Assignment II: Graph Algorithms
Release Date: 17th May 2021 (Monday)
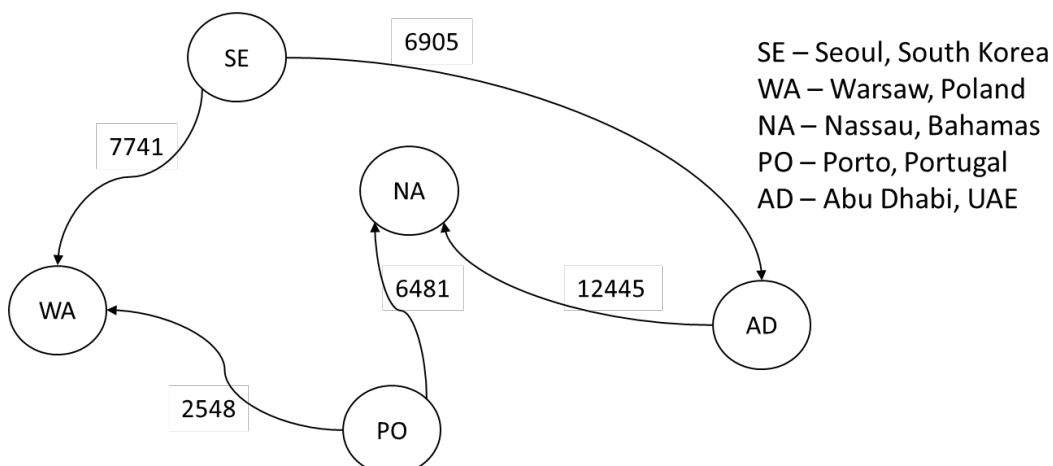**Due Date: 20th June 2021 (Sunday) by 11:59pm sharp**

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 individuals.
2. Each group will have **ONE** wiki 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 kilometers) 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):

SE – Seoul, South Korea
WA – Warsaw, Poland
NA – Nassau, Bahamas
PO – Porto, Portugal
AD – Abu Dhabi, UAE

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 **THREE** 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. Include an additional function to **RESET** the graph to default.
   e. 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.

Wiki Specifications

Please see appendix for tips on how to use the Wiki and some sample pages. There is no specific format for the wiki but it **MUST** contain the following information:

    a. Font: You can use any of the default eLearn fonts. For font size, preferably use "Medium" for regular text and "Extra Large" for headings.

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

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

        a. Member 1 – Strong connectivity

        b. Member 2 – Cycle detection

        c. Member 3 – Shortest path

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

    e. 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

    f. Results:

        a. 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 Wiki. Please do **not** attach the file there.

        b. You can record 3 separate videos for each problem or 1 video for the whole group.

        c. Provide a discussion (in the wiki) of the results to accompany your video. Highlight what was achieved, errors, problems, etc.

    g. Please refer to the grading rubrics to ensure you fulfilled all requirements
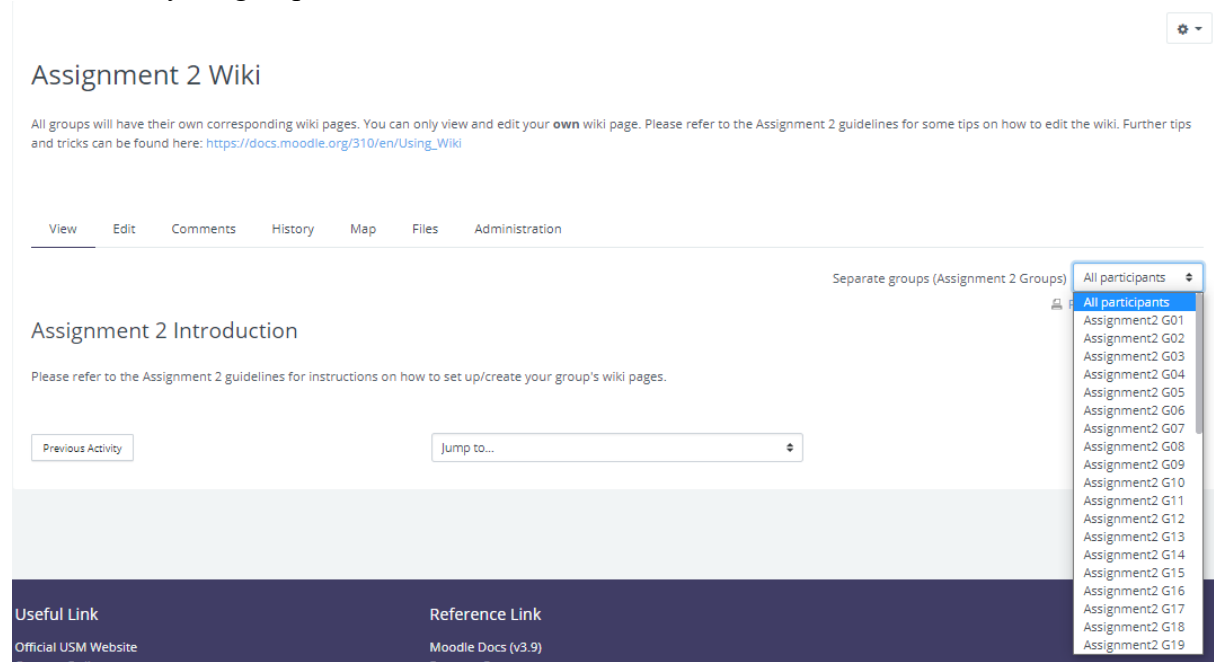
Rubric (100%) – LO2/PO2

Category	Weak	Average	Good
Data structure (Group)	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)	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)	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)	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)	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 Wiki (Group)	Badly written and structured (0-5%)	Reasonable language and structure (6-10%)	Well-written and structured (11-15%)

Appendix

A. Editing/Creating New Pages

First, select your group from the list:

To edit the wiki, click on the "Edit" button.

The main page must at least contain group information and some form of table of contents.

To **create a new page**, just include the page name in **double square brackets**. Example [[This is a new page]]:

The link will be red if the page is yet to be created. Click on it to create the new page.

Use HTML format and click on "Create page". The new page will be successfully created!

B. Adding Navigation/Hyperlinks

To add **links** to any pages in the wiki, all you need to do is to copy the URL of the page and create a hyperlink. You can view all the pages in the wiki under "Map".
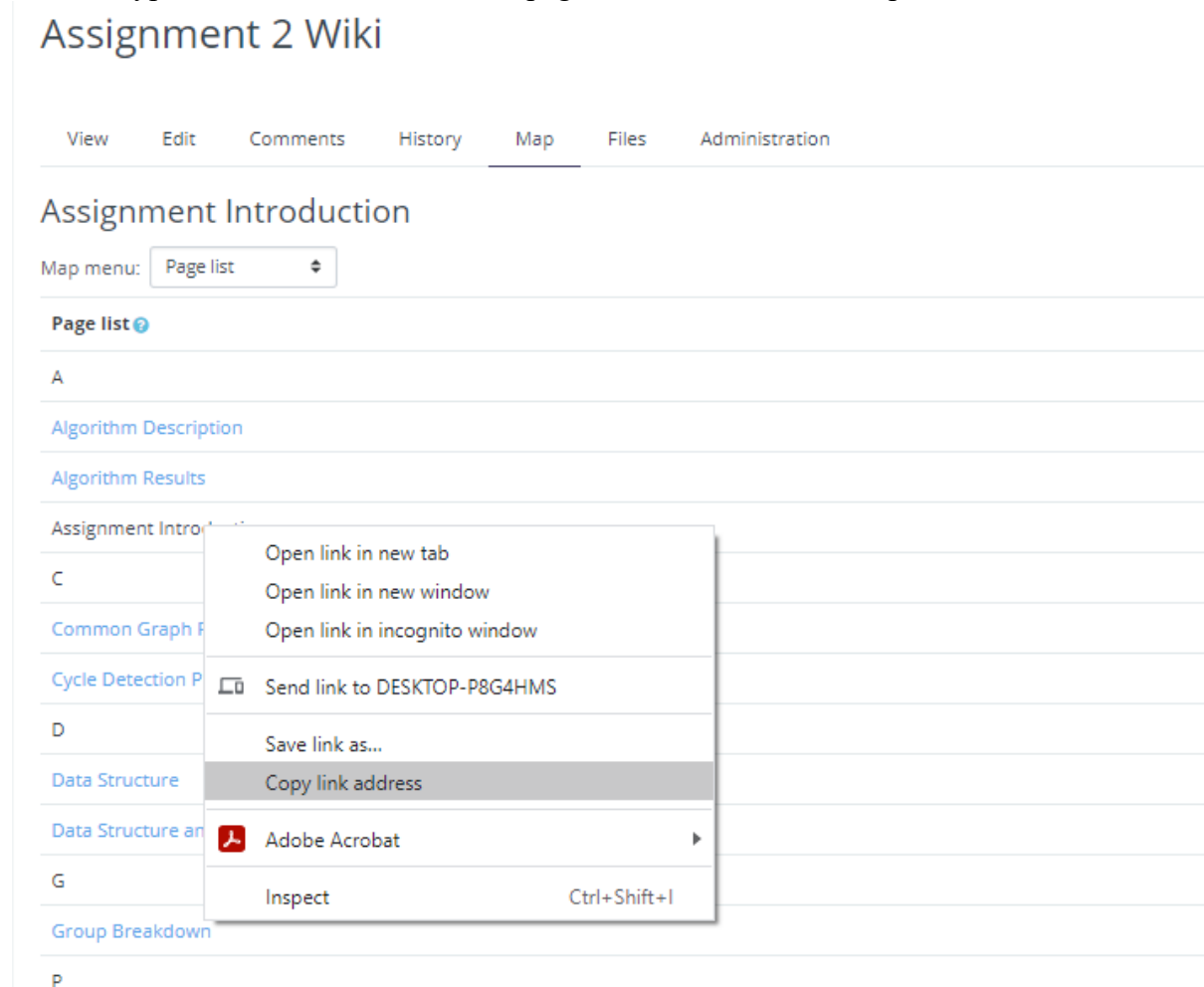
Right click, copy the link address and use it to create your hyperlinks.

First click on the Link button:
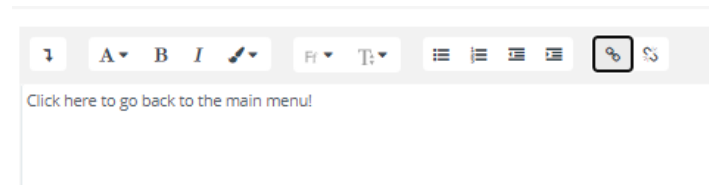
Paste your link.
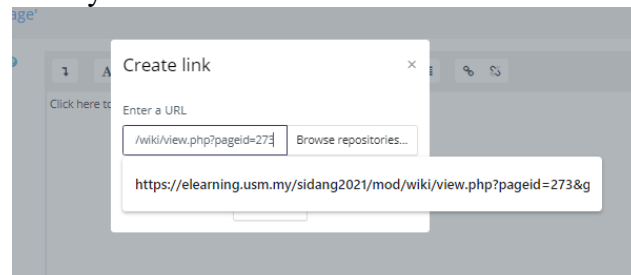
Your link is now ready!

C. Other Features

You can leave **comments** on specific pages by using the "Comments" tab. Each page can have its own comments.

You can **track changes** by using the "History" tab. You can even restore versions.

By including a **YouTube URL** when creating links, videos are automatically embedded (remember this is not compulsory, you can just leave the link as well)

D. Other Sample Pages

⚙ ▾

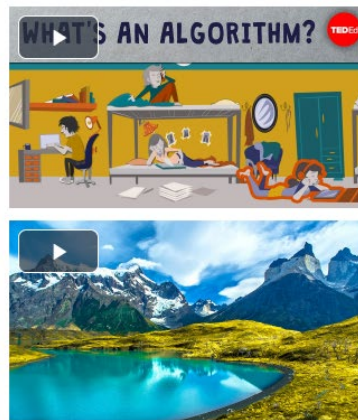## Assignment 2 Wiki

View   Edit   Comments   History   Map   Files   Administration

Separate groups (Assignment 2 Groups)   Assignment2 G01 ⬍

🖨 Printer-friendly version

This is a new page

Click here to go back to the main menu!

| Previous Activity | Jump to...   ⬍ |

⚙ ▾

Assignment 2 Wiki

View   Edit   Comments   History   Map   Files   Administration

Separate groups (Assignment 2 Groups)   Assignment2 G01 ⬍

🖨 Printer-friendly version

Strong Connectivity Problem

Please use the following buttons to navigate:

Algorithm Description - Flowchart and discussion of how the problem was solved

Algorithm Results - Video recording and discussion of the results

Click here to go back to the main page.

| Previous Activity | Jump to...   ⬍ |

⚙ ▾

Assignment 2 Wiki

View   Edit   Comments   History   Map   Files   Administration

Separate groups (Assignment 2 Groups)   Assignment2 G01 ⬍

🖨 Printer-friendly version

Algorithm Results

ALGORITHM

Discussion:

"Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum."

| Previous Activity | Jump to...   ⬍ |