# DV2618/DV2619 – Assignment 1 - Graph, Graph Searches and Implementation

**This is individual assessed assignment**. You are allowed to discuss this assessment with other students, but you should not copy their code, and you should not share your own code with other students. Note that we will carry out plagiarism checks on all submissions.

This assignment requires you to apply your knowledge of Graph, Graph Searches and implementation, **preferably in Python**. These topics are covered in the learning materials in Lectures 2 and 3, and it is strongly recommended that you work through the presentations from these weeks.

The **deadline for completing this assignment is the 13th of September 2024**. You should submit your work (**e.g., zip file including your report, implementation etc.**) on Canvas course webpage. Note that, the zip file should be named with your name and surname (e.g., name_surname.zip).

You will implement and compare different search algorithms (e.g., BFS, DFS, and A*) to find the shortest path in a complex 10x20 graph. You will analyze the efficiency, optimality, and performance of each algorithm.

**Graph-based maze design**: Find the original graph-based maze problem on the **second page**. For graph representation, you are free to use any format (e.g., XML, JSON etc.).

**Algorithm implementation**: Implement the following search algorithms:

- Breadth-First Search (BFS),
- Depth-First Search (DFS),
- A* Search.

**Pathfinding**: Use each algorithm to find the shortest path from the start to the goal in the maze.

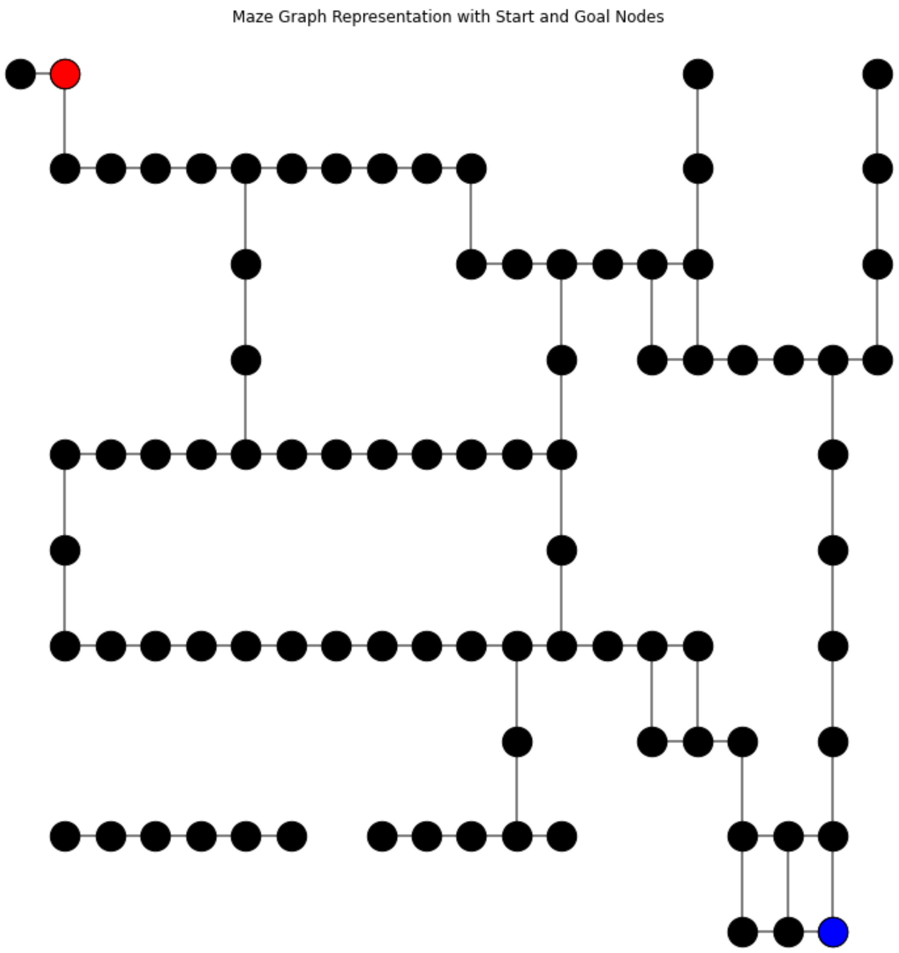
**Performance Comparison**: Compare the algorithms based on:

- Path optimality (shortest path)
- Path length (shortest path length): obtained by the algorithm
- Memory usage

**Visualization and Output**: Create a visual representation of the graph and the paths found by each algorithm in the implementation and report. Print the maze with the path from the start point to the goal point marked as '*’. If no path is found, indicate that no path exists.

**Report**: Write a report discussing the implementation details, performance metrics, and the strengths/weaknesses of each algorithm. Provide insights into which algorithm is most suitable for the maze problem. Discuss the results in detail.

Each algorithm will run in the implementation to find the shortest path between the starting point (**red '0' in the maze**) and the destination point (**blue '0' in the maze**) in the maze. The program should visually display the final path to the user. The maze consists of 10 rows and 20 columns, where **'1' should represent walls** and **'0' should represent free space in your graph representation**, as shown below:



**start = (0, 1)**
**end = (9, 18)**

**This coursework will be assessed according to:**

- implementing program with full functionality,
- your code must work properly,
- your code is appropriately commented,
- the output provides correct solution path and visually show the path,
- discussing the results in a report (e.g. Maximum 1 page).

**Important Note:** Please ensure that your code is fully functional and does not require any debugging or modifications. Teachers will not spend additional effort to troubleshoot or run your code. For full consideration and a successful evaluation of your assignment, your code must work seamlessly as provided. Additionally, you are allowed to use ChatGPT or Gemini as a helper tool, but you are not allowed to use it for plagiarism. Ensure that all work is original and properly credited.