S.No: 15	Exp. Name: ***Program to find the solution of fractional knapsack problem using greedy approach***	**Date:**

Aim:

Program to find the solution of fractional knapsack problem using greedy approach

Source Code:

knapasak.c

```
# include<stdio.h>
void knapsack(int n, float weight[], float profit[], float capacity){
   float x[20], tp= 0;
   int i, j, u;
   u=capacity;
   for (i=0;i<n;i++)
   x[i]=0.0;
   for (i=0;i<n;i++) {
      if(weight[i]>u)
      break;
      else
      {
         x[i]=1.0;
         tp= tp+profit[i];
         u=u-weight[i];
         }

   }
   if(i<n)
   x[i]=u/weight[i];
   tp= tp + (x[i]*profit[i]);
   printf("The result vector is:- \n");
   for(i=0;i<n;i++)
   printf("%.2f   ",x[i]);
   printf("\nMaximum profit is:- %.2f", tp);

}
void main() {
   float weight[20], profit[20], capacity;
   int n, i ,j;
   float ratio[20],     temp;
   printf ("Enter the no. of objects:- ");
   scanf ("%d", &n);
   printf ("Enter the Weight, Value(Profit) of each object:- \n");
   for (i=0; i<n; i++){
      printf("item %d:",i+1);
      scanf("%f %f", &weight[i], &profit[i]);
      }     printf ("Enter the capacity of knapsack:- ");
      scanf ("%f", &capacity);
      for (i=0; i<n; i++){ratio[i]=profit[i]/weight[i];
      }
      for(i=0; i<n; i++)
      {
         for(j=i+1;j< n;
         j++)
         {
            if(ratio[i]<ratio[j])
            {
               temp= ratio[j];
               ratio[j]= ratio[i];
               ratio[i]= temp;
               temp= weight[j];
```

```
                weight[j]= weight[i];
                weight[i]= temp;
                temp= profit[j];
                profit[j]= profit[i];
                profit[i]= temp;
                }
                }
                }
                knapsack(n, weight, profit, capacity);
                }
```

Execution Results - All test cases have succeeded!

Test Case - 1

User Output

```
Enter the no. of objects:-  3
Enter the Weight, Value(Profit) of each object:-  10 60
item 1: 10 60
item 2: 20 100
item 3: 30 120
Enter the capacity of knapsack:-  50
The result vector is:-
1.00    1.00    0.67
Maximum profit is:- 240.00
```

Test Case - 2

User Output

```
Enter the no. of objects:-  5
Enter the Weight, Value(Profit) of each object:-  10.0 25.0
item 1: 10.0 25.0
item 2: 10.0 25.0
item 3: 10.0 25.0
item 4: 4.0 6.0
item 5: 2.0 2.0
Enter the capacity of knapsack:-  70
The result vector is:-
1.00    1.00    1.00    1.00    1.00
Maximum profit is:- 83.00
```