Cmput 204, Winter 2016 **Assignment 4** **Due: Apr 1, before class**

- Acknowledge all sources and collaborations. Give an acknowledgement statement, even if you just say: "I did not collaborate with anyone". If you do not give an acknowledgement statement, your assignment may not be graded.
  Exceptions: you do not need to list course resources such as textbook, slides, seminar notes.
- Write clearly or print. If your TA cannot read your assignment, it may not be fully graded.
- Marks: the total weight of this assignment is 7% of this course.
  Marks per question: Question 1 - 20 marks, question 2 - 20 marks, question 3 - 16 marks, question 4 - 14 marks plus possible bonus marks.

1. Tracing depth-first search (dfs) on an undirected graph:

   Below are some CMPUT course offered by our department, and their (slightly simplified) prerequisites.

   (a) 174: (none)
   (b) 175: 174
   (c) 201: 175
   (d) 204: 175, 272
   (e) 272: 174
   (f) 304: 204
   (g) 325: 201, 204
   (h) 366: 204
   (i) 466: 366

   Now consider the graph G_u, where nodes are labeled by course numbers, and there is an **undirected edge** between a and b if course a is a prerequisite for b.

   Here are the problems you have to solve:

   (a) Write the adjacency list representation for G_u, where nodes are sorted in increasing numerical order. both within the top-level array and within each adjacency list.
   (b) Draw the adjacency matrix of G_u, again with nodes sorted in increasing numerical order.
   (c) Run dfs on G_u with the adjacency list representation. Label nodes with pre- and postvisit timestamps.
   (d) How many connected components are there?
   (e) List the degree of each node. Which is the most important course, as measured by its degree?

2. Directed graph and strongly connected components: Now consider the graph G, where nodes are labeled by course numbers, and there is a **directed edge** from a to b if course a is a prerequisite for b.

   (a) Write the adjacency list representation for G. Same format as for the previous question.
   (b) Is G a tree? Answer yes or no, and justify your answer.
   (c) Is G a DAG? Answer yes or no, and justify your answer.

(d) Which nodes of G are sources, and which are sinks?

(e) Which is the most important course, as measured by how many other nodes can be reached by dfs `explore` starting from that node?

(f) Compute the strongly connected components (SCC) of G using Kosaraju's algorithm as discussed in class (and in Chapter 3.4.2 of the textbook). Show the high-level steps, in similar level of detail as in the example in the book.

(g) Would there have been a simpler way to determine the SCC of this graph? If yes, explain it. (Note that you should do the algorithm trace in the previous step, even if you do find a simpler way here.)

3. Runtime of dfs: in class we stated (and gave some arguments for it in a non-formal way) that the runtime of dfs on a graph $G = (V, E)$ is $\Theta(|V| + |E|)$, when using an efficient implementation for `visited`.

   This question asks you to show formally that in some sense this is the best possible runtime we can hope for, for general graphs.

   (a) Show that $\Theta(|V|)$ is not a valid bound: As a function of v, construct a series of graphs with $|V| = v$ nodes such that the runtime of dfs on those graphs increases at a rate that is faster than $\Theta(v)$.

   (b) Show that $\Theta(|E|)$ is not a valid bound: As a function of e, construct a series of graphs with $|E| = e$ edges such that the runtime of dfs on those graphs increases at a rate that is faster than $\Theta(e)$.

4. Node expansion order of dfs versus bfs: For which graphs and which representations of graphs is the node expansion order of dfs exactly the same as for bfs? Develop as many necessary and/or sufficient conditions as you can think of. For each condition you give, describe whether it is necessary or sufficient, and give an example of such a graph.

   Necessary condition: if the condition is false, then the expansion order will certainly not be the same. But just because the condition is true for a graph G does not yet prove that dfs and bfs will expand the nodes of G in the same order.

   Sufficient condition: if the condition is true, then the expansion order will be the same. But there could be other graphs for which the condition is false, but the expansion order is still the same.

   If we have a condition that is both necessary and sufficient, then we have completely solved the problem. This is an open-ended question and I do not expect (or know) a complete answer. Just do the best you can.

   Example: If all the nodes in an undirected graph are in a "straight line", $V = \{v_1, \ldots, v_n\}$ and $E = \{v_i, v_{i+1}\}$ for $1 \leq i < n$, and exploration starts at v_1, then both bfs and dfs will visit the nodes in order from v_1 to v_n. So this is an example of a sufficient condition. In all such graphs dfs and bfs will explore nodes in the same order when given the same starting node. But it is not a necessary condition - there are other graphs which are not "straight line" but the exploration order of both algorithms is still the same.

   Hint: one type of special cases that you can look at is when the graph is a tree.

   (End of Assignment 4)