# GTU Department of Computer Engineering
# CSE 222/505 - Spring 2021
# Homework 8
# Due date: June 26 2021– 23:55 PM

**This homework is extra homework and will replace the homework with the lowest grade if the grade of this extra homework is greater.**

## PART 1: 40 pts.

Generalize the implementation of Dijkstra's algorithm in the book so that:

1. Your method should run efficiently for both ListGraph and MatrixGraph representations of the graph.

2. Suppose the edges of a graph carry several properties. For example, the properties could be distance, time, or quality. Your method should run for any specified property of the edges.

3. In Dijkstra's algorithm implementation in the book, addition operator is used to combine an edge weight with a path weight. The addition operator can be replaced with any associative operator such as *addition*, *multiplication*, or $*$ operator. Note that $*$ be a binary operator defined as $a * b = a + b - ab$. The operator is associative since $(a * b) * c = a * (b * c)$. Your method should run for any specified associative operator.

You should write only one method to implement Dijkstra's algorithm and your method should provide all the properties above.

## PART 2: 30 pts.

i. Write two methods to find the number of connected components in a graph: one using BFS and the other using DFS.
ii. Perform a performance comparison of your methods
   - generate 10 random graphs for each graph size of $1000$, $2000$, $5000$ and $10000$ vertices and various sparsity. Note that your graph should contain several connected components.
   - run your methods and collect the running times.
   - Present the results using proper tables and figures.

## PART 3: 40 pts.

Let $G(V, E)$ be an undirected and unweighted graph. The importance of a vertex $v$ in $V$ can be calculated using the following formula.

$$\sum_{u \neq w \neq v \in V} \frac{\sigma_{uw}(v)}{\sigma_{uw}}$$

where
$\sigma_{uw}$ : The number of shortest paths between any two vertices $u$ and $w$ in the connected component including $v$, and
$\sigma_{uw}(v)$ : The number of shortest paths between any two vertices $u$ and $w$ (in the connected component including $v$) in which the vertex $v$ is an intermediate vertex on the shortest path from $u$ to $w$.

The calculate fair importance value, the result obtained by summation above is normalized by dividing the result by the square of the number of vertices in the connected component including $v$.

Implement a method that calculates the normalized importance values defined above for each vertex in a graph $G$.

**RESTRICTIONS:**

- Can be only one main class in project
- Don't use any other third part library

**GENERAL RULES:**

- For any question firstly use course news forum in Moodle, and then the contact TA.
- You can submit assignment one day late and will be evaluated over sixty percent (%60).

**TECHNICAL RULES:**

- You must write a driver function that demonstrates all possible actions in your homework. For example, if you are asked to implement an array list and perform an iterative search on the list then, you must at least provide the following in the driver function:
  - Create an array list and add items to the list. Append items to head, tail, and $k^{th}$ index of the list.
  - Perform at least two different searches by using two items in the list and print the index of the items.
  - Perform another search with an item that isn't in the array list and inform the user that the item doesn't exist in the array list.
  - Delete an existing item from the list and repeat the searches.
  - Try to delete an item that is not on the array list and throw an exception for this situation.

  The driver function should run when the code file is executed.

- Implement clean code standards in your code;
    - Classes, methods and variables names must be meaningful and related with the functionality.
    - Your functions and classes must be simple, general, reusable and focus on one topic.
    - Use standard java code name conventions.

**REPORT RULES:**

- Add all javadoc documentations for classes, methods, variables …etc. All explanation must be meaningful and understandable.
- You should submit your homework code, Javadoc and report to Moodle in a "studentid_hw8.tar.gz" file.
- Use the given homework format including selected parts from the table below:

| | |
|---|---|
| Detailed system requirements | X |
| The Project use case diagrams (extra points) | |
| Class diagrams | X |
| Other diagrams | |
| Problem solutions approach | X |
| Test cases | X |
| Running command and results | X |

**GRADING :**

| | |
|---|---|
| - **No OOP design:** | **-100** |
| - **No error handling:** | **-50** |
| - **No inheritance:** | **-95** |
| - No javadoc documentation: | -50 |
| - No report: | -90 |
| - Disobey restrictions: | -100 |
| - **Cheating:** | **-200** |

- Your solution is evaluated over 100 as your performance.

**CONTACT :**

- Teaching Assistant : Başak Karakaş
- bkarakas2018@gtu.edu.tr