COMPSCI330 Design and Analysis of Algorithms
Assignment 4

Due Date: Thursday, February 28, 2019

Guidelines

- **Describing Algorithms** If you are asked to provide an algorithm, you should clearly define each step of the procedure, establish its correctness, and then analyze its overall running time. There is no need to write pseudo-code; an unambiguous description of your algorithm in plain text will suffice. If the running time of your algorithm is worse than the suggested running time, you might receive partial credits.

- **Typesetting and Submission** Please submit the problems to GradeScope. You will be asked to **label your solution for individual problems**. Failing to label your solution can cost you 5% of the total points (3 points out of 60 for this homework).

- LaTeX is preferred, but answers typed with other software and converted to pdf is also accepted. Please make sure you submit to the correct problem, and your file can be opened by standard pdf reader. **Handwritten answers or pdf files that cannot be opened will not be graded.**

- **Timing** Please start early. The problems are difficult and they can take hours to solve. The time you spend on finding the proof can be much longer than the time to write. If you need more time for your homework please use this form and submit a STINF.

- **Collaboration Policy** Please check this page for the collaboration policy. You are **not** allowed to discuss homework problems in groups of more than 3 students. **Failure to adhere to these guidelines will be promptly reported to the relevant authority without exception.**

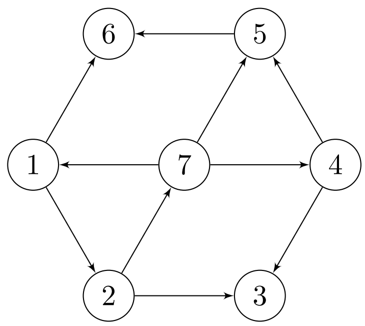
Problem 1 (Depth First Search)**.** (15 points) Given the following graph (Figure **??**), you are going to perform a Depth First Search.

When you have the option of choosing multiple vertices (in the main loop or when choosing the next vertex inside the recursive call), always choose the vertices with smaller indices first. That is, think of the main loop as

```
FOR i = 1 to 7
    DFS_visit(i)
```

For enumerating the edges, think of

```
For v = 1 to 7
    IF (u,v) is an edge and v is not visited THEN
        DFS_visit(v)
```



(a) (5 points) List the edges of DFS tree.

(b) (5 points) List the pre-order and post-order.

(c) (5 points) For all the non-tree edges, classify them into forward edges, back edges and cross edges.

Problem 2 (Super Mario)**.** (20 points) In the new Super Mario game, your goal is to lead Mario through a maze. The maze has n locations (labeled as $1, 2, ..., n$), where 1 is the starting point and n the the destination. There are m (directed) edges between the locations. Mario has two states - small or big. Edges will be of one of three types: only small Mario can go through edges of type 1; only big Mario can go through edges of type 2; both small and big Mario can go through edges of type 3. Locations also have three types: type 1 locations do not change the state of Mario; type 2 locations contain mushrooms so whenever Mario gets to these locations he will become big (no matter what the previous state was); type 3 locations contain enemies so whenever Mario gets to these locations he will become small (no matter what the previous state was). You are given a map of the maze that contains the types of all locations and all edges.

(a) (12 points) Construct a new graph, such that every path in the new graph correspond to a valid path in the maze, and every valid path in the maze corresponds to a path in the new graph. Prove this property.

(b) (8 points) Design an algorithm that finds a path from the starting location 1 to the ending location n. Mario starts small. Analyze the running time of your algorithm.

Problem 3 (2-Coloring)**.** (25 points) In this problem we are given an undirected graph G. The goal is to color every vertex in the graph G with red or blue, such that no edge in the graph connects two vertices of the same color.

(a) (5 points) A cycle in the graph is a sequence of vertices $(v_1, ..., v_k)$ such that (v_i, v_{i+1}) are edges for $i = 1, 2, ..., k-1$, and (v_k, v_1) is also an edge. Prove that if the graph contains a cycle whose length k is odd, then it is impossible to color the graph using only two colors while ensuring no edge connects two vertices of the same color.

(b) (8 points) Design an algorithm that produces a valid coloring of the graph if one exists; otherwise output "Impossible". Analyze the running time of the algorithm.

(c) (12 points) Prove the correctness for the algorithm you designed in (b). (Hint: When your algorithm outputs "Impossible", find a cycle of odd length in the graph.)