TUTORIAL SESSION 4
GRAPHS

Question 1.

Given the graph in Figure 1, complete the following tasks:

a. Find
   a1. All noncyclic paths from A to D
   a2. All noncyclic paths from B to H
   a3. All noncyclic paths from E to C

b. Give the adjacency matrix representation of the graph.

c. Give the adjacency list representation of the graph.

d. Give the depth-first traversal of the graph (supposed we start from A).

e. Give the breadth-first traversal of the graph (supposed we start from A).

Question 2.

The un-directed version of the graph in Figure 1 is presented in Figure 2. For that un-directed graph (Figure 2), complete the tasks (from a. to e.) in question 1 again.

Question 3.

Given the graph in Figure 3, complete the following tasks:

a. Give the adjacency matrix representation of the graph.

b. Give the adjacency list representation of the graph.

c. Find the shortest path between node B and all other nodes in the above graph.

Question 4.

a. Given the graph in Figure 4, using the implementation of topological order in Appendix A to generate the topological list.

b. Write pseudocode for the algorithm given in Appendix A.

Question 5.

A graph can be used to solve the maze problem (as represent in Figure 5). Every start point, dead end, goal, and decision point can be represented by a node. The arts
between the nodes represent one possible path through the maze. For example, a path \{A,B,E,F,H,J,L,O\} is a solution of the next maze. Assume that, there is only one start point and one goal.

a. Write a pseudocode to search for a path through the maze (a solution) using depth-first traversal. Print out the solution.

b. Write a pseudocode to search for a path through the maze (a solution) using breadth-first traversal. Print out the solution.

c. Manually execute your code in a. and b. for the graph in Figure 5. What are the results?

d. (somewhat advanced) Write a pseudocode to simulate a mouse’s movement through the maze using a graph and a depth-first traversal. When the program completes, print the solution (a path through the maze).

For example, the simulation may be:

A >> B >> C >> D >> back to C >> G >> F >> H >> I >> back to H >> J >> K >> back to J >> L >> M >> back to L >> N >> back to L >> O

And the path is printed as: A, B, C, G, F, H, J, L, O

**Hint:**
- Modify the depth-first traversal algorithm to put two nodes to the stack at each step: the current node and its child. When a node is retrieved from the stack, if its attribute “processed” is 2, the status is “back to”.
- Use another stack to keep the current path. Remember to remove the “back to” steps.

**Question 6.**

a. Convert the map of cities in Figure 6 to a graph (redraw it as a graph).

**Hint:** make a table that maps a city to a vertex

b. Represent the map in an adjacency list

c. Represent the map in an adjacency matrix

d. Write a pseudocode of an interactive program that given the start and destination will display the shortest route between them.
Appendix A – An implementation of topological sorting

Given a directed graph $G$ whose vertex set $V = \{v_1, v_2, \ldots, v_n\}$. The topological order list $L$ on $V$ can be found applying the following steps:

Initially, make $L$ empty
Construct a vector $\text{deg} = \{d_1, d_2, \ldots, d_n\}$ where $d_i$ is the indegree of $v_i$.
Repeat the following tasks until all of vertices in $V$ are put into $L$:
- Find the largest $i$ such that $d_i = 0$ and $v_i$ has not been put into $L$
- Put $v_i$ into $L$
- For each $v_j$ such that $v_j$ is adjacent to $v_i$ and $v_j$ has not been put in $L$, make $d_j = d_j - 1$

Example 1: Consider the graph in Figure 7, we have $V = \{0, 1, 2, 3, 4\}$, the initial $\text{deg} = \{0, 4, 1, 1, 0\}$ and $L = ()$.

At the beginning, we have $d_0 = 0$ and $d_4 = 0$, in the meantime neither 0 nor 4 have been put in $L$. Thus we choose $i = 4$ (the largest) and put 4 into $L$. Thus, $L$ becomes (4). Since 1 and 2 are adjacent to 4, we decrease the value of $d_1$ and $d_2$ accordingly. Therefore, $\text{deg}$ becomes $\{0, 3, 0, 1, 0\}$.

Similarly, in the next step we choose $i = 2$ and put i into $L$. $L$ becomes (4,2) and $\text{deg} = \{0, 2, 0, 1, 0\}$. Next, we choose $i = 0$. $L$ becomes (4,2,0) and $\text{deg} = \{0, 1, 0, 0, 0\}$. Next, we choose $i = 3$. $L$ becomes (4,2,0,3) and $\text{deg} = \{0, 1, 0, 0, 0\}$. The last one to be put into $L$ is 1, and the final topological list $L$ to be found is (4,2,0,3,1).