CS3230
Tutorial 9

1. Compute the longest common subsequence of

   (a) $SLWOVNNDK$ and $ALWGQVNBBK$.

   (b) $AGCGATAGC$ and $ACAGATGAG$

2. Show that if X and Y are two sequences starting with a, then the longest common subsequence of X and Y starts with an a.

3. Give a counterexample to the following claims:

   (a) If $X = X[1] \dots X[m]$ and $Y = Y[1] \dots Y[n]$, and $X[m] \neq Y[n]$, then the longest common subsequence of X and Y must end in either $X[m]$ or $Y[n]$.

   (b) If $X = X[1] \dots X[m]$ and $Y = Y[1] \dots Y[n]$, and $X[m] \neq Y[n]$, then the longest common subsequence of X and Y must not end with either $X[m]$ or $Y[n]$.

4. Consider the following problem:

   A relation on a set A is a subset of $A \times A$.

   A relation T is called transitive if the following holds for all $a, b, c \in A$:

   $$\text{if } (a, b) \in T \text{ and } (b, c) \in T, \text{ then } (a, c) \in T.$$

   A relation T is called a transitive closure of a relation R on a set A if it is the smallest relation (on A) which is a superset of R and is transitive. In other words, (a, b) is in T iff there exist $b_1, b_2, \dots, b_k$ such that, $a = b_1, b = b_k$, and $(b_1, b_2), (b_2, b_3), (b_3, b_4), \dots, (b_{k-1}, b_k)$ are all in R (here k may be equal to 2).

   Give a dynamic programming algorithm to compute transitive closure T of a relation R, given the relation R as a matrix.

5. (a) A Chomsky Normal Form grammar G is of the form $G = (\Sigma, V, S, \delta)$, where Σ is the alphabet set, V is a set of non-termimals (where $V \cap \Sigma = \emptyset$), $S \in V$ is a starting symbol and δ is a set of productions of the form:

   $$A \rightarrow BC \text{ or } A \rightarrow a, \text{ where}$$

   $A, B, C \in V$ and $a \in \Sigma$ is a terminal.

   In the following α, β, γ, w are strings in $(\Sigma \cup V)^*$.

(b) We say that $\alpha A \beta \Rightarrow_G \alpha w \beta$, where $A \rightarrow w$ is a production in G (that is member of δ).

(c) We say that $\alpha \Rightarrow_G^* \beta$ if one of the following holds: (i) $\alpha = \beta$, (ii) $\alpha \Rightarrow_G \beta$ or (iii) for some γ, $\alpha \Rightarrow_G \gamma$ and $\gamma \Rightarrow_G^* \beta$.

(d) We say that $L(G) = \{w : w \in \Sigma^* \text{ and } S \Rightarrow_G^* w\}$.

Given a Chomsky Normal Form grammar G, give a dynamic programming algorithm to determine if a string $w \in \Sigma^*$ is a member of $L(G)$.