1. **Write a program to solve a fractional Knapsack problem using a greedy method. (DAA)**

#include <iostream>

#include <algorithm>

using namespace std;

class Item {

public:

float weight;

float value;

Item(float w = 0.0, float v = 0.0) {

weight = w;

value = v;

}

};

bool compare(Item a, Item b) {

return (a.value / a.weight) > (b.value / b.weight);

}

float fractionalKnapsack(Item items[], int num\_items, float capacity) {

sort(items, items + num\_items, compare);

float total\_value = 0.0;

for (int i = 0; i < num\_items; i++) {

if (capacity <= 0) {

break;

}

if (items[i].weight <= capacity) {

total\_value += items[i].value;

capacity -= items[i].weight;

} else {

total\_value += (capacity \* (items[i].value / items[i].weight));

capacity = 0;

}

}

return total\_value;

}

int main() {

int num\_items;

cout << "Enter the number of items: ";

cin >> num\_items;

Item items[num\_items];

cout << "Enter the weights and values of the items:" << endl;

for (int i = 0; i < num\_items; i++) {

float weight, value;

cout << "Item " << i + 1 << " - Weight: ";

cin >> weight;

cout << "Item " << i + 1 << " - Value: ";

cin >> value;

items[i] = Item(weight, value);

}

float capacity;

cout << "Enter the capacity of the knapsack: ";

cin >> capacity;

float max\_value = fractionalKnapsack(items, num\_items, capacity);

cout << "Maximum value in the knapsack: " << max\_value << endl;

return 0;

}