Keegan Vaz
SE CMPN B
42

AOA EXPERIMENT 7

Aim: Write a program to solve the **0/1 Knapsack** problem using **DynamicProgramming**.

Problem statement:
Write a program to solve the 0-1 Knapsack problem taking as inputs the capacity of the knapsack, the number of items, the weights and values of n items.

Input: Give the below input:

Capacity of the knapsack=**11**,
number of items=**5**,
weights=**{1,2,5,6,7}**,
values={**1,6,18,22,28}**

Output: Display the table generated and also display the final optimal solution and maximum profit received.

(Paste your code and output below)

Code:

```
#include <bits/stdc++.h>

using namespace std;



int dp[101][100001];

int kps(int W, int wt[], int val[], int n)
{
   int i, w, j;
   int dp[n + 1][W + 1];

   for(i = 0; i <= n; i++)
   {
      for(w = 0; w <= W; w++)
      {
         if (i == 0 || w == 0)
            dp[i][w] = 0;
         else if (wt[i - 1] <= w){
            dp[i][w] = max(val[i - 1] + dp[i - 1][w - wt[i - 1]],
                           dp[i - 1][w]);
         }
         else
            dp[i][w] = dp[i - 1][w];
      }
   }
   cout<<"DP Table: "<<endl;
   for(i=0;i<n+1;i++){
      for(j=0;j<W+1;j++){
         cout<<dp[i][j]<<"";
      }
      cout<<endl;
   }
   int ans[100],c=0;
   i=n,j=W;
   while(i>0 && j>0){
      if (dp[i-1][j]!=dp[i][j]){
         i--,j-=wt[i];
         ans[c]=i;
```

```
            c++;
        }else i--;
    }
    cout<<"Items included in Knapsack: "<<endl;
    cout<<"Weight\tValue"<<endl;
    for (i=0;i<c;i++){
        cout<<wt[ans[i]]<<"\t"<<val[ans[i]]<<endl;
    }
    cout<<"Final Optimal Solution : "<<endl;
    j=c-1;
    for(i=0;i<n;i++){
        if (ans[j]==i){
            cout<<1<<"";
            j--;
        }else{
            cout<<0<<"";
        }
    }
    cout<<endl;
    return dp[n][W];
}

int main()
{
    int n,W,i,j;
    cout<<"Enter Number of items and Capacity of Knapsack: "<<endl;
    cin>>n>>W;
    int v[n],w[n];
    cout<<"Enter Weights and their Values: "<<endl;
    for(i=0;i<n;i++){
        cin>>w[i]>>v[i];
    }
    memset(dp,-1,sizeof(dp));
    cout<<"Maximum Profit Obtained: "<<kps(W,w,v,n)<<endl;
    return 0;
}
```

Output:

```
Enter Number of items and Capacity of Knapsack:
5 11
Enter Weights and their Values:
1 1
2 6
5 18
6 22
7 28
DP Table:
0 0 0 0 0 0 0 0 0 0 0 0
0 1 1 1 1 1 1 1 1 1 1 1
0 1 6 7 7 7 7 7 7 7 7 7
0 1 6 7 7 18 19 24 25 25 25 25
0 1 6 7 7 18 22 24 28 29 29 40
0 1 6 7 7 18 22 28 29 34 35 40
Items included in Knapsack:
Weight  Value
6       22
5       18
Final Optimal Solution :
0 0 1 1 0
Maximum Profit Obtained: 40
```