PUSAT PENGAJIAN SAINS KOMPUTER
(School of Computer Sciences)

CPT212- Design and Analysis of Algorithms

Assignment II: Graph Algorithms Release Date: 10th June 2022 (Friday)

Due Date: 10th July 2022 (Sunday) by 11:59 pm

Objectives

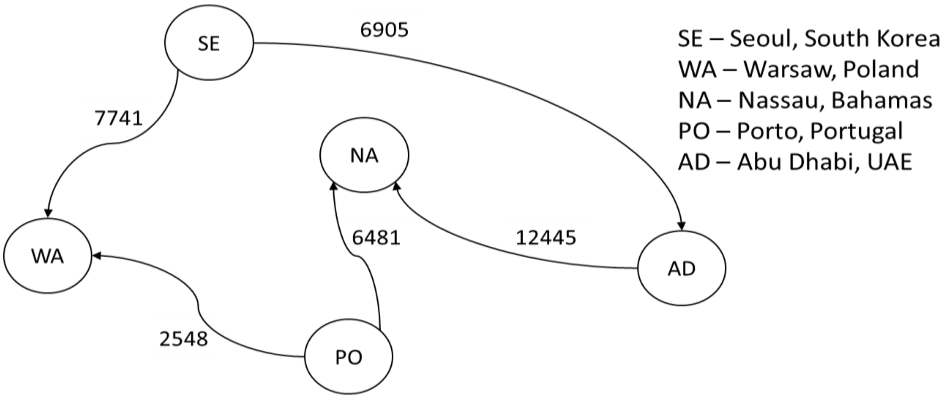
The purpose of this assignment is to test your understanding of **graph representation** and **algorithms**. You will be required to **code** the actual algorithms to solve problems that have been discussed in class.

Prerequisites

1. Each group must consist of 3 to 4 (max) persons. Individual work will not be accepted.
2. Each group will have **ONE** interactive report to document the functionality of their program.
3. Each group must produce a **ONE** cohesive program (one source file or project file).
4. You can choose to code in Python or C/C++ but choose only **ONE**.
5. **Please specify instructions on how to run your codes**.
6. You can use algorithms/codes from online sources to solve the given problems but must be **cited**.
7. Although this is a group project, a large percentage of the grade will be individual work (70% individual, 30% group).

Default Graph

1. Go to https://randomcity.net/ and pick 5 random cities and their corresponding countries.
2. Go to https://www.distancefromto.net/ and find out the distance (in kilometres) between each of these cities (you can round up/round down to get whole numbers).
3. Between each city, add only one directed, weighted edge.
4. Example graph (you cannot use the exact same one):



Instructions

1. Choose **ONE** data structure to represent the graph (E.g., adjacency list, adjacency matrix or incidence matrix).
2. There must be a list of common graph functions (E.g., add new edge, remove edge etc.). These functions must be used to modify the graph.
3. The **same data structure and functions** must be used by every group member.
4. Each time the program starts up, the **default graph** must be already **initialized** and can be modified by any of the functions. Do **NOT** prompt the user to key in the entire graph.
5. The group must write a program to solve **FOUR** problems:
   a. Function 1:
      Check if the graph is **strongly connected**. If it is not, **generate random edges* between random cities** until the graph is strongly connected. Print the resulting graph.
   b. Function 2:
      Check if the graph has a **cycle**. If it is not, **generate random edges* between random cities** until the graph has a cycle. Print the resulting cycle.
   c. Function 3:
      Allow the user to select two vertices and compute the **shortest path** between the vertices. If there is no path between the selected vertices, **generate random edges* between random cities** until the path exists. Print the shortest path.
   d. Function 4:
      Allow the user to select few edges and compute the minimum spanning tree (MST). If there is no MST can be generated, **generate random edges* between random cities** until MST exist. Print the MST.
   e. Include an additional function to **RESET** the graph to default.
   f. Include an additional function for the user to **REMOVE EDGES** of their choice.

   ** Any new edges that are added must have the correct **distance** between two particular cities. E.g., if an edge is created between Dhaka, Bangladesh and Kuala Lumpur, Malaysia, it should be around 2587km.*
6. Graphs that have been modified by any of the functions can be further modified by other functions. Do **NOT** reset the graph after each function. The graph will only reset if the program is closed, or the reset function is used.
7. The group can implement **ANY** algorithm to solve the problems.
8. Each member must be assigned **ONE** problem to solve.

Interactive Report Specifications

There is no specific format for the interactive report, but it **MUST** contain the following information:

a. Software: You can use any software which provide interactivity elements such as PowerPoint, Word/PDF, any smart document software. Make sure your interactive report can be read offline.

b. Font: You can use any of the default software fonts.

c. Please include **navigation buttons or links** to go back to your main page.

d. Front Page indicating division of tasks. Please include names, photos of your members, matric number and the problem being solved. Suggestion:

   i. Member 1 – Strong connectivity
   ii. Member 2 – Cycle detection
   iii. Member 3 – Shortest path
   iv. Member 4 – Minimum spanning tree

e. Description and **justification** of the chosen data structure and graph functions. **Cite** the source (website) where the functions were taken from.

f. Description of how all the problems were solved. Use **flowcharts** to aid your explanations.
   ****Please do not copy/paste your source code into the report****

g. Results:

   i. **Video** recording to highlight the features and functionalities of your program. Please **upload** the video to YouTube, Youku or any other video hosting website and **embed/include the link** in interactive report. Please do **not** attach the file there.
   ii. You can record 4 separate videos for each problem or 1 video for the whole group.
   iii. Provide a discussion (in the report) of the results to accompany your video. Highlight what was achieved, errors, problems, etc.

h. Please refer to the grading rubrics to ensure you fulfilled all requirements

Rubric (100%) – LO2/PO2

Category	Weak	Average	Good
Data structure (Group) – **5%**	No specific data structure was implemented (0-1%)	A data structure was chosen for the problem (2-3%)	A suitable data structure was chosen for the problem with proper justifications (4-5%)
Graph functions (Group) – **10%**	Minimal graph functions were implemented, or the functions were implemented in an ad-hoc manner (0-2%)	A workable set of graph functions were implemented and used in all the searching algorithms (3-5%)	An efficient set of graph functions were implemented and used in all the searching algorithms with proper justifications (6-10%)
Algorithm description (Individual) **– 30%**	Flowchart and discussion indicate a lack of understanding of the algorithm (0-10%)	The basic idea of the algorithm is apparent in the flowchart and discussion (11-20%)	Flowchart is easy to understand and in-depth discussion available (21-30%)
Algorithm results (Individual) **– 30%**	Algorithm produces wrong results or has errors (0-10%)	Algorithm is functioning but cannot add random edges (11-20%)	Algorithm is functioning and implemented well (21-30%)
Creativity (Individual) **– 10%**	Only basic functionality is present (0-2%)	Additional features have been included into the function (3-5%)	Additional features or modifications have been made to the basic algorithm (6-10%)
Overall Interactive Report (Group) **– 15%**	Badly written and structured (0-5%)	Reasonable language and structure (6-10%)	Well-written and structured (11-15%)