Name	Parth Gandhi
UID No.	2021300033
Class & Division	S.E. COMPS A (BATCH B)
Experiment No.	5

Aim: Experiment on solving fractional knapsack problem using greedy approach.

Theory:

The Fractional Knapsack Problem is a well-known optimization problem that involves packing items of different sizes and values into a knapsack, subject to a weight constraint. The goal is to maximize the total value of the items packed into the knapsack while not exceeding the knapsack's weight capacity.

The greedy approach is a natural way to solve this problem. In the greedy approach, we always choose the item with the highest value-to-weight ratio first and keep packing until we reach the knapsack's weight limit.

Algorithm:

1. Compute the value-to-weight ratio for each item by dividing its value by its weight.
2. Sort the items in decreasing order of their value-to-weight ratio.
3. Initialize the total value of packed items and the remaining capacity of the knapsack to zero.
4. Loop through each item in the sorted list: a. If the entire item can fit, pack it and add its value to the total value of packed items, and subtract its weight from the remaining capacity of the knapsack. b. If the entire item can't fit, pack a fractional amount of it that can fit, add the value of the packed fraction to the total value of packed items, and set the remaining capacity of the knapsack to zero. c. If there is no remaining capacity in the knapsack, exit the loop.
5. Return the total value of packed items.

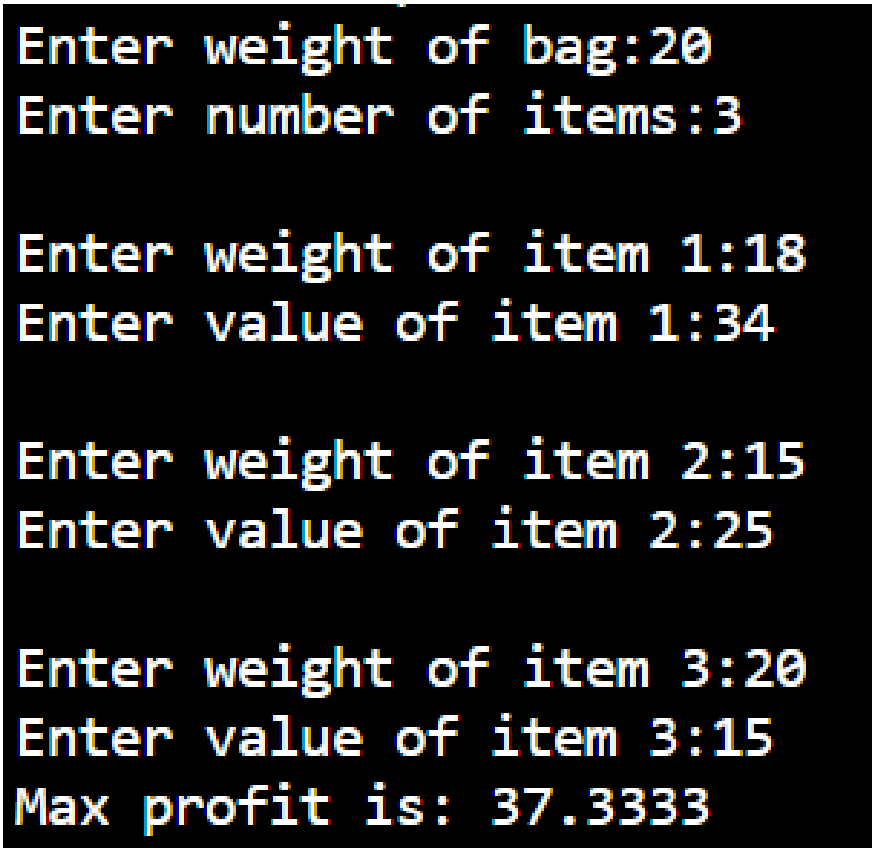
Code:

```
#include<bits/stdc++.h>
using namespace std;

int main()
{
   double n,w;
   cout<<"Enter weight of bag:";
   cin>>w;
   cout<<"Enter number of items:";
   cin>>n;
   double m=3;
   vector<vector<double>> a;
   for(int i=0;i<n;i++)
   {
      double x,y,z;
      cout<<"\nEnter weight of item "<<i+1<<":";
```

```
        cin>>x;
        cout<<"Enter value of item "<<i+1<<":";
        cin>>y;
        z=y/x;
        a.push_back({z,x,y,double(i+1)});
    }
    sort(a.begin(),a.end(),greater <>());
    double max_profit=0;
    for(int i=0;i<n;i++)
    {
        if(a[i][1]<=w)
        {
            w-=a[i][1];
            max_profit+=a[i][2];
        }
        else
        {
            max_profit+=a[i][0]*w;
            break;
        }
    }
    cout<<"Max profit is: "<<max_profit;
    return 0;
}
```

Output:



Conclusion: Successfully wrote a program to implement fractional knapsack problem using

greedy method.