Practical-8

Aim: Knapsack-algorithm

Code:

```
#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++){

      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("\nThe result vactor is :- ");

   for(i=0;i<n;i++){

      printf("%f\t",x[i]);

   }

   printf("\nMaximum profit is:- %f", tp);
```

```
}
int main() {
        float weight[20], profit[20],capacity;
        int num,i,j;
   float ratio[20],temp;
   printf("\nEnter the no. of objects:- ");
        scanf("%d",&num);
        printf("\nEnter the wts and profits of each
objects:- ");
        for(i=0;i<num;i++){
      scanf("%f %f",&weight[i],&profit[i]);
        }
        printf("\nEnter the capacity of knapsack:- ");
        scanf("%f",&capacity);
   for(i=0;i<num;i++){
            ratio[i] = profit[i]/weight[i];
        }
        for(i = 0;i<num;i++){
            for(j=i+1;j<num;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(num, weight,profit,capacity);
        return 0;
}
```

OUTPUT:

Status Successfully executed **Date** 2022-05-24 04:25:09 **Time** 0.008021 sec **Mem** 5456 kB

Input

```
5
4 40
3 30
2 20
1 10
6 60
20
```

Output

```
Enter the no. of objects:-
Enter the wts and profits of each object:-
Enter the capacityacity of knapsack:-
The result vector is:- 1.000000 1.000000    1.000000    1.000000    1.000000
Maximum profit is:- 160.000000
```