TSYS School of Computer Science
CPSC 2018 – Data Structures
Fall 2022

Instructor: Dr. Abid

Project#3: Graphs

Due date: Monday, Nov 28th, before 11:59 am.
Graded: /100

Specifications:

- This project is meant to practice in-class acquired 'conceptual knowledge' about 'Graphs'.
- You will write a "Program" to manage a set of vehicles communicating between each other – alike in '*automotive cars'*. Every vehicle is equipped with an *'Antenna'* whose '*Connectivity Range*' is given as input (in meters).
- We assume that we know the exact position of every vehicle using GPS (Global Positioning System)

Implementation:

☛ Adopt the '***Adjacency List'*** implementation of Graphs.

- Build your graph by reading from a text file.

Input

- The file will have the following format:

```
5
250
25.11 50.5
20.23 50.5
50.23 100.11
100.00 -200.01
-5.05 -150.01
```

- The First line specifies the number of vehicles.
- The second line specifies the transmission range (in meters / a double).
- Every successive line contains the coordinates of the n Vehicle.
  Seeking simplicity, every vehicle/vertex will be assigned an incremental integer ID starting from 0.
  - ☛ We will use our own file to test your programs. So, use the same file naming *"data.txt*", and put the data file in the same directory as the .exe!

☛ ***Be careful when building the 'Adjacency List':***

NB ☛ The insertion (in the adjacency list (linked list/vector) of each vertex)) *should be done in an increasing order of the **"Distance"*** value between the vehicles, i.e., the first node in the adjacency list/queue is the closest among all adjacent nodes.

- We assume "Wireless Communication" is a Full-Duplex one (i.e., Undirected Graphs)

Output:

The program will display a menu with the following operations:

1. *Display All Edges*
   - *in the format (1, 2, 5) [Third parameter is the weight of the edge (i.e., distance)]*
2. *Display Adjacent Vehicles*
   - *for a given vehicle (by ID)*
3. *Move a Vehicle*
   - *Changing the Vehicle Coordinates (for a Vehicle given by ID)*
   - *Input: VehicleID, (new) Coordinates*
   - *The corresponding Adjacency List will change as a consequence!*
4. *DFS*
   - *from a given vehicle*
   - *Ask for a vehicle to start from, then display 'reachable' vertices*
5. *BFS*
   - *from a given vehicle*
   - *Ask for a vehicle to start from, then display 'reachable' edges*

   ☛ ***V. IMP**: The DFS and BFS traversals output should be unique, thanks to the "Constraint" on the in-order insertion of edges (based on distance value) in the Adjacency List.*
6. *MST*
   - *Prim Algorithm*
   - *Display MST Edges in increasing order of distance.*
7. *Shortest Path: (Optional/Bonus: +10 pts)*
   - *Ask for Start and Destination then*
   - *Display the Path: Edge by Edge*
   - *Display the Min Distance/Cost (Sum of distances)*
8. *Quit*

The Menu is a looped one and the only way to exit is to select 'Quit'

Deliverables

- *You have to record a 5-10 min demonstration with the following ingredients:*
  - *Briefly introducing your name and course/semester*
  - *Skimming over the code (explain major functionalities and Graph implementation)*

    - *Running the program – going through all specified functionalities (1-7) and explaining/highlighting any defect in case there are any.*
- *☛ Upload your recording to Youtube and share the link as part of your submission in CougarView (You don't need to upload the video: only share the youtube link – e.g., in a comment/text in the submission page). Make sure your recording is unlisted or public (unlisted means only the ones with a link can access it).*
- *.zip your project folder (in IntellijIDEA) – name it Stud#1LastName_Stud#2LastName_Stud#3LastName.zip and submit it (only one team member submits)*

☛ Good Will!

Lateness:

According to the Syllabus, a 24-hrs lateness induces a -15% (cumulative) penalty.

V. IMP

Remember ☛ PLAGIARISM will be SEVERLY PENALIZED!
