NAME:	Arnav Hoskote
UID:	2021300044
SUBJECT	Design and Analysis of Algorithm
EXPERIMENT NO :	05
DATE OF PERFORMANCE	03/04/2023
DATE OF SUBMISSION	11/04/2023
AIM:	To implement fractional knapsack problem and calculate profit.
PROBLEM STATEMENT 1:	**Fractional knapsack problem**

ALGORITHM and THEORY:	Algorithm: Greedy-Fractional-Knapsack (w[1..n], p[1..n], W)
	for i = 1 to n
	  do x[i] = 0
	weight = 0
	for i = 1 to n
	  if weight + w[i] ≤ W then
	  x[i] = 1
	  weight = weight + w[i]
	  else
	  x[i] = (W - weight) / w[i]
	  weight = W
	  break
	return x

PROGRAM:

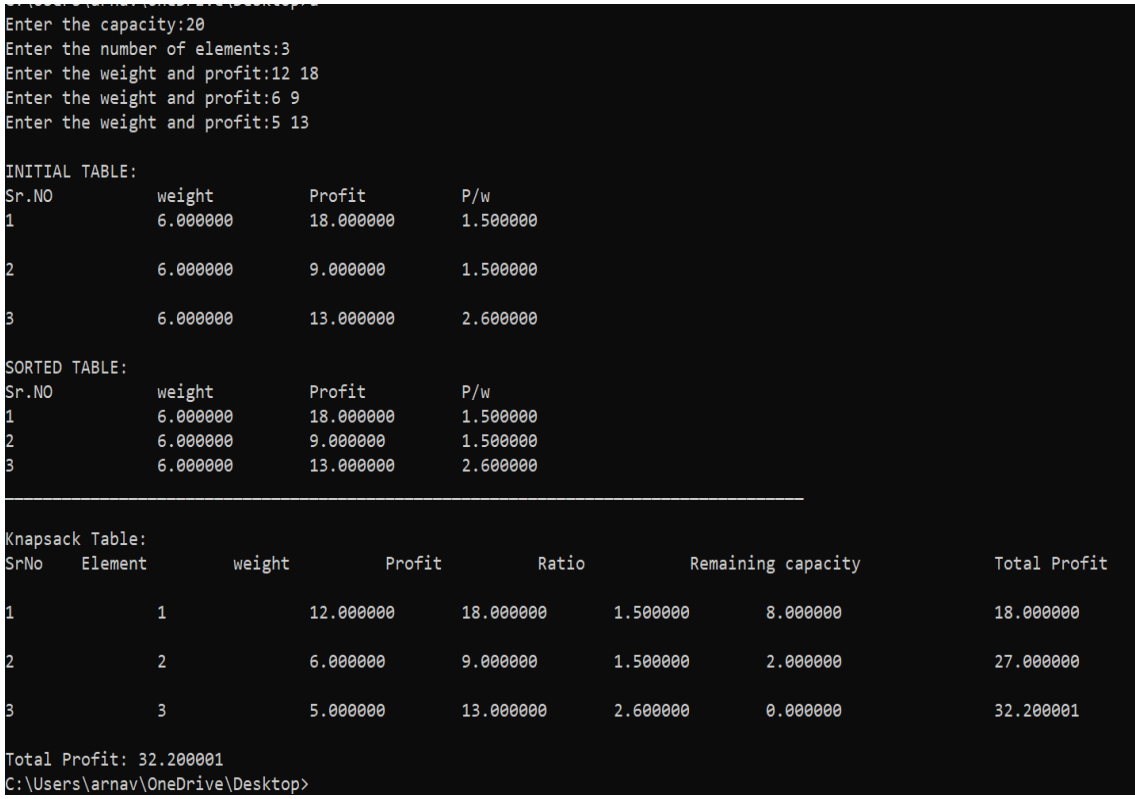
```
#include<stdio.h>
#include<stdlib.h>
struct Item
{
   int SrNo;
   float w,profit,ratio;
};
void sort(int n,struct Item a[n])
{
   int i,j;
   struct Item temp;
   for(i=0;i<n-1;i++)
   {
      for(j=0;j<n-1;j++)
      {
         if(a[j].ratio>a[j+1].ratio)
         {
            temp=a[j];
            a[j]=a[j-1];
            a[j-1]=temp;
         }
      }
   }
}
void main()
{
   int n,i;
   float W,p=0;
   printf("Enter the capacity:");
   scanf("%f",&W);
   printf("Enter the number of elements:");
   scanf("%d",&n);
   struct Item a[n];
   for(i=0;i<n;i++)
```

```
    {
        printf("Enter the weight and profit:");
        scanf("%f %f",&a[i].w,&a[i].profit);
        a[i].ratio=a[i].profit/a[i].w;
        a[i].SrNo=i+1;
    }
    printf("\nINITIAL TABLE:\nSr.NO\t\tweight\t\tProfit\t\tP/w");
    for(i=0;i<n;i++)
    {
        printf("\n%d\t\t%f\t%f\t%f\n",a[i].SrNo,a[1].w,a[i].profit,a[i].ratio);
    }
    sort(n,a);
    printf("\nSORTED TABLE:\nSr.NO\t\tweight\t\tProfit\t\tP/w\n");
    for(i=0;i<n;i++)
    {
        printf("%d\t\t%f\t%f\t%f\n",a[i].SrNo,a[1].w,a[i].profit,a[i].ratio);
    }

printf("_____
    printf("Knapsack Table:\nSrNo\tElement\t\tweight\t\tProfit\t\tRatio\t\tRe
    for(i=0;i<n;i++)
    {
        if(W>=a[i].w)
        {
            W-=a[i].w;
            p+=a[i].profit;
        }
        else if(W<=a[i].w)
        {
            p+=W*a[i].ratio;
            W=0;
        }
        printf("\n%d\t\t%d\t\t%f\t%f\t%f\t%f\t\t%f\n",(i+1)
,a[i].SrNo,a[i].w,a[i].profit,a[i].ratio,W,p);
```

```
        if(W==0)
        {
            break;
        }
    }
    printf("\nTotal Profit: %f",p);
}
```

OUTPUT:



```
Enter the capacity:20
Enter the number of elements:3
Enter the weight and profit:12 18
Enter the weight and profit:6 9
Enter the weight and profit:5 13

INITIAL TABLE:
Sr.NO           weight          Profit          P/w
1               6.000000        18.000000       1.500000

2               6.000000        9.000000        1.500000

3               6.000000        13.000000       2.600000

SORTED TABLE:
Sr.NO           weight          Profit          P/w
1               6.000000        18.000000       1.500000
2               6.000000        9.000000        1.500000
3               6.000000        13.000000       2.600000
--------------------------------------------------------------------------------

Knapsack Table:
SrNo    Element         weight          Profit          Ratio           Remaining capacity              Total Profit

1               1               12.000000       18.000000       1.500000        8.000000                18.000000

2               2               6.000000        9.000000        1.500000        2.000000                27.000000

3               3               5.000000        13.000000       2.600000        0.000000                32.200001

Total Profit: 32.200001
C:\Users\arnav\OneDrive\Desktop>
```

CONCLUSION:

By performing above experiment I have understood knapsack problem and I have been able to calculate the profit accurately.