**PROGRAM-15**

**AIM –Write an algorithm and program to implement 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>

#include<conio.h>

int sum=0;

int max(int a,int b)

{

if(a>b)

return a;

else

return b;

}

void knapsack(int m,int n,int w[],int p[])

{

int v[100][200],x[10],i,j;

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

v[0][i]=0;

for(i=1;i<=n;i++)

{

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

{

if(j>=w[i])

v[i][j]=max(v[i-1][j],v[i-1][j-w[i]]+p[i]);

else

v[i][j]=v[i-1][j];

}

}

for(i=1;i<=n;i++)

x[i]=0;

i=n;

j=m;

while(i>0 && j>0)

{

if(v[i][j]!=v[i-1][j])

{

x[i]=1;

j=j-w[i];

}

i--;

}

printf("\nTHE OPTIMAL SET OF WEIGHTS IS:\n");

for(i=1;i<=n;i++)

{

if(x[i]==1)

{

printf("X%d=1\t",i);

sum=sum+p[i];

}

else

printf("X%d=0\t",i);

}

printf("\nTotal profit = %d",sum);

}

void main()

{

int w[10],p[10],i,m,n;

printf("\t0/1 KNAPSACK PROBLEM\n\n");

printf("ENTER THE NUMBER OF ITEMS: ");

scanf("%d",&n);

printf("ENTER THE WEIGHTS OF THE ITEMS:\n");

for(i=1;i<=n;i++)

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

printf("ENTER THE PROFITS OF THE ITEMS:\n");

for(i=1;i<=n;i++)

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

printf("ENTER THE CAPACITY OF KNAPSACK: ");

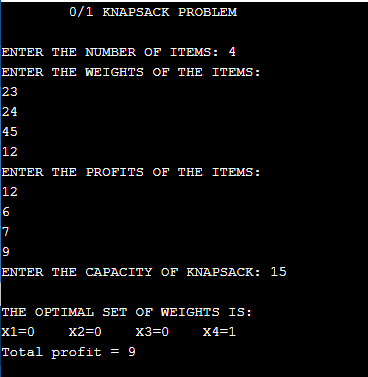
scanf("%d",&m);

knapsack(m,n,w,p);

getch();

}

**OUTPUT-**

****