Problem 1 and 2 solving Strategy:
Both the problem uses dynamic programming approach. We are given with 2 lists of length and prices respectively.
The solution computes a T matrix(dictionary in python code) of order n*(n+1) where n is the length of “lengths” list taken in the program.All the values are 0 in column 0 and for row 1 we compute it as:
T[i,j]=T[i,j-1]+price[i-1] i.e., it computes values if we make the rods of given length n with all of length 1 taken together.
Later we compute the other values using the relation:
T[i,j]=max(T[i-1,j],price[i-1]+T[i,j-i])

We find the matrix and now we backtrack to find which all rods will be taken using the function backtrack_profit.We start from the bottom-rightmost element and check from where that value was obtained. If the value was copied from previous row we move up the row and consider the new element found, Else we move to the element which was responsible for computing the value we are currently looking at. We add the latter to our solution list and repeat the same till we don't reach 0.

The T[n,n] contains the final cost and solution list contains the lengths of rods taken for the purpose as our final solution.

Complexity of the program is: O(n^2)

Similarly in Problem 2 we repeat the same till matrix T computation but while backtracking we subtract the cost of the rod we are adding from T[n,n] and thus obtain the final solution for the required.
The complexity for this program again is O(n^2)