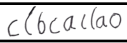




**CCDSALG Term 3, AY 2023 – 2024**  
**Project 2 Documentation – Social Network (Undirected Graph Application)**

**DECLARATION OF INTELLECTUAL HONESTY / ORIGINAL WORK**

We declare that the project that we are submitting is the product of our own work. No part of our work was copied from any source, and that no part was shared with another person outside of our group. We also declare that each member contributed to the project as indicated in the table below.

Section	Names and Signatures	Task 1	Task 2	Task 3	Task 4	Task 5
S18	Cailao, Carlos Luis B. 	X	X	X	X	X
	Castillo, Marvien Angel C. 	X	X	X	X	X
	Sena, Sophia Pauline V. 	X	X	X	X	X

Fill-up the table. For the tasks, put an X mark if you have performed the specified task. Please refer to the project specifications for the description of each task. Don't forget to affix your signature.

1. Indicate how to compile (if it is a compiled language) your codes, and how RUN (execute) your program from the COMMAND LINE. Examples are shown below highlighted in yellow. Replace them accordingly. **Make sure that all your group members test what you typed below because I will follow them verbatim. I will initially test your solution using the sample input text file that you submitted. Thereafter, I will run it again using my own test data:**

- How to compile from the command line (for compiled language only):

```
C:\CCDSALG> gcc -Wall main.c -o main.exe
```

- How to run from the command line

```
C:\CCDSALG>main
```

2. How did you implement your Graph data structure? Did you implement it using adjacency matrix? adjacency list? Why? Explain briefly (at most 5 sentences).

- We implemented it using an adjacency list since linked lists felt more intuitive and efficient in terms of memory and time since you would only need to initialize the vertices once and store the adjacencies through an array of pointers, unlike the matrix wherein you would still need a 2D array and such.

5. How did you implement the DFS and BFS traversals? Explain your algorithm succinctly.

a. DFS Traversal:

- Initialize a stack of vertex pointers and push the start vertex. While the stack of vertex pointers is not empty, pop and visit the not yet visited vertex, sort the neighbors in increasing order, and push all not yet visited neighbors into the stack. The loop will repeat until all vertices are visited.

b. BFS Traversal:

- First, initialize a queue of vertex pointers. Then, enqueue the start vertex and mark it as visited. While the queue is not empty, dequeue a vertex from the queue, visit it, enqueue the yet to be visited neighbors in the order they are discovered, and repeat until all vertices are marked as visited. In the case of several neighbors to be visited from the current vertex, the neighbors are sorted in ascending sequence so the vertex with the lowest vertex ID is visited first.

6. Disclose what is NOT working correctly in your solution. Be honest about this. Explain briefly the reason why your group was not able to make it work.
- Due to time constraints, we were not able to dynamically allocate vertices based on the number of inputs. Alternatively, we have set the maximum number of vertices and edges to 100. Additionally, we have also set the maximum number of characters for a vertex ID to be 50.

Cailao, Carlos: This project was relatively easier than MCO1 since it does not deal with multiple input types, only strings, unlike MCO1, wherein multiple operation interactions had to be taken into account, aside from the queue and stack implementation.

Sena, Sophia Pauline: The project is more manageable and less stressful than the previous MCO. It's fairly easy in a way that we still get challenged with our knowledge of the algorithms somehow.

Castillo, Marvien Angel: The difficulty of this project is ok for me. It was also interesting because we got to experience using other tools (for example making the graph) and it was more fun compared to MCO1.

Castillo, Marvien Angel: The difficulty of this project is ok for me. It was also interesting because we got to experience using other tools (for example making the graph) and it was more fun compared to MCO1.

Request: If you know in advance that your solution to the bonus portion is not working correctly, then please DO NOT submit it so as not to waste precious time for both of us.

**Screenshot of the original graph** (corresponding to the data encoded in the sample input text file you submitted):

9. Fill-up the table below. Refer to the rubric in the project specs. It is suggested that you do first an individual self-assessment. Thereafter, compute the average evaluation for your group, and encode it below.

REQUIREMENT	AVE. OF SELF-ASSESSMENT
1. Correctly produced the list of vertices and their corresponding degrees	<u>15</u> (max. 15 points)
2. Correctly produced the BFS traversal	<u>35</u> (max. 35 points)

3. Correctly produced the DFS traversal	<u>35</u> (max. 35 points)
4. Documentation	<u>10</u> (max. 10 points)
5. Compliance (deduct 1 point for every instruction not complied with)	<u>5</u> (max. 5 points)
Bonus opportunity #1: Drawing of the given graph	<u>0</u> (0 or 10 points)
Bonus opportunity #2: Drawing of the BFS tree	<u>0</u> (0 or 10 points)
<p style="text-align: right;"><b>TOTAL SCORE</b>    <u>100</u> over 100.</p> <p style="color: red;">NOTE: The evaluation that the instructor will give is not necessarily going to be the same as what you indicated above. The self-assessment serves primarily as a guide.</p>	