Fractional Knapsack

Problem

We are given n items with {weight, value} of each item and the capacity of knapsack (bori) W. We need to put these items in the knapsack such that the final value of items in the knapsack is maximum.

Example

Value	21	24	12	40	30
Weight	7	4	6	5	6

W = 20

Approach (Think greedily)

1. Calculate value per unit weight.

Value	21	24	12	40	30
Weight	7	4	6	5	6
Value/ weight	3	6	2	8	5

2. Sort in decreasing order according to value/weight.

Value	40	24	30	21	12
Weight	5	4	6	7	6
Value/ weight	8	6	5	3	2

3. Pick from the starting till our knapsack has capacity.

So maximum value = 40 + 24 + 30 + 3(5) = 109.

Code

```
#include<bits/stdc++.h>
using namespace std;
#define int long long

struct item {
    double value, weight, valuePerWeight;
};

bool compare(item i1, item i2) {
    return i1.valuePerWeight > i2.valuePerWeight;
}

signed main() {
    int n; cin >> n;

    vector<item> items;
    for(int i=0; i<n; i++) {
        double v,w;
        cin >> v >> w;

        items.push_back({v,w,v/w});
    }

    double W; cin >> W;
```

```
    sort(items.begin(), items.end(), compare);

    int ans = 0;
    for(int i=0; i<n; i++) {
        if(W >= items[i].weight) {
            W -= items[i].weight;
            ans += items[i].value;
        }
        else {
            ans += W * items[i].valuePerWeight;
            W = 0;
            break;
        }
    }
    cout << ans << endl;
    return 0;
}
```