# **Tutorial 9**

# **Depth First Search and Breadth First Search**

This task requires you to study the code from the repository and write the Depth First Search (DFS) and Breadth First Search (BFS) method for the dataset provided.

Source code: <https://github.com/zairulmazwan/Tutorial_9_Graphs.git>

1. Draw the graph for the dataset provided.
2. In pair study the following methods:

* addNode
* setEdge
* getNode
* getEdges
* printEdges
* containNode

1. Based on the algorithm from the lecture slides, write these methods to traverse the graph:

* Depth First Search
* Breadth First Search

# **Minimum Spanning Tree**

This task requires you to study the MST program and run an experiment for MST algorithm. We are going to look into how well the MST algorithm scales according to input size.

Conduct a set of experiments where you generate a number of random graphs varying in size from 100 nodes to 1500 nodes, and then time how long the MST takes to run. Repeat each of these experiments a number of iterations (each dataset), and then average the results. Plot your results to a graph (average).

Source code: <https://github.com/zairulmazwan/Tutorxial9_MST.git>

**Tasks:**

1. Write a method to generate a dataset (a return method). The dataset should be double type. The method should take a parameter (integer) for the size of the dataset (2D array). The dataset should be in a symmetric table. Example:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 0 | 60.04 | 88.19 | 56.92 | 28.2 |
| 60.04 | 0 | 13.47 | 22.42 | 75.31 |
| 88.19 | 13.47 | 0 | 91.56 | 34.45 |
| 56.92 | 22.42 | 91.56 | 0 | 77.98 |
| 28.2 | 75.31 | 34.45 | 77.98 | 0 |

1. In the main method of MST class, declare a variable for the experiment results (2D array of double type, row: 10, col: 2). Write a for loop that iterates for 10 times. Call a function namely ***runMST*** inside the for loop. The ***runMST*** method should take ***data size***, ***result***, and ***iteration*** (i).
2. Write a void method namely ***runMST*** in the MST class. This method should take the ***data size***, ***iteration***, and ***result***. The ***runMST*** method does the following:

* Create a timestart variable and set to 0.
* Initialise a dataset by calling the method in task 1.
* Call the ***PrimsMST*** method.

1. Modify the ***PrimsMST*** method in the MST class, where it should receive variables for ***timestart***, ***iteration***, ***data size***, and ***result***. Inside this method you should have the following:

* Start the timestart.
* Create a variable for timeEnd that takes the current time (milliseconds).
* Store the results into the ***result*** variable (2D array).
* Call the method namely ***writeResult***, pass the result and data size.

1. Write a method (***writeResult***) in the ***ReadWriteFile*** class for writing the results of the experiment. The method should receive the results.