week 6

2024-08-08

5896: 효율적으로 소 사기

n 마리의 소 , k 개의 쿠폰 , 가지고있는 돈 m

소의 가격은
그냥 가격 , 쿠폰을 적용한 가격이 있음
(그냥가격 >= 쿠폰적용가격)

쿠폰은 다 안 써도 됨

소를 최대 몇 마리 살 수 있는지 ??

input)
4 1 7
3 2
2 2
8 1
4 3

5896: 효율적으로 소 사기

그리디로 생각하기

cin >> t1 >> t2;

a.push_back({t1, 0, i});

a.push_back({t2, 1, i});

배열에 전부 넣고 정렬한 뒤 싼거부터 사기

쿠폰가격이고 , 쿠폰을 보유중이라면 사용하고 없다면 넘어간다 .

튜플 { 가격 , 쿠폰 적용한 가격인지 , 인덱스 (같은걸 두번 사면 안되니깐)}

정답같아보임

5896: 효율적으로 소 사기

2 1 6
10 4
2 1

가장 싼 쪽에 쿠폰을 써버리면 비싼 쪽에서 큰 할인을 놓칠 수 있다 .

5896: 효율적으로 소 사기

쿠폰을 최대한 다 쓰려고 하되 ,

다른 소에게 쿠폰을 쓰는게 전체적으로 더 큰 이득일 경우

쿠폰을 사용하는 소를 바꿔주어야 함 .

5896: 효율적으로 소 사기

소를 살 때 3 가지 선택지

1. 쿠폰이 남아있다면 쿠폰을 적용해서 사기 (베스트)

2. 기본 가격으로 사기

3. 앞에서 쿠폰 적용해서 산 소를 기본 가격으로 바꾸고 이번 차례에 쿠폰을 사용하기

5896: 효율적으로 소 사기

ex)
2 1 6
10 4
2 1

2 번 소가 제일 싸기 때문에 쿠폰을 적용해서 산다 .
쿠폰은 더이상 남아있지 않다 . 1 번 선택지는 불가능 .

2. 기본 가격으로 사기
   -> +10 원

5896: 효율적으로 소 사기

ex)
2 1 6
10 4
2 1

3. 앞에서 쿠폰 적용해서 산 소를 기본 가격으로 바꾸고 이번 차례에 쿠폰을 사용해서 사기

1 원이 2 원으로 바뀌고 4 원을 사용했으므로

-> +1, +4 -> +5 원

5896: 효율적으로 소 사기

매 선택지마다

이전에 쿠폰을 사용한 것중에 쿠폰가 , 기본가의 차이가 가장 작은 것 + 현재 쿠폰 가격

현재 기본 가격보다 작다면

이전에 쿠폰을 쓴 걸 기본가격으로 바꿔주고 이번에 쿠폰을 적용하여 사야한다 .

5896: 효율적으로 소 사기

~~sort(a)

```
for(int i = 0; i < k; i++) {
    if(cost + a[i].second <= m) {
        cost += a[i].second;
        ans++;
        pq.push(a[i].first - a[i].second);
    }
}
```

쿠폰가격으로 정렬하고 앞에서부터 구매하면서 ,

쿠폰가격과 기본 가격의 차이를 MINHEAP 에 넣어준다 .

5896: 효율적으로 소 사기

```
for(int i = k; i < n; i++) {
    if(!pq.empty() && pq.top() + a[i].second < a[i].first) {
        if(cost + pq.top() + a[i].second <= m) {
            cost += pq.top() + a[i].second;
            pq.pop();
            pq.push(a[i].first - a[i].second);
            ans++;
        }
    } else {
        if(cost + a[i].first <= m) {
            cost += a[i].first;
            ans++;
        }
    }
}

cout << ans << '\n';
```