#include <stdio.h>

#define MAX 20

int W[MAX];

int p[MAX];

int n;

int M;

int v[MAX][MAX];

int keep[MAX][MAX];

void ReadObjects();

int Knapsack();

int main()

{

int Optsoln;

ReadObjects();

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

v[0][i]=0;

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

v[i][0]=0;

Optsoln=Knapsack();

printf("Optimal solution=%d\n",Optsoln);

return 0;

}

void ReadObjects()

{

printf("Knapsack problem-Dynamic programming solution:\n");

printf("Enter the max capacity of Knapsack:");

scanf("%d",&M);

printf("Enter number of objects:");

scanf("%d",&n);

printf("Enter the weights:\n");

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

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

printf("Enter the profits:\n");

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

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

}

int Knapsack()

{

int r;

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

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

if((W[i]<=j)&&(p[i]+v[i-1][j-W[i]]>v[i-1][j]))

{

v[i][j]=p[i]+v[i-1][j-W[i]];

keep[i][j]=1;

}

else

{

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

keep[i][j]=0;

}

r=M;

printf("Items=\n");

for(int i=n;i>0;i--)

if(keep[i][r]==1)

{

printf("%d\n", i);

r=r-W[i];

}

printf("\n");

return v[n][M];

}

Output:

Knapsack problem-Dynamic programming solution:

Enter the max capacity of Knapsack:15

Enter number of objects:7

Enter the weights:

1 3 5 4 1 3 2

Enter the profits:

5 10 15 7 8 9 4

Items=

7

6

5

3

2

1

Optimal solution=51

=== Code Execution Successful ===