NAME:	Shubham Solanki
UID:	2022301015
SUBJECT	Design and Analysis of Algorithms
EXPERIMENT NO:	05
AIM:	To implement 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 Value \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 << "\nEnter the weight of the sack: ";
```

```
    cin >> c;
    cout << "\nEnter the no of items: ";
    cin >> n;
    cout << "\nEnter weight and value of each item:
\n\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];
            }
        }
    }
    cout<<endl;
    printarr(arr,n);
    cout << "\nSorted based on ratio: \n" << 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 << "\nTotal weight: " << weight << endl;
    cout << "\nTotal profit: " << profit << endl;
    cout << "\nAll items in the bag: {";
    for (int i = 0; i < s.size(); i++)
    {
```

```
            cout << s[i] << ",";
    }
    cout << "}";
    cout<<endl;
    cout<<endl;

    return 0;
}
```

Output:

Knapsack problem

```
shubham@shubham-virtual-machine:~/semester4/daa/experiments/exp 5$ g++ knapsack.cpp
shubham@shubham-virtual-machine:~/semester4/daa/experiments/exp 5$ ./a.out

Enter the weight of the sack: 28

Enter the no of items: 7

Enter weight and value of each item:

Enter weight and value for [0][1]: 2
Enter weight and value for [0][2]: 9
Enter weight and value for [1][1]: 5
Enter weight and value for [1][2]: 5
Enter weight and value for [2][1]: 6
Enter weight and value for [2][2]: 2
Enter weight and value for [3][1]: 11
Enter weight and value for [3][2]: 7
Enter weight and value for [4][1]: 1
Enter weight and value for [4][2]: 6
Enter weight and value for [5][1]: 9
Enter weight and value for [5][2]: 16
Enter weight and value for [6][1]: 1
Enter weight and value for [6][2]: 3

Item                    Weight                  Value           Value/Weight
1                       2                       9                       4.5
2                       5                       5                       1
3                       6                       2                       0.333333
4                       11                      7                       0.636364
5                       1                       6                       6
6                       9                       16                      1.77778
7                       1                       3                       3
```

```
Sorted based on ratio:

Item                    Weight                  Value           Value/Weight
5                       1                       6                       6
1                       2                       9                       4.5
7                       1                       3                       3
6                       9                       16                      1.77778
2                       5                       5                       1
4                       11                      7                       0.636364
3                       6                       2                       0.333333

Total weight: 28

Total profit: 45.3636

All items in the bag: {5,1,7,6,2,4 (10/11),,}

shubham@shubham-virtual-machine:~/semester4/daa/experiments/exp 5$
```

Conclusion:

Thus we have performed Fractional Knapsack Problem using Greedy Approach. Greedy algorithms help us to find the efficient and optimal or near-optimal solution to many real-life related problems.