CS 2420-001 Algorithms and Data Structures

Fall Semester, 2018

Assignment 9: Graph Algorithms II

Due Date: 1:30 p.m., Wednesday, Dec. 5, 2018 (at the beginning of CS 2420 class)

(**Note:** This assignment has two programming exercises. Please do not miss the note at the end of this document, which contains important information that can help you finish this assignment.)

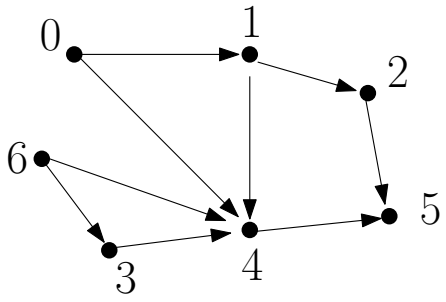

Figure 1: A directed acyclic graph (DAG) in the input file for Question 1.

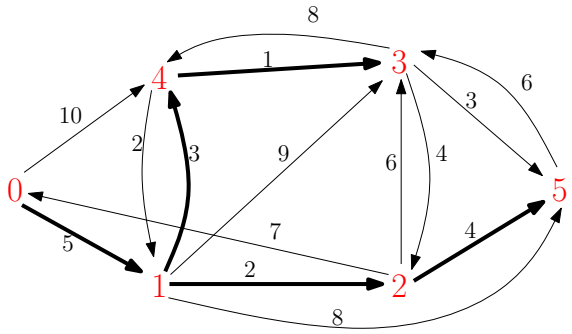

Figure 2: A weighted directed graph for Question 2: the edges have weights and the thick edges form a shortest path tree from vertex 0.

1. In this exercise, we will implement the topological sort algorithm. **(25 points)**

   On Canvas, go to the following folder: homework/hw9/question1. There are a starter java file "hw9_Q1.java" and an input file "hw9_Q1_input.txt".

   Like in the last assignment (Assignment 8), in the input file, the first line is the number of vertices of the input graph, followed by an adjacency matrix of the graph, which is the one in Figure 1. The graph is a directed acyclic graph (DAG). 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.

   You task is to complete the method topSort() and output a topological order of the vertices of the graph. The file "solution_hw1_Q1_output.txt" gives a correct output. Since the topological order may not be unique, you may give different answers depending on how you implement the algorithm.

2. In this exercise, we will implement Dijkstra's shortest path algorithm. **(25 points)**

   On Canvas, go to the following directory: homework/hw9/question2. There are a starter java file "hw9_Q2.java" and an input file "hw9_Q2_input.txt".

Here the input file is not the same as before because the graph edges have weights. The number in the first line is still the number of vertices. For the adjacency matrix, for each element $M[i, j]$, if it is zero, then it means that there is no edge from i to j; otherwise there is an edge and its weight is $M[i, j]$. The graph in the input file is the one in Figure 2.

When the program constructs the adjacent lists, the weights of the edges will also be stored in the adjacency lists. For this, as discussed in class, there is a data field "weight" in the Vertex class.

You task is to complete the function Dijkstra(). You should use **vertex 0** as the source vertex. Essentially the function Dijkstra() is used to compute the predecessor array *pre* and the shortest path distance array *dis*. Both arrays have been defined in the constructor of the Graph class. After you compute the two arrays in the function Dijkstra(), you can use the two methods *printSP*() and *getSPdis*() that have already been provided to print the shortest paths from the source vertex 0 to all other vertices and their shortest path distances. The file "solution_hw9_Q2_output.txt" contains the correct output.

As we discussed in class, there are two ways to implement the priority queue Q in Dijkstra's algorithm, an array or a heap. Although a heap is preferred since it is generally more efficient, you may also use an array, which is much easier to write the program. You may feel free to use it if Java already has a priority queue data structure in its standard libraries (which I am not sure).

Note: One the one hand, topological sort and Dijkstra's algorithm are important topics, and I believe it is better for you to have a chance to implement them. On the other hand, I understand that it is the end of the semester and you may be quite busy. Therefore, I would like to do some extra work to help you. Here is the plan. You go ahead and work on the assignment. In order to help you, I will upload my solution code on Canvas by 6:00 p.m., Monday (Dec. 3). You can take a look at my code and then finish yours.

However, I do encourage you to finish the assignment without looking at my code. Therefore, you will receive **5 bonus points** if you can submit your code (for both questions in this assignment) by 1:30 p.m., Monday, Dec. 3 (Canvas will keep a record on your submission time). (Note that in case you make a submission but later make changes to your submission, then you will not receive any bonus points if the changes happen after the above time.)

Total Points: 50 (not including the 5 bonus points)