# CSc 353: Homework Assignment 4

**Assigned: Wednesday October 25th, 2023**
**Due: 00:01am, Monday November 6th**

*Clear, neat and concise solutions are required in order to receive full credit so revise your work carefully before submission, and consider how your work is presented. If you cannot solve a particular problem, state this clearly in your write-up, and write down only what you know to be correct. Give credit when using external resources.*

## Introduction

In Homework 3, you further extended the intuitiveness of the graph visualization tool, by experimenting with different layout algorithms, searching for nodes, and extracting the largest connected component. Now, in Homework 4, the focus will be on graph editing and customization. You will improve upon the usability of the tool you have built so far, by allowing the user not only more options to interact with the tool, but also to use the tool as an interface to change the underlying graph data e.g. for correcting errors.

Specifically, for this homework assignment you are to implement the following functionality for the following node- and link-level tasks:

## (100 pts) Programming Part

1. **Change node label (10 pts)**: Until now our nodes have been labeled with movie IDs. Now we would like to allow the user to be able to change what type of label they want to see on each node. Specifically, the user should have an option to change the type of node label to each of the following movie attributes: movie ID, movie name, genre/s, or director name/s.

2. **Details on Demand (25 points)**: Design and implement an interface to display various attributes for a single movie. A json file with corrected movie poster image links is available under resources on the class Piazza page. Display the poster along with the following attributes in a single view:

   - movie name
   - movie ID
   - movie rank
   - year of release
   - imdb rating
   - total duration
   - genre/s
   - director name/s.

   Think about how to best a user can access this view from an HCI perspective e.g. you might consider using `Mouseover` events. For displaying, you might use a sidebar, or a popup window.

3. **Calculate Average Shortest Path Length (APL) (5 pts)**: Many real-world graphs exhibit a "Six degrees of separation" property, meaning that any two vertices are highly likely to have a shortest path length of no more than 6 (or proportional to $\log |V|$). These graphs are sometimes called "small-world networks", and often arise in social and collaboration type networks. Does the IMDB graph dataset exhibit this property? Compute the Average Shortest Path Length (APL), defined as the average length over all shortest paths between all pairs $(u, v) \in V$. Display this statistic along with the other computed statistics. Note: APL is not well defined for disconnected graphs. Determine how you will handle this (i.e. you cannot return infinity or zero as the APL value) and communicate what you have done to the user.

4. **Enhanced Node Search Capabilities (10 pts)**: In Homework 3, you implemented a feature that allowed users to search for nodes by movie ID. Now, you are to extend that feature & allow users to search for nodes based on attributes like genre and director name, in addition to movie ID. The nodes returned by the search should be highlighted in some way. For instance, highlighting all nodes from a specific genre or director. Specifically, your search must check the following fields and highlight all nodes that match:

    - rank
    - id
    - name
    - genre
    - cast_name
    - director_name
    - writter_name (writer is misspelled in the json)

5. **Modify Node attributes (10 pts)**: Add a feature in your application that allows users to select a specific node and modify its attributes, to allow expert users to correct errors in the data. This feature should allow a user to edit any existing attribute of a node from the list in question 4. Your tool should store the edits (i.e. the edits should be reflected in the underlying graph data). Think carefully how you will handle user inputs for each attribute.

6. **Delete nodes & edges (15 pts)**: Provide methods to remove nodes and edges from the visualization and ensure these changes are reflected in the underlying graph data. Think carefully about how to do this effectively. It should not be easy to accidentally remove a node or edge. Document how a user can achieve this functionality (and all others from above) in your readme file.

7. **Ensure backward compatibility (5 pts)**: The new functionality that you implement for this Homework should be compatible with all features you have introduced up to this point in previous Homeworks for e.g. the extraction of the largest connected component and the layout switching features introduced in Homework 3.

8. **Pass unit tests in Github classroom repository (20 pts)**: You are given a unit test which tests whether you computed the Average Shortest Path Length (APL) correctly for the IMDB graph data. You must pass this unit test locally and in the GitHub Classroom repository, as described below.

## Instructions for Github classroom

You will implement and submit your programs using GitHub Classroom. Click on the repository's link provided below and follow the instructions in the README file. **20 points of credit in this assignment are based on passing the unit tests** in the GitHub Classroom repository. Please see the submission instructions for GitHub Classroom by clicking on the link provided here: `https://classroom.github.com/a/cbSF3Qj8`

Very important note: **make sure the unit tests in your project pass on GitHub, after you submit your pull request!** If they do not, you will lose 20 points of credit.