Problem Set 1

1. Design a Dynamic Programming algorithm to count the number of choices for Text Segmentation. That is, input is an array $A[1 \dots n]$ of letters and output is the no. of partitions of A into words.

2. Input is two arrays $X[1 \dots k]$ and $Y[1 \dots n]$ of letters, where $k \leq n$. (i) Design and analyze an algorithm to decide whether X is a subsequence of Y. (ii) Describe and analyze an algorithm to determine whether X occurs as two disjoint subsequences of Y .

3. Let $A[1 \dots m]$ and $B[1 \dots n]$ be two arbitrary arrays. A common subsequence of A and B is another sequence that is a subsequence of both A and B. Describe an efficient algorithm to compute the length of the longest common subsequence of A and B.

4. Let $A[1 \dots m]$ and $B[1 \dots n]$ be two arbitrary arrays. A common super-sequence of A and B is another sequence that contains both A and B as subsequences. Describe an efficient algorithm to compute the length of the shortest common super-sequence of A and B.

5. Design a dynamic programming algorithm to find the n^{th} Fibonacci number. (In a Fibonacci sequence, each number is the sum of the two preceding ones, starting from 0 and 1). Compare the running time of the DP algorithm with the naive implementation of the recurrence relation.