NAME:	Prithvi Singh
UID:	2022301014
SUBJECT	DAA
EXPERIMENT NO:	05
AIM:	Knapsack problem

Algorithm:	Algorithm GREEDY_FRACTIONAL_KNAPSACK(X, V, W, M)
	// Description : Solve the knapsack problem using greedy approach
	
	// Input:
	X: An array of n items
	V: An array of profit associated with each item
	W: An array of weight associated with each item
	M: Capacity of knapsack
	
	// Output :
	SW: Weight of selected items
	SP: Profit of selected items
	// Items are sorted in decreasing order of pi = vi / wi ratio
	
	S ← Φ // Set of selected items, initially empty
	SW ← 0 // weight of selected items
	SP ← 0 // profit of selected items
	i ← 1
	
	while i ≤ n do
	  if (SW + w[i]) ≤ M then
	  S ← S ∪ X[i]
	  SW ← SW + W[i]
	  SP ← SP + V[i]
	  else
	  frac ← (M - SW) / W[i]
	  S ← S ∪ X[i] * frac // Add fraction of item X[i]

	SP ← SP + V[i] * frac // Add fraction of profit SW ← SW + W[i] * frac // Add fraction of weight end i ← i + 1 end
Code:	

```
#include <bits/stdc++.h>
#include <iostream>
#include <algorithm>
#include <string>
#include <cmath>
using namespace std;

void printarr(double **arr,int n) {
    cout << "item \t\t\t weight \t\t\t Value \t\t\t Value/weight\n";
    for (int i = 0; i < n; i++) {

        for (int j = 0; j < 4; j++) {

            cout << arr[i][j] << "\t\t\t";
        }
        cout << "\n";

    }
}


int main()
{
    int c, n;
    double profit = 0.0,weight = 0.0;

    cout << "Enter the weight of the sack: ";
    cin >> c;
    cout << "Enter the no of items: ";
    cin >> n;
    cout << "Enter weight and value of each item: \n";

    vector<string> s (n);

    double **arr = new double*[n];
    for (int i = 0; i < n; i++) {
        arr[i] = new double[4];
        arr[i][0]=i+1;
```

```
        for (int j = 1; j < 4; j++) {

            if (j == 3)
            {
                arr[i][j] = arr[i][2] / arr[i][1];
            }

            else {
            cout << "Enter weight and value for [" << i << "][" << j
<< "]: ";
            cin >> arr[i][j];
            }
        }
    }

    printarr(arr,n);
    cout << "Sorted based on ratio: " << endl;

    sort(arr, arr + n, [](const double* a, const double* b) {
        return a[3] > b[3];
    });

    printarr(arr,n);

    int remain = 0;
    double remain_pro = 0.0;
    string coco = "";
    ostringstream ss;

    for (int i = 0; i < n; i++) {
        if (c >= weight + arr[i][1]){
            weight += arr[i][1];
            s[i] = to_string(lround(arr[i][0]));
            profit += arr[i][2];
        }
        else
        {

            remain = c - weight;
            weight += remain;
            remain_pro = (remain * arr[i][2]) / arr[i][1];
            profit += remain_pro;
            ss << remain << "/" << arr[i][1];
            coco = ss.str();
```

```
            s[i] = to_string(lround(arr[i][0])) + " (" + coco + ")";
            break;
        }

    }

    cout << "Total weight: " << weight << endl;
    cout << "Total profit: " << profit << endl;
    cout << "All items in the bag: {";
    for (int i = 0; i < s.size(); i++)
    {
            cout << s[i] << ",";
    }
    cout << "}";


    return 0;
}
```

Output:

- **Knapsack Problem:**

```
PS C:\Users\prith\OneDrive\Desktop\Semester 4\DAA Practicals\Exp5> g++ knapsack.cpp
PS C:\Users\prith\OneDrive\Desktop\Semester 4\DAA Practicals\Exp5> ./a.exe
Enter the weight of the sack: 70
Enter the no of items: 4
Enter weight and value of each item:
Enter weight and value for [0][1]: 20
Enter weight and value for [0][2]: 60
Enter weight and value for [1][1]: 10
Enter weight and value for [1][2]: 40
Enter weight and value for [2][1]: 30
Enter weight and value for [2][2]: 100
Enter weight and value for [3][1]: 25
Enter weight and value for [3][2]: 56
item                    weight                          Value                   Value/weight
1                       20                      60                      3
2                       10                      40                      4
3                       30                      100                     3.33333
4                       25                      56                      2.24
Sorted based on ratio:
item                    weight                          Value                   Value/weight
2                       10                      40                      4
3                       30                      100                     3.33333
1                       20                      60                      3
4                       25                      56                      2.24
Total weight: 70
Total profit: 222.4
All items in the bag: {2,3,1,4 (10/25),}
PS C:\Users\prith\OneDrive\Desktop\Semester 4\DAA Practicals\Exp5>
```

Conclusion:	Thus we have performed Fractional Knapsack Problem using Greedy Approach. Greedy algorithms are used to find an optimal or near-optimal solution to many real-life problems.