**DAA Assignment 3**

**Fractional Knapsack Using Greedy Algorithm**

**Neeti Kurulkar**

**#include <iostream>**

**#include <vector>**

**#include <algorithm>**

**using namespace std;**

**// Structure to represent an item with weight and value**

**struct Item {**

**int weight;**

**int value;**

**};**

**// Comparator function to sort items by value-to-weight ratio in descending order**

**bool compareByRatio(const Item &a, const Item &b) {**

**double ratioA = static\_cast<double>(a.value) / a.weight;**

**double ratioB = static\_cast<double>(b.value) / b.weight;**

**return ratioA > ratioB;**

**}**

**// Function to calculate the maximum value for the fractional knapsack problem**

**double fractionalKnapsack(int capacity, vector<Item> &items) {**

**// Sort items by value-to-weight ratio**

**sort(items.begin(), items.end(), compareByRatio);**

**double totalValue = 0.0;**

**for (const auto &item : items) {**

**if (capacity >= item.weight) {**

**// If the knapsack can hold the entire item, take it completely**

**capacity -= item.weight;**

**totalValue += item.value;**

**} else {**

**// If the knapsack can't hold the full item, take the fraction**

**totalValue += item.value \* (static\_cast<double>(capacity) / item.weight);**

**break; // Knapsack is full**

**}**

**}**

**return totalValue;**

**}**

**int main() {**

**int capacity, n;**

**// Input knapsack capacity**

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

**cin >> capacity;**

**// Input number of items**

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

**cin >> n;**

**vector<Item> items(n);**

**// Input weight and value for each item**

**for (int i = 0; i < n; i++) {**

**cout << "Enter weight and value for item " << i + 1 << ": ";**

**cin >> items[i].weight >> items[i].value;**

**}**

**// Calculate maximum value that can be put in the knapsack**

**double maxValue = fractionalKnapsack(capacity, items);**

**cout << "Maximum value in knapsack = " << maxValue << endl;**

**return 0;**

**}**

**Output:**

**A screenshot of a computer

AI-generated content may be incorrect.**