

NAME:	Vedant Deshmukh
UID:	2021300025
SUBJECT	DAA
EXPERIMENT NO :	05
AIM:	Experiment based on greedy approach ,performing fractional Knapsack.
PROBLEM STATEMENT 1:	
ALGORITHM	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 + $w[i] \leq W$ then $x[i] = 1$ weight = weight + $w[i]$ else $x[i] = (W - \text{weight}) / w[i]$ weight = W break return

SOLVED EXAMPL E

* Knapsack problem
 n = 7
 m = 15
 objects 0 1 2 3 4 5 6 7
 profits p 10 5 15 7 6 18 3
 weights w 2 3 5 7 1 4 1
 p/w 5 1.3 3 1 6 4.5 3
 x_1 x_2 x_3 x_4 x_5 x_6 x_7
 $0 \leq x_i \leq 1$
 Ex: $1 \times 2 + 2 \times 3 + 1 \times 5$
 $3 \times 0 + 7$
 $1 \times 1 + 1 \times 4 + 1 \times 1$
 $15 - 1 = 14$
 $14 - 2 = 12$
 $12 - 4 = 8$
 $8 - 5 = 3$
 $3 - 1 = 2$
 $2 - 2 = 0$
 $\Rightarrow 9 + 5 + 1$
 $\Rightarrow 15$
 weight \checkmark
 Ex: $1 \times 10 + \frac{2}{3} \times 5 + \frac{1 \times 15}{3} + 0 \times 7 + \frac{1 \times 6}{3} + \frac{1 \times 18}{3}$
 $\Rightarrow 10 + 3.34 + 5 + 6 + 18 + 3$
 $\Rightarrow 55.34$

**PROGRA
M:**

```
#include<bits/stdc++.h>
using namespace std;
struct Knapsack
{
    int id;
    float prof,wt,ratio;
};
void sort(int n,struct Knapsack arr[10])
{
    struct Knapsack temp;
    int i,j;
    for(i=0;i<n;i++)
    {
```

```

        for(j=i+1;j<n;j++)
        {
            if(arr[j].ratio>arr[i].ratio)
            {
                temp=arr[j];
                arr[j]=arr[i];
                arr[i]=temp;
            }
        }
    }
}

int main()
{
    int cap,i,j,n;
    float p,w;
    float profit;

    cout<<"Enter Capacity "<<endl;
    cin>>cap;
    cout<<"Enter number of Elements"<<endl;
    cin>>n;
    struct Knapsack kn[n];
    for(i=0;i<n;i++)
    {
        cout<<"Enter Profit and weight element of "<<(i+1)<<" item"<<endl;
        cin>>p>>w;
        kn[i].prof=p;
        kn[i].wt=w;
        kn[i].ratio=p/w;
        kn[i].id=(i+1);

    }
    printf("\nID\t\tPROFIT\t\t\tWEIGHT\t\t\t\n");
    for(i=0;i<n;i++)
    {
        printf("\n%d\t\t%f\t\t%f\t\t\t\n",kn[i].id,kn[i].prof,kn[i].wt);
    }
    sort(n,kn);
    printf("Sorted with p/w Ratio");
    printf("\nID\t\tPROFIT\t\t\tWEIGHT\t\t\tRATIO\t\t\t\n");
    for(i=0;i<n;i++)
    {
        printf("\n%d\t\t%f\t\t%f\t\t%f\t\t\t\n",kn[i].id,kn[i].prof,kn[i].wt,kn[i].ratio);
    }
}

```

```
for(i=0;i<n;i++)
{
    if(cap>kn[i].wt)
    {
        cap=cap-kn[i].wt;
        profit=profit+kn[i].prof;

    }
    else if(cap<=kn[i].wt)
    {
        profit=profit+kn[i].ratio*cap;
        cap=0;

    }
    if(cap==0)
    {
        break;
    }
}
printf("Total Profit Obtained after performing fractional knapsack is %f",pr
```

RESULT (SNAPSHOT)

```
Search results - vedant.d... GitHub Link_SY Div-A 202... Online C Compiler - online...  
https://www.onlinegdb.com/online_c_compiler 110%  
input  
Enter Capacity  
15  
Enter number of Elements  
7  
Enter Profit and weight element of 1 item  
10  
2  
Enter Profit and weight element of 2 item  
5  
3  
Enter Profit and weight element of 3 item  
15  
5  
Enter Profit and weight element of 4 item  
7  
7  
Enter Profit and weight element of 5 item  
6  
1  
Enter Profit and weight element of 6 item  
18  
4  
Enter Profit and weight element of 7 item  
3  
1  
ID          PROFIT          WEIGHT
```

```
Sorted with p/w Ratio  
ID          PROFIT          WEIGHT          RATIO  
5           6.000000          1.000000          6.000000  
1           10.000000         2.000000          5.000000  
6           18.000000         4.000000          4.500000  
3           15.000000         5.000000          3.000000  
7           3.000000          1.000000          3.000000  
2           5.000000         3.000000          1.666667  
4           7.000000         7.000000          1.000000  
Total Profit Obtained after performing fractional knapsack is 55.333332
```

CONCLUSION:	Through this experiment I learned how to implement Fractional knapsack using greedy approach