605.202 Data Structures and Algorithms Lab 3

This assignment repeats Lab 2. Instead of using an array to represent the adjacency matrix, you are to use the adjacency matrix to build the graph using a linked allocation.

The recursion is not the most important part of this assignment. You may choose to find the path iteratively. In any event, compare your solution for Lab 3 to your recursive solution of Lab 2. Also, compare the implementations.

Remember, you are responsible for showing that your program does all that it is supposed to do. If you put in any special features, you need to have an I/O set which demonstrates the features You will lose points for having insufficient I/O.

Your program should use the same I/O format as Lab2 and the same required input file. Make sure to allow for reasonable error situations. Your code should generate all the possible start-node end-node pairs for each graph. It is okay for the start and end nodes to be the same.

Make up some other graphs as additional input. You will lose points for having insufficient I/O.

Remember, you are responsible for showing that your program does all that it is supposed to do. If you put in any special features, you need to have an I/O set which demonstrates the features

Your program should read an adjacency matrix from the file, build the graph using linked structures, and process the graph to find all the paths, then read an new adjacency matrix, etc., until the file is empty. Make sure to allow for reasonable error situations. Programs which use the adjacency nmatrix to find the paths will receive no credit under correctness.

You may not use Array lists or List classes from the Java library.

You may store the adjacency matrix in any way you like. You may use any variation of linked strucutres to represent the graph. Justify your implementation and design decisions.