COMP 3270 - Programming Assignment 2
All Answers will be in RED

Node 1	Node 2	Distance (BFS)	Time (ms) (BFS)	Distance (DFS)	Time (ms) (DFS)
N_0	N_1	1	1.93ms	1	0.04ms
N_0	N_2	2	2.14ms	2	0.12ms
N_0	N_3	3	1.36ms	3	0.27ms
N_0	N_4	8	1.10ms	8	0.08ms
N_0	N_5	3	2.09ms	3	0.11ms
N_0	N_6	2	1.43ms	2	0.04ms
N_0	N_7	3	1.39ms	3	0.04ms
N_0	N_8	6	2.22ms	6	0.13ms
N_0	N_9	7	2.32ms	7	0.15ms
N_0	N_10	4	2.58ms	4	0.09ms
N_0	N_11	5	2.44ms	5	0.08ms
N_0	N_12	4	1.99ms	4	0.30ms
N_0	N_13	5	2.33ms	5	0.06ms
N_0	N_14	6	2.05ms	6	0.07ms
N_0	N_15	5	1.76ms	5	0.09ms
N_0	N_16	6	2.03ms	6	0.08ms
N_0	N_17	5	0.74ms	5	0.05ms
N_0	N_18	6	2.18ms	6	0.08ms
N_0	N_19	7	1.53ms	7	0.08ms
N_0	N_20	6	1.97ms	6	0.38ms
N_0	N_21	7	2.80ms	7	0.12ms
N_0	N_22	6	2.54ms	6	0.06ms
N_0	N_23	7	2.25ms	7	0.06ms
N_0	N_24	8	2.60ms	8	0.07ms

1. Suppose you want to find a path between nodes at a shallow depth to your start node. Would you use BFS or DFS?
    a. If you wanted to find a path between nodes at a shallow depth, you would want to use BFS as that search starts at the root and explores the neighbor nodes at the current level before moving on to other nodes at the next level. This is better for finding nodes at a smaller depth or level.
2. Suppose that the end node is at a very large depth from the start node. Would you use BFS or DFS?
    a. If you wanted to find a path between nodes at a much larger depth, you would want to use DFS as that search starts at the root and explores the deepest depth as it can before moving back to the next unexplored path. This is better for finding nodes at a larger depth or level.

3. Your report should also have a brief description of the data structure that you will use to represent the graph and a short justification
    a. The data structure I used to represent the graph was as an adjacency list. This is because in an adjacency list, each node has its own list that contains only its neighbors, so no space was wasted on nodes that weren't connected by an edge. Because of this, traversal throughout the graph is efficient because you can access any node's neighbors directly from its list, which is useful for BFS and DFS.