**Practical-11**

***AIM:-*Implementation of a knapsack problem using dynamic programming**

**Input:**

#include<stdio.h>

#include<conio.h>

#include<stdlib.h>

int table[5][6];

void main()

{

int w[]={0,2,3,4,5};

int v[]={0,3,4,5,6};

int W=5;

int n=4;

void DKP(int n,int W,int w[],int v[]);

clrscr();

printf("\n\t\t 0/1 Knapsack Problem using Dynamic Programming");

/\*initialization of table\*/

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

{

for(int j=0;j<=W;j++)

{

table[i][j]=0;

}

}

DKP(n,W,w,v);

}

int max(int a,int b)

{

if(a>b)

return a;

else

return b;

}

void DKP(int n,int W,int w[5],int v[5])

{

void find\_item(int,int,int[]);

int i,j;

int val1,val2;

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

{

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

{

table[i][0]=0;

table[0][j]=0;

}

}

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

{

for(j=1;j<=W;j++)

{

if(j<w[i])

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

else if(j>=w[i])

{

val1=table[i-1][j];

val2=v[i]+table[i-1][j-w[i]];

table[i][j]=max(val1,val2);

}

}

}

printf("\n Table constructed using dynamic programming is ...\n");

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

{

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

printf(" %d",table[i][j]);

printf("\n");

}

find\_item(n,W,w);

}

void find\_item(int i,int k,int w[5])

{

printf("\nFor the Knapsack...");

while(i>0 && k>0)

{

if(table[i][k]!=table[i-1][k])

{

printf("\nItem %d is selected",i);

i=i-1;

k=k-w[i];

}

else

i=i-1; }