ESO207: Data Structures and Algorithms
Team: Queute Coders
Aayush (190015)
Abhimanyu Sethia (190023)
Rishabh Dugaye (190701)

Assignment 4

Queute Coders

Q4 (c) *What is the complexity of your algorithms in (a) and (b). Do you see difference in the time taken by programs in (a) and (b), in practice on large inputs*

Time Complexity Analysis for Top-Down Approach:
It is easy to see that the end_space_cost function takes constant time to execute. Now, the for loop in the top_down_cost will have at most min(M,n) iterations. Since the loop goes from 0 to n-k-1, and k varies from 0 to n-1, the loop will have at most n iterations. Also, any line can have at most $(M + 1)/2$ words (since every word has atleast length 1, and is followed by a space). Since we have a break condition if the end_space_cost becomes negative (i.e. length of words + space exceeds M), the number of iterations of the for loop is also bounded by M.
Now, the recursive call will be bounded by n, as each recursive call is on a smaller sub-problem, and the number of sub-problems is itself bounded by n. Thus the overall complexity of the algorithm is O(n*min(M,n)).

Time Complexity Analysis for Bottom- Up Approach:
We have argued in the previous case that the inner loop will be bounded by O(min(M,n)) iterations. Now, the outer loop in this approach has n-1 iterations, hence, the resulting overall complexity is O(n*min(M,n)).

In-Practise Comparison
In practice the top down approach took 2153 microseconds on average for 100 runs on a test case with $n = 10^5$ whereas bottom up approach took 1506 microseconds for the same test case. Although both the algorithms have same time complexities but the top down approach takes more time to execute because it involves the cost to be calculated recursively which takes more time in practice as compared to bottom up approach which calculates the cost iteratively.