Theory Assignment no. 3. CS 22203

Ex. 1. Refer to the recursive procedure that follows the Dynamic Programming (DP) backward approach discussed in class for solving the 0/1 Knapsack problem. Draw the recursion tree depicting the implementation of the procedure for the following instance. Give the solution along with the total profit.
n=3, $(p_1,p_2,p_3)=(10,20,15)$, $(w_1,w_2,w_3)=(3,4,7)$, M=7.

Ex. 2 Let G =(V,E) be a directed graph with nodes $v_1, v_2, .., v_n$. We say that G is an ordered graph if it has the following properties:

Each edge goes from a node with a lower index to a node with a higher index. That is, every directed edge has the form (v_i, v_j) with $i<j$.

Each node except v_n has at least one edge leaving it. That is, for every node v_i, i=1, 2, …n-1, there is at least one edge of the form (v_i, v_j). The length of a path is the number of edges in it. The goal in this question is to solve the following problem:

Given an ordered graph G, find the length of the longest path that begins at v_1 and ends at v_n.

a) Show that the following algorithm does not correctly solve this problem by giving an example of an ordered graph on which it does not return the correct answer.

```
Set w = v1
Set L = 0
While there is an edge out of the node w
        Choose the edge (w, vj) for which j is as small as possible
        Set w = vj
        L = L+1
end While
Return L as the length of the longest path.
```

In your example, say what the correct answer is and also what the algorithm above finds.

b) Give an efficient Dynamic Programming algorithm that takes an ordered graph G and returns the length of the longest path that begins at v_1 and ends at v_n. [Hint: First, derive a solution (i.e., a recursive function) that exploits the principle of optimality (as was done for I/O knapsack in class) and build the algorithm from it].
