Date: 11/8/25 to 16/08/25

Experiment No: 05

I. Aim: Implement solution for knapsack problem.

II. Theory:

Problem: Loading objects in a sack by satisfying weight constraint.
Objects given as input have to be loaded in a sack. Maximum weight sack can store is the constraint for solving knapsack problem.
The aim of greedy algorithm is to find optimal solution that gives maximum profit after loading sack. Fractional knapsack solution allows storing the object with partial weight.

Algorithm:

Example:

Object	1	2	3	4	5	6	7
Weight	2	3	5	7	1	4	1
Profit	10	5	15	7	6	18	3

Show stepwise procedure:

III. Program:

```
#include <stdio.h>

struct Item
{
   int weight;
   int profit;
};

void fractionalKnapsack(int W, struct Item items[], int n)
{
   struct Item tempI;
   float ratio[10];
   int i, j;
   int currentWeight = 0;
   float totalprofit = 0.0;

   for (i = 0; i < n; i++)
   {
        ratio[i] = (float)items[i].profit / items[i].weight;
    }

    // Sorting
    for (i = 0; i < n - 1; i++)
    {
        for (j = i + 1; j < n; j++)
        {
            if (ratio[i] < ratio[j])
            {
                float tempR = ratio[i];
                ratio[i] = ratio[j];
                ratio[j] = tempR;

                tempI = items[i];
                items[i] = items[j];
                items[j] = tempI;
            }
        }
    }
    for (i = 0; i < n; i++)
    {
        if (currentWeight + items[i].weight <= W)
        {
            currentWeight += items[i].weight;
            totalprofit += items[i].profit;
        }
        else
        {
            int remain = W - currentWeight;
            totalprofit += items[i].profit * ((float)remain / items[i].weight);
            break; // Knapsack is full
        }
    }

    printf("Maximum value in Knapsack = %.2f\n", totalprofit);
}

void main()
{
    int W = 15; // Knapsack capacity
    struct Item items[] = {{2,10}, {3,5}, {5,15},{7,7},{1,6},{4,18},{1,3}};
    int n = 7;

    fractionalKnapsack(W, items, n);
    getch();
}
```

IV. Output:

V. Conclusion:

Successfully implemented solution for Knapsack problem.