NAME:	Aditya Choudhary
UID:	2021300022
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>
#include <time.h>
clock_t start ,end;
struct Item
{
   int SrNo;
   float w,profit,ratio;
};
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[i].w,a[i].profit,a[i].ratio);
   }
   start = clock();
   int j;
   struct Item temp;
   for(i=0;i<n;i++)
```

```
    {
        for(j=i;j<n;j++)
        {
            if(a[j].ratio>a[i].ratio)
            {
                temp=a[j];
                a[j]=a[i];
                a[i]=temp;
            }
        }
    }
    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[i].w,a[i].profit,a[i].ratio);
    }

printf("_____
    printf("Knapsack Table:\nSrNo\tElement\t\tweight\t\tProfit\t\tRatio\t\tRem
    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\n",p);
    end=clock();
    printf("Time taken by program:");
    printf("%f",(double)(end-start)/CLOCKS_PER_SEC);
}
```

OUTPUT:

```
Enter the capacity:20
Enter the number of elements:3
Enter the weight and profit:18
24
Enter the weight and profit:15
25
Enter the weight and profit:20
15
INITIAL TABLE:
Sr.NO        weight        Profit        P/w
1        18.000000 24.000000 1.333333

2        15.000000 25.000000 1.666667

3        20.000000 15.000000 0.750000

SORTED TABLE:
Sr.NO        weight        Profit        P/w
2        15.000000 25.000000 1.666667
1        18.000000 24.000000 1.333333
3        20.000000 15.000000 0.750000
```

	Knapsack Table: SrNo Element weight Profit Ratio Profit 1 2 15.000000 25.000000 1.666667 5.000000 2 1 18.000000 24.000000 1.333333 0.000000 Total Profit: 31.666668 Time taken by program:0.000078
CONCLUSION:	By performing above experiment I have understood knapsack problem and I have been able to calculate the profit accurately.