```
 1: #include<iostream>
 2:
 3: #define MAXW 20
 4: #define MAXN 20
 5:
 6: using namespace std;
 7:
 8: void ZeroOneKnapsackDP(int wt[], int p[], int W,
    int n, int m[][MAXW])
 9: {
10:     int w,i,x,y;
11:
12:     for(w=0;w<=W;w++)
13:         m[0][w] = 0;
14:
15:     for(i=0;i<=n;i++)
16:         m[i][0] = 0;
17:
18:     for(i=1;i<=n;i++)
19:     {
20:         for(w=1;w<=wt[i]-1;w++)
21:             m[i][w] = m[i-1][w];
22:
23:         for(w=wt[i];w<=W;w++)
24:         {
25:             x = m[i-1][w];
26:             y = p[i] + m[i-1][w-wt[i]];
27:
28:             if(x>y)
29:                 m[i][w] = x;
```

```
30:                 else
31:                     m[i][w] = y;
32:             }
33:         }
34: }
35:
36: int main()
37: {
38:     int i,j;
39:
40:     int wt[MAXN] = {0, 3, 4, 5, 6};
41:     int p[MAXN] = {0, 2, 3, 4, 1};
42:     int W = 8;
43:     int n = 4;
44:     int m[MAXN][MAXW];
45:
46:     ZeroOneKnapsackDP(wt,p,W,n,m);
47:
48:     cout<<"\n\nProfit Matrix - m\n";
49:     for(i=0;i<=n;i++)
50:     {
51:         for(j=0;j<=W;j++)
52:         {
53:             cout<<"\t"<<m[i][j];
54:         }
55:         cout<<"\n";
56:     }
57:
58:     cout<<"\n\nMaximum Profit: "<<m[n][W];
59:
```

```
60: }
```