 뺄셈을 기준으로 나머지 덧셈 연산을 실행할 때, 덧셈의 결합 법칙에 의해 어떻게 괄호로 묶어도 계산 결과가 동일합니다.
- 따라서 모든 덧셈 계산을 한 후 뺄셈을 계산하면 최솟값을 얻을 수 있습니다.

BOJ 1621
조삼모사

#dp #backtracking

난이도 – Gold III

BOJ 1621 조삼모사

- 바나나를 가져갈 수 있는 모든 경우의 수를 다 시도해 볼 수는 없습니다.
- 바나나를 한 개씩 집어가거나 연속한 바나나들을 들고 가야 하므로 첫 번째부터 순서대로 바나나를 집어가도 문제에 차이가 없습니다.
- 이전 바나나까지의 바나나들을 모두 집어가는 데에 필요한 시간을 이용해 첫 바나나부터 현재 바나나까지의 바나나들을 모두 집어가는 데 필요한 시간을 계산할 수 없을까요?

BOJ 1621 조삼모사

- D[i]를 1번째 바나나부터 i번째 바나나까지를 집어가는 데 필요한 시간의 최솟값으로 둡시다.
- i번째 바나나를 가져가는 경우는

  (1) (i-1)번째 바나나까지 가져간 후 i번째 바나나 하나를 가져가는 경우와

  (2) (i-K)번째 바나나까지 가져간 후 (i-K+1) ~ i번째까지 K개의 바나나를 가져가는 경우만이 존재합니다.

BOJ 1621 조삼모사

- 이때 D[i]의 값이 최소화되어야 하므로, 이를 식으로 정리하면
  D[i] = min(D[i−1]+BANANA[i], D[i−K] + C) 입니다.
- 이때 i가 K보다 작으면 K개의 바나나를 집어가는 것이 불가능하므로
  D[i] = D[i−1] + BANANA[i] 입니다.
- 1부터 N까지 순서대로 D[i]의 값을 구하면서, 하나만 집어가는 것과
  K개 집어가는 것 중 어느 쪽이 유리한 지를 저장해 둡시다.
  (두 경우의 시간이 같은 경우에는 하나만 집어갑니다.)

BOJ 1621 조삼모사

- K개의 바나나를 가져가야 하는 위치들을 찾아 봅시다.
- N번째부터 거꾸로 훑어보면서, i번째 바나나까지 가져갔을 때
  마지막에 K개의 바나나를 동시에 가져가는 경우가 최소라면
  그 맨 왼쪽 위치를 저장해 두고 K개 왼쪽의 바나나를 확인,
  아니면 1개 왼쪽의 바나나를 확인합니다.
- 마지막으로 저장해 두었던 위치들을 역순으로 출력해주면 됩니다.
- 이때 시간 복잡도는 O(N) 입니다.