المملكة العربية السعودية
وزارة التعليم العالي
جامعة الملك عبد العزيز
كلية الحاسبات وتقنية المعلومات

Department of Computer Science

CPCS 324: Algorithms and Data Structures (II)

Spring 2023

Group Term Project – Part 2 (10%)

Issue Date: ***Saturday 20th May 2023***

Objective:

- Provide an opportunity to work with complex data structures
- Develop advanced programming skills.

Learning Outcomes:

- CLO #10: Design an algorithm/Pseudocode based on Greedy techniques for a computing problem. **(SO # 2)**

Project description:

Goal: Implement Dijkstra algorithm and using it to compute the all-pair-source shortest path problem.

Required Tasks (Total 10 points):

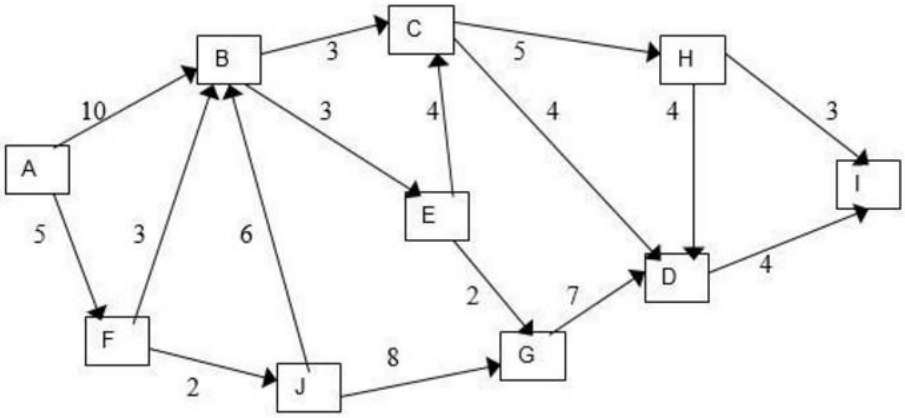
Due Date: ***Sunday 4th June 2023, 11:59 PM***

"Air freight is a system of transporting goods which can be commercial or non-commercial by aircraft. … Aircraft can travel for a certain number of kilometers to deliver freight from one location to another. … The problem, however, is an optimization problem to cut down the distances travelled minimizing the number of stops-over so as to optimize the time for delivery of time-sensitive goods and minimize the total cost ". [1] Implement the Dijkstra algorithm and develop an application that uses this algorithm to compute the shortest path between every pair of locations, so that an Air freight company can use it to speed up its delivery service and minimize fuel costs.

follow the design presented in ***Appendix I*** and implement it -*as instructed*- to do the following:

(Total 10 points)

Requirement 1: Consider the following directed weighted graph:



Required task:

1.1 Implement the Dijkstra algorithm to compute the length of the shortest paths from all locations to the rest of the locations. Make sure to print the shortest paths with their lengths.

- Store the above graph within a file and use readGraphFromFile() method to store it in memory

Required Output:

- print the shortest path with its length from the chosen vertex to every other vertex in the graph.
- Print the results in the format shown in the following figure:

The starting point location is A

The routes from location A to the rest of the locations are:

loc. A: city 5– loc. B: city 2 – loc. C: city 8 --- route length: 20

loc. A: city 5 – loc. B: city 2 – loc. C: : city 8– loc. D: : city 1 --- route length: 15

……..

The starting point location is B

The routes from location B to the rest of the locations are:

loc. B: city 2 – loc. C : city 8 --- route length: 10

……

Requirement 2: Define a function called make_graph() to randomly generate Graphs, assuming that the edge weights are positive integers, for the following cases:

- n=2000 m=10000
- n=3000 m=15000
- n=4000 m=20000
- n=5000 m=25000
- n=6000 m=30000

note: reuse the Graph methods developed in part 1 to generate the graphs specified above.

Required task:

2.1 Use the Dijkstra algorithm implemented in ***requirement 2*** to compute the length of the shortest paths for each pair of vertices and apply it on the above **five** randomly generated graphs. Compute the run duration and compare the algorithm's time efficiency computed from the experiment's results with the one determined by the formula stated in the textbook.

Required Output:

- print the *empirical* time efficiency for each of the seven experiments and compare them with corresponding expected theoretical time efficiency. Use the following Tables:

n	Running time of Algorithm	Expected theoretical time efficiency
5000		
10000		
Complete the rest of the results …		

	Ratio of running time	Ratio of expected time efficiency
Time of 5000/ Time of 1000		
Time of 10000/ Time of 5000		
Time of 15000/ Time of 10000		
Complete the rest of the results …		

note:

1- you need to make some simple straight-forward modification to the formula in the textbook in order to derive the theoretical order of growth – time efficiency.

Due Date: ***Sunday 4th June 2023, 11:59 PM***

Expectations

1. For each phase: Students are expected to submit *three* deliverables:
    - A source code for the implementation of the algorithm(s) uploaded in GitHub such that
        1. The code is clear and documented (commented)
        2. The functions are well structured.
    - A report (.docx) that includes the following:
        1. A small introduction (one paragraph)
        2. Clear screenshots for the outputs with an explanation for each figure
        3. Difficulties faced during the phase design
        4. Conclusion
2. A final oral presentation (with slides) for executing the project phases in pre-assigned date and time by the instructor

---

Instructions:

1. Each group must have only *four* students.
2. Use Java programming language.
3. Students are required to submit *a soft copy* of their work
4. Avoid *plagiarism* (Avoid copying work from your class-fellows).
5. Submit the Project by given deadline to avoid deduction of marks.
6. Only one member of the group should submit the project under its slot in the course Blackboard page.
7. Be Creative!!

Consequences / Penalties:

Plagiarism/Copying or Outsourcing will not be tolerated. **If a student is caught cheating, then the grade of the project for all students knowingly involved will be *Zero*.** Furthermore, based on the severity of the case, the entire course grade for the student may be lowered a full letter grade, an **"F"** in the course, dismissal from an academic unit, revocation of admission, suspension from the university, etc.

Resources/Recommended Software:

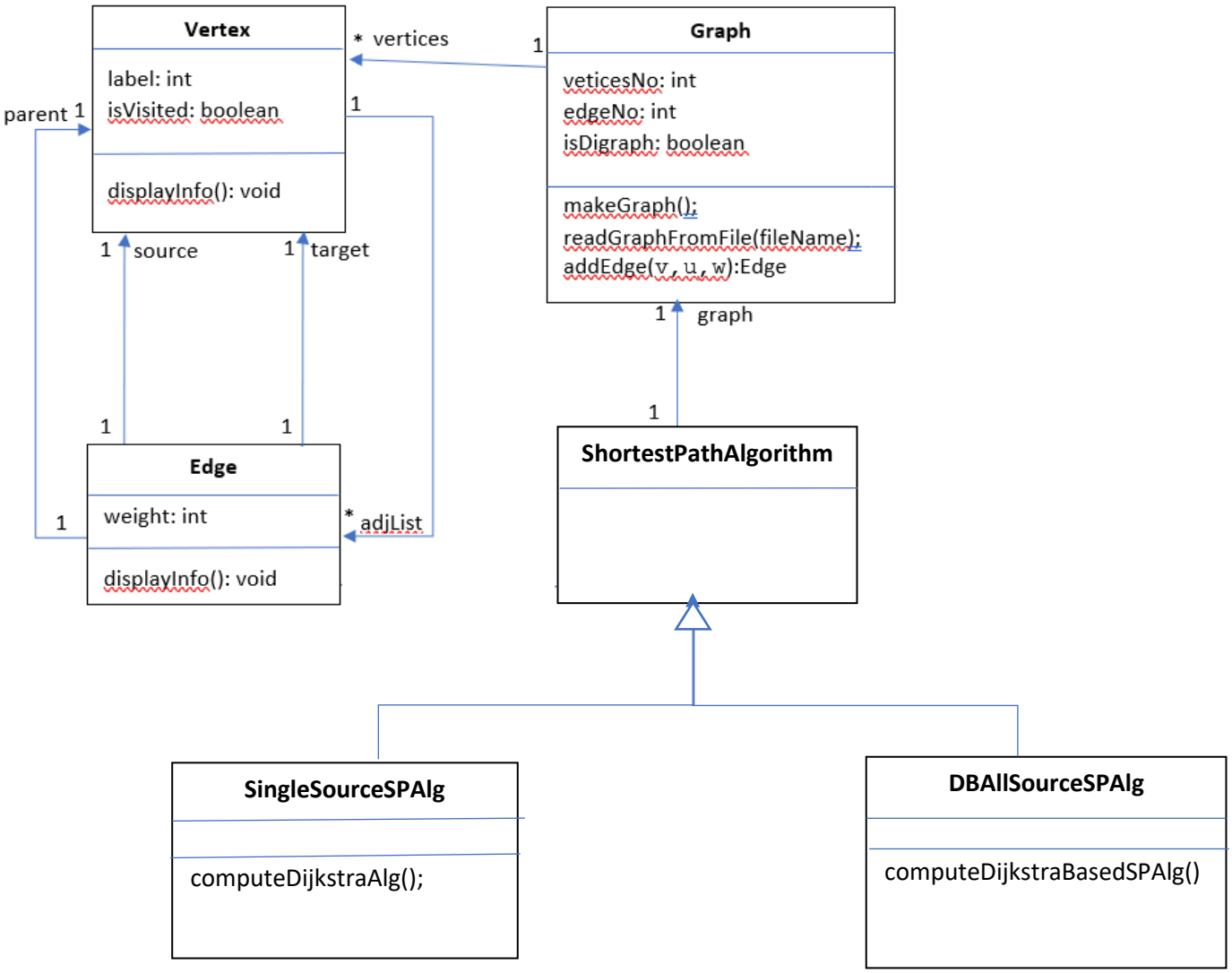
- Git
- Visual Studio Code (*recommended*: install Git first)

Course Textbook: Anany Levitin, Introduction to The Design and Analysis of Algorithms, Addison-Wesley (Pearson International Edition), 3rd edition, 2011. ISBN: 027376411X

Appendix I

Use the TWO Class diagrams presented below to implement the Air Freight Application. The relationship between the two parts of the system is shown in the following package diagram.

The following class diagram represents the Graph Framework:



Given the above diagram do the following:

1. Use the above classes, their attributes and member functions in your implementation.

PS. Marks will be **deducted** in case any of the elements of the class diagrams was not used as defined, or *extra* variables were used for the same purpose of those elements.

2. According to UML notation, the name on the side of an association should be used as a member variable of the class. For example, `parent` on the association between `Vertex` and `Edge` classes should be defined as a member variable within the Class `Edge`, the variable's type is `Vertex` and its name is `parent`.

3. Reuse all classes defined in Part I, Except for the class hierarchy responsible for implementing the algorithms for computing the minimum spanning tree for a graph.

4. The same constraint of not creating objects of Vertex, Edge, Location or Route, instead use the methods createVertex() and createEdge() that are defined in the Graph class.

4. The **parent** attribute of the Edge class **must** be used. It is not optional.

5. The parameters of the member method are left to the students to be defined/modified as appropriate.

Classes documentation:

==========

Classes that have the same names as the ones in the Graph framework in project-phase 1, are the same.

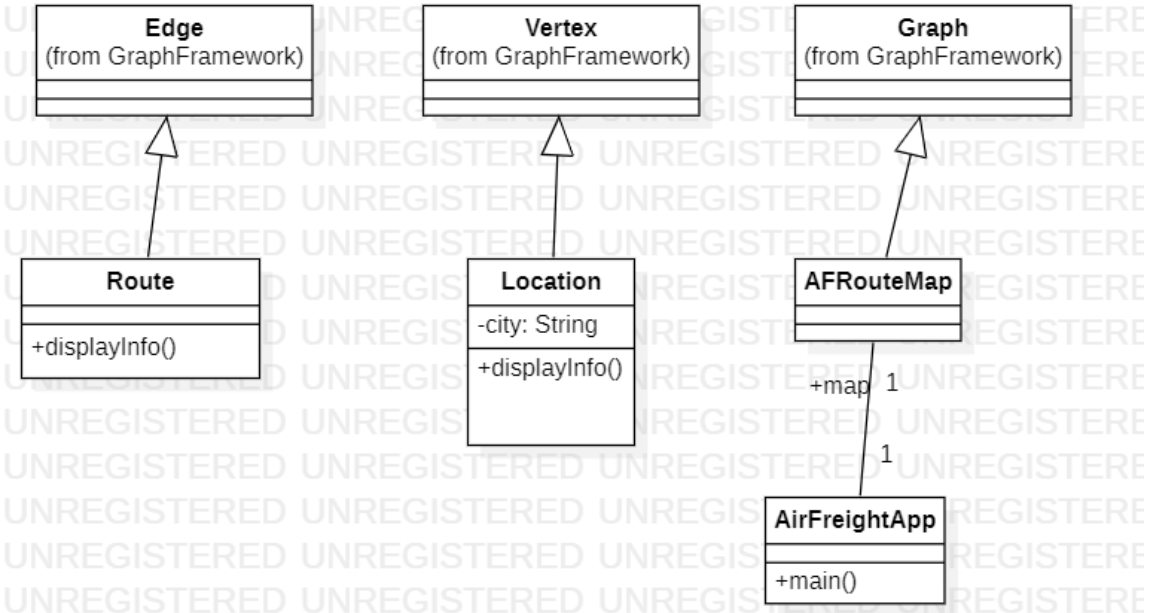
The ShortestPathAlgorithm super class has two subclasses.
Attributes displayed via association relationship in the diagram: `graph` attribute.

SingleSourceSPAlg class represents the implementation of the algorithm responsible for computing the shortest path for a specified source, which is located in the `computeDijkstraAlg()` method.

DBAllSourceSPAlg class represents the implementation of the Dijkstra-based shortest path algorithm for computing the shortest path from each vertex to the rest of the vertices, which is located in the `computeDijkstraBasedSPAlg()` method. The method will need to call `computeDijkstraAlg()` method.

Suggestion: make a design decision of what data structure you will be using to store the *value of the result of the Dijkstra algorithm and the Dijkstra-based shortest path algorithm*. Think of making use of the Edge class as the unit of your data structure Dijkstra-based shortest path algorithm.

The following class diagram shows the representation of the `AirFreightApp`:



`AirFreightApp`: is a class. It is the starting point of the program and contains the main method.

`main()`: this function should be responsible for running the readGraphFromFile method for requirement 1 and running the make graph function for requirement 2 to initialize the graph and invoking the Dijkstra -based all source shortest path algorithm and displaying the returned result and the measured running time.

Important: Make sure that you create objects of the `Graph`, `Route` and `Location` classes, ***do not make objects*** of Vertex and Edge. Any object creation in the AirFreightApp package should be of the form

```
Vertex v = new Location();      or     Location loc = new Location();
```

The same goes for Edge and route object creation.

`Location`: is a subclass of Vertex, it inherits all attributes, operations & relationships.

`Label the inherited` attribute should store Loc1, Loc2,… or Loc15 ect. – starts with Loc followed by a unique number.

Override/complete the `displayInfo()` method to display the information of the class attributes.

The output described previously:

C : city 8

Should use this method (not the Vertex method)

`Route`: it is a subclass of Edge, it inherits all attributes, operations & relationships.

Override/complete the `displayInfo()` method to display the information of the class attributes. You may or may not need to use (call) this method.

The output described previously:

--- route length: 10

Should use this method (not the Edge method).

[1] Tinuke Omolewa Oladele1 , Adekanmi Adeyinka Adegun2 , Roseline Oluwaseun Ogundokun3,* , Aderemi Elisha Okeyinka4 & Lucas Ayeni5, Application of Floyd-Warshall's Algorithm in Air Freight Service in Nigeria. **International Research Publication House** International Journal of Engineering Research and Technology. ISSN 0974-3154, Volume 12, Number 12 (2019), pp. 2529-2535

ijertv12n12_63.pdf (ripublication.com)

last accessed: 15-5-2023