COEN 352 Fall 2023 Assignment 3

This assignment practices graph data structure and associated algorithms. We aim to develop a simple application handling prerequisites of courses. We take the example of the description in the document here.

https://www.concordia.ca/academics/undergraduate/calendar/current/section-71-gina-cody-school-of-engineering-and-computer-science/section-71-30-department-of-electrical-and-computer-engineering/section-71-30-2-course-requirements-beng-in-computer-engineering-.html#19034

COEN 212 Digital Systems Design I (3.5 credits)

Prerequisite/Corequisite: The following course must be completed previously: MATH 204 (Cegep Mathematics 105).

Description: Modulo arithmetic: representations of numbers in binary, octal and hexadecimal formats; binary arithmetic. Boolean algebra; theorems and properties, functions, canonical and standard forms. Logic gates and their use in the realization of Boolean algebra statements; logic minimization, multiple output circuits. Designing with MSI and LSI chips, decoders, multiplexers, adders, multipliers, programmable logic devices. Introduction to sequential circuits; flip-flops. Completely specified sequential machines. Machine equivalence and minimization. Implementation of clock mode sequential circuits.

Component(s): Lecture 3 hours per week; Tutorial 2 hours per week; Laboratory 15 hours total

If we trace from a given core course, such as coen 212, we will find its prerequisite course is math 204. The core courses and the prerequisite contain the course sequences. In the graph, the node is a course. The prerequisite is the edge between two courses. By traversing such a graph, we can retrieve the prerequisite of a given course. This assignment is about developing a graph based application to

Problem 1(5 marks): Define a graph file named core_course.gph for all the core courses. The file format is shown the example source code released on Moodle site.

```
# Graph example for course sequences
5               # Number of vertices
D               # Directed weighted graph
0 1             # List of Edges; 0 is for Math204 and 1 is for COEN212
0 2             # 0 is for Math204 and 2 is for COEN231
0 3             # 0 is for Math204 and 3 is for COEN243
3 4             # 3 is for COEN243 and 4 is for COEN244

```

Problem 2 : Program a class CourseSequenceSolver that contains a graph of all the courses and their relations on prerequisite based on the file core_course.gph. The graph is created by a function that reads the file core_course.gph line by line.

```
Graph createGraph(BufferedReader file, Graph G)
```

(1) (10 marks) Program a function of CourseSequenceSolver that returns the direct prerequisites given a specific CourseCode. For example, returning MATH204 as the prerequisite of COEN212.

public string[] getPrerequisiste (string CourseCode)

(2) (10 marks) Program a function of CourseSequenceSolver that returns the course sequence of a given course. For example, MATH204, COEN243 are the course sequence of COEN244.

public string[] getCourseSequence (string CourseCode)

(3) (10 marks) Program a function of CourseSequenceSolver that returns boolean value if a source course is a prerequisite of the destination course.

public boolean isPrerequisite(string sourceCourse, string destinationCourse)

(4) (10 marks) Program a function of CourseSequenceSolver that returns boolean value if a given sequence of course is possible. For example, if the course sequence misses a certain prerequisite, false is returned.

public boolen getCourseSequence (string[] CourseSequence)

Problem 3 (20 marks): Program a unit test that has test cases for each of Problem 2 functions (1) – (4),

Please do not give any absolute path or other relative path. The reason is, the grading process will replace your file with a test file under the same name. Therefore, changing the file path rather than the current directory may cause your submission code not working.

```
BufferedReader f;
f = new BufferedReader(new InputStreamReader(new FileInputStream("core_course.gph")));
Graph G = new Graphm();
createGraph(f, G);
```

Submission Specification:

This is a group of two or individual assignment. The submission should contain

(1) all the source code of problem 1,2 and 3;
(2) the graph file;
(3) Please have screenshots of running the test cases and the status of the unit testing result from your IDE. A pdf file of the screenshots of running unit test cases.

in a single archive as .z, .gz. .tar, .zip. NO RAR file will be graded, following the naming convention [SID_1]_[SID_2]_A3.zip or [SID_1]_[SID_2]_A3.gz or [SID_1]_[SID_2]_A3.tar