CS 2420-003 Algorithms and Data Structures

Fall Semester, 2018

Assignment 8: Graph Algorithms I

Due Date: 4:30 p.m., Monday, Nov. 26, 2018 (at the beginning of CS 2420 class)

(**Note:** This assignment has four writing exercises and two programming exercises. Please start early because this assignment might be more time-consuming than the previous ones.)

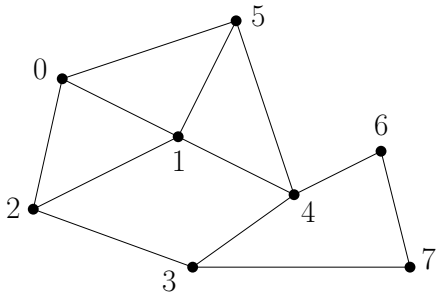

Figure 1: An undirected graph: the numbers besides the nodes are the indices of the nodes.

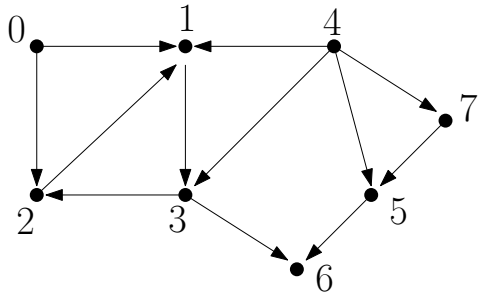

Figure 2: A directed graph: the numbers besides the nodes are the indices of the nodes.

1. Consider the graph in Figure 1. Please give the adjacency lists of the graph. For each list, please order the vertices in the ascending order of their indices. For example, the adjacency list of vertex 0 should be $1 \rightarrow 2 \rightarrow 5$. **(5 points)**

2. Consider the graph in Figure 2. Please give the adjacency lists of the graph. For each list, please order the vertices in the ascending order of their indices. **(5 points)**

3. Apply the BFS traversal algorithm on the graph in Figure 1 with vertex 0 as the starting vertex and using the adjacency lists in Question 1 (i.e., the vertices in each adjacent list are in ascending order of their indices). **(15 points)**

   (a) Please give the BFS traversal order of the vertices that are visited by the algorithm.

   (b) Please give the BFS tree (which is also the shortest path tree) produced by the algorithm.

4. Apply the DFS traversal algorithm on the graph in Figure 2 using the adjacency lists in Question 2 (i.e., the vertices in each adjacent list are in ascending order of their indices). Your algorithm should start from vertex 0. Since this is a directed graph, the DFS traversal from vertex 0 may not be able to reach all vertices. Hence, as discussed in class, after the DFS traversal from 0, your algorithm should check other vertices in their index order, and if there is an unvisited vertex, then start a new traversal from that vertex. **(15 points)**

   (a) Please give the DFS traversal order of the vertices that are visited by the algorithm.

   (b) Please give the DFS tree/forest generated by the algorithm. Again, if the DFS traversal from vertex 0 cannot reach all vertices, then the algorithm will generate a forest (i.e., multiple trees) rather than a single tree.

5. In this exercise, we will implement BFS algorithm. **(25 points)**

   On Canvas, go to the following directory: homework/hw8/question5. There are a starter java file "hw8_Q5.java" and an input file "hw8_Q5_input.txt".

   In the input file, the first line is the number of vertices of the input graph, followed by an adjacency matrix of the input graph. The input graph is actually the one in Figure 1.

   The program first reads the graph information from the input file. The adjacency matrix will be stored in an array M, and then the program will construct the adjacent lists from the matrix. A function $printAdjLists()$ is provided for you to print out all adjacency lists.

   I have provided lots of comments to explain my starter code. I suggest you read all of them.

   With the adjacent lists, you task is to implement the BFS traversal, starting from the vertex 0. Unlike the previous assignments where I normally gave you the functions and you only needed to fill in the functions with your code, this time you will need to provide your own functions as well. Your program should output the following information to the screen. First, output the BFS traversal order of the vertices that are visited by the algorithm. Second, as discussed in class, the BFS algorithm can compute shortest paths from the source vertex (i.e., vertex 0) to all other vertices by using the BFS tree (i.e., the information stored in pre); for each vertex i with $i \neq 0$, you are required to output a shortest path from vertex 0 to vertex i.

   There is a file "solution_hw8_Q5_output.txt" in the same directory, which contains the correct output. You may design your own output following the similar style.

   When implementing your BFS algorithm, you may use the same approach as we discussed in class. I already define the arrays $color$ and pre in the starter file for you to use. Alternatively, you may also use a different way of implementation, in which case please provide sufficient comments. In either case, please make your output format follow the similar style as in the file "solution_hw8_Q5_output.txt".

6. In this exercise, we will implement DFS algorithm. **(25 points)**

   On Canvas, go to the following directory: homework/hw8/question6. There are a starter java file "hw8_Q6.java" and an input file "hw8_Q6_input.txt". The input file now contains the adjacency matrix of the graph in Figure 2. As in Question 5, the program first reads the graph information from the input file and then construct the adjacency lists. Some parts of the code are similar to those in my starter file for Question 5, so I omitted the comments.

   You task is to implement the DFS algorithm and output the DFS traversal vertex list to the screen. You will also need to provide your own functions. You should start from vertex 0. Again, since this is a directed graph, the DFS traversal from vertex 0 may not be able to reach all vertices. Hence, as discussed in class, after the traversal from 0, your algorithm should check other vertices in their index order, and if there is an unvisited vertex, then start a new traversal from that vertex.

   There is a file "solution_hw8_Q6_output.txt" in the directory, which contains the correct output. You may design your output following the similar style.

Total Points: 90