1. (30 points) Let $G = (V, E)$ be a directed graph.
   (a) Assuming that G is represented by an adjacency matrix $A[1..n, 1..n]$, give a $\Theta(n^2)$-time algorithm to compute the adjacency list representation of G. (Represent the addition of an element v to a list l using pseudocode by $l \leftarrow l \cup \{v\}$.)
   (b) Assuming that G is represented by an adjacency list $\text{Adj}[1..n]$, give a $\Theta(n^2)$-time algorithm to compute the adjacency matrix of G.

2. (70 points) We would like count the number the connected components in an undirected graph, and assign to each vertex the number (or name) of its connected component. For each vertex, put its component number into the array CompNum[1, ..., n] (where n is the number of vertices). You will give three algorithms for this problem. You do not have to justify their running times.
   (a) Give a linear time algorithm for this connected components problem using depth-first-search. This should look like the algorithm covered in class.
   (b) Give a linear time algorithm for this connected components problem using breadth-first-search.
   (c) Give a linear time algorithm for this connected components problem using depth-first-search without recursion. Use a stack instead.