Optional Lab Specification – A practical exercise to implement Graph data structure and graph-based algorithms.
50 points

Lab Goals

- Do multiple exercises on Graph data structure.

Suggestions for Success

- Take a look at the suggestions for successfully completing the lab assignment, which is available at: `https://www.cs.allegheny.edu/sites/amohan/resources/suggestions.pdf`

Learning Assignment

If you have not done so already, please read all of the relevant "GitHub Guides", available at the following website:
`https://guides.github.com/`
that explains how to use many of the features that GitHub provides. This reading will help you to understand how to use both GitHub and GitHub Classroom. To do well on this assignment, you should also read

- **Sedgewick ch4: 4.1 & 4.2**

Assignment Details

Now that we have completed discussing important concepts related to graph data structure and graph-based algorithms in the last few lectures, it is your turn. In this lab, you will practice a variety of problems to retain the knowledge of Graph-based algorithms. This includes solving one or more problems and analyzing algorithms.

This lab is primarily intended to be done remotely and completely optional. The score of this lab will be switched with any of your low lab scores.

Please make sure to push your code file(s) and commit by the deadline, April 17th, 2020, morning 8:00 AM.

It is required for all students to follow the honor code. Some important points from the class honor code are outlined below for your reference:

1. Students are not allowed to share any solution files and/or other implementation details. It is acceptable to have a healthy discussion with your peers. However, this discussion should be limited to sharing ideas only.
2. Submitting a copy of the other's solution is strictly not allowed. Please note that all work done during lab sessions will be an opportunity for students to learn, practice, and master the materials taught in this course. By doing the work individually, students maximize the learning and increase the chances to do well in other assessments such as skill test, exams, etc $\cdots$

At any duration during and/or after the remote lab session, students are recommended to team up with the Professor and the TL's to clarify if there is any confusion related to the lab and/or class materials. The low-details of

implementation is purposefully left out from this specification, to encourage students to think creatively and explore the details of implementation themselves.

Section 1 - Graph Implementation (20 points)

- **ADD** the following statement to all your solution files including the file named `GraphDriver.java`.

  This work is mine unless otherwise cited - Student Name

- In this section, you are required to implement the Graph data structure using the adjacency list format. We learned in our remote classes that, a Graph may be represented using adjacency matrix or adjacency list.
- A sample Linked List implementation is provided in the lab repository, along with some basic examples to add, remove and get items from the linked list. The example code `LinksStub.java` file and other code related to Linked List should be familiar from earlier classes such as CS-101.
- The implementation of your Graph data structure should be done in the `GraphDriver.java` file.
- Please access the lecture slides in lesson 14 to review our discussion on Graphs and Adjacency Lists.

Section 2 - Graph Traversal - DFS (15 points)

- **ADD** the following statement to all your solution files including the file named `DFS.java`.

  This work is mine unless otherwise cited - Student Name

- In this section, you are required to implement the DFS Graph Traversal algorithm using the Graph represented through the adjacency list format implemented from the previous part. We learned in our remote classes how to perform the DFS Graph traversal and the pseudocode for the algorithm is available in the slides.
- A sample Linked List implementation is provided in the lab repository, along with some basic examples to add, remove and get items from the linked list. The example code `LinksStub.java` file and other code related to Linked List should be familiar from earlier classes such as CS-101.
- The implementation of your Graph data structure should be done in the `DFS.java` file.
- Please access the lecture slides in lesson 14 to review our discussion on Graphs Traversals and Adjacency Lists.

Section 3 - Graph Traversal - BFS (15 points)

- **ADD** the following statement to all your solution files including the file named `BFS.java`.

  This work is mine unless otherwise cited - Student Name

- In this section, you are required to implement the BFS Graph Traversal algorithm using the Graph represented through the adjacency list format implemented from Section 1. We learned in our remote classes how to perform the BFS Graph traversal and the pseudocode for the algorithm is available in the slides.
- A sample Linked List implementation is provided in the lab repository, along with some basic examples to add, remove and get items from the linked list. The example code `LinksStub.java` file and other code related to Linked List should be familiar from earlier classes such as CS-101.
- The implementation of your Graph data structure should be done in the `BFS.java` file.
- Please access the lecture slides in lesson 14 to review our discussion on Graphs Traversals and Adjacency Lists.

Submission Details

For this assignment, please submit the following to your GitHub repository before the **deadline** by using the link shared to you by the Professor:

1. Submission Files, namely, the “GraphDriver”, “DFS”, and “BFS”.
2. It is highly important, for you to meet the honor code standards provided by the college. The honor code policy can be accessed through the course syllabus. Make sure to add the statement ”This work is mine, unless otherwise cited.” in all your deliverables such as markdown and PDF files.

Grading Rubric

1. There will be full points awarded for the lab if all the requirements in the lab specification are correctly implemented. Partial credits will be awarded if deemed appropriate.
2. Failure to upload the lab assignment code to your git repo will lead you to receive no points given for the lab submission. In this case, there is no solid base to grade the work.
3. There will be no partial credit awarded if your code doesn’t compile correctly. It is highly recommended to validate if the correct version of the code is being submitted before the due date and make sure to follow the honor code policy described in the syllabus. If it is a late submission, then it is the student’s responsibility to let the professor know about it after the final submission in GitHub. In this way, an updated version of the student’s submission will be used for grading. If the student did not communicate about the late submission, then automatically, the most updated version before the submission deadline will be used for grading purposes. If the student had not submitted any code, then, in this case, there are no points awarded to the student.
4. If you need any clarification on your lab grade, talk to the Professor. The lab grade may be changed if deemed appropriate.