**PROGRAM-11**

**AIM -** Write an algorithm and program to implement Knapsack problem.

1)0/1 Knapsack-

**ALGORITHM-**

Dynamic-0-1-knapsack (v, w, n, W)

for w = 0 to W do

c[0, w] = 0

for i = 1 to n do

c[i, 0] = 0

for w = 1 to W do

if wi ≤ w then

if vi + c[i-1, w-wi] then

c[i, w] = vi + c[i-1, w-wi]

else c[i, w] = c[i-1, w]

else

c[i, w] = c[i-1, w]

**SOURCE CODE-**

#include<stdio.h>

int max(int a, int b) { return (a > b)? a : b; }

int knapSack(int W, int wt[], int val[], int n)

{

int i, w;

int K[n+1][W+1];

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

{

for (w = 0; w <= W; w++)

{

if (i==0 || w==0)

K[i][w] = 0;

else if (wt[i-1] <= w)

K[i][w] = max(val[i-1] + K[i-1][w-wt[i-1]], K[i-1][w]);

else

K[i][w] = K[i-1][w];

}

}

return K[n][W];

}

int main()

{

int i, n, val[20], wt[20], W;

printf("Enter number of items:");

scanf("%d", &n);

printf("Enter value and weight of items:\n");

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

scanf("%d%d", &val[i], &wt[i]);

}

printf("Enter size of knapsack:");

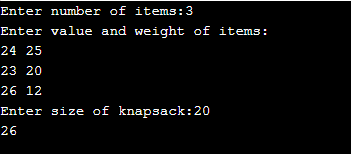
scanf("%d", &W);

printf("%d", knapSack(W, wt, val, n));

return 0;

}

**OUTPUT-**

****

**AIM -** Write an algorithm and program to implement Knapsack problem.

**2) Fractional Knapsack-**

**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] ≤ W then

x[i] = 1

weight = weight + w[i]

else

x[i] = (W - weight) / w[i]

weight = W

break

return x

**SOURCE CODE-**

#include<stdio.h>

#include<time.h>

#include<conio.h>

int knapsack(float capacity, int n, float weight[], float profit[])

{

float x[20], totalprofit,y;

int i,j;

y=capacity;

totalprofit=0;

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

x[i]=0.0;

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

{

if(weight[i] > y)

break;

else

{

x[i]=1.0;

totalprofit=totalprofit+profit[i];

y=y-weight[i];

}

}

if(i < n)

x[i]=y/weight[i];

totalprofit=totalprofit+(x[i]\*profit[i]);

printf("The selected elements are:-\n ");

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

if(x[i]==1.0)

printf("\nProfit is %f with weight %f ", profit[i], weight[i]);

else if(x[i] > 0.0)

printf("\n%f part of Profit %f with weight %f", x[i], profit[i], weight[i]);

printf("\nTotal profit for %d objects with capacity %f = %f\n\n", n, capacity,totalprofit);

}

int main()

{

float weight[20],profit[20],ratio[20], t1,t2,t3;

int n;

time\_t start,stop;

float capacity;

int i,j;

printf("Enter number of objects: ");

scanf("%d", &n);

printf("\nEnter the capacity of knapsack: ");

scanf("%f", &capacity);

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

{

printf("\nEnter %d(th) profit: ", (i+1));

scanf("%f", &profit[i]);

printf("Enter %d(th) weight: ", (i+1));

scanf("%f", &weight[i]);

ratio[i]=profit[i]/weight[i];

}

start=time(NULL);

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

for(j=0;j < n;j++)

{

if(ratio[i] > ratio[j])

{

t1=ratio[i];

ratio[i]=ratio[j];

ratio[j]=t1;

t2=weight[i];

weight[i]=weight[j];

weight[j]=t2;

t3=profit[i];

profit[i]=profit[j];

profit[j]=t3;

}

}

knapsack(capacity,n,weight,profit);

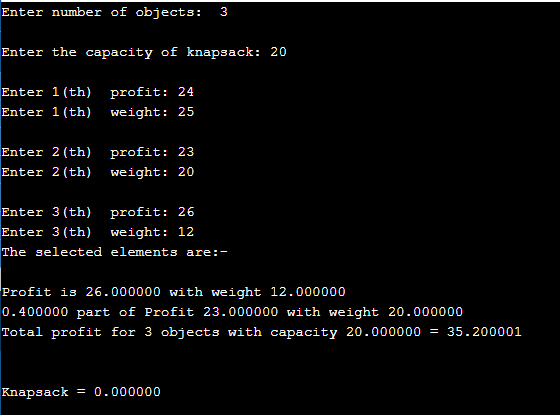
stop=time(NULL);

printf("\nKnapsack = %f\n", difftime(stop,start));

return 0;

}

**OUTPUT-**

****