**Lab Report 09**

Problem

According to the specifications, create an adjacent graph with methods to satisfy the given driver file

Solution Description

Using the class file from the lecture, I added methods to print the traversals from a vertex index input by the user.

Problems Encountered

Given the simplicity of the problem, I did not run into any problems while creating the class solution

4. DFS is a stack-based algorithm used to traverse through a graph. Given a vertex, print it to the console and check if an edge exists. If it does, follow it, add the following vertex to another data structure representing a marked list – used to store visited vertices – and repeat recursively. If a vertex has already been visited, backtrack and continue.

5. BFS is a queue-based algorithm used to traverse through a graph. Given a vertex, print it to the console and add it to a marked list and queue. As long as the queue is not empty, dequeue the vertex and check if a unique edge exists. If so, iterate through the matrix and print all unique vertices that are not currently in the queue, and enqueue each vertex.

6.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **v1** | **v2** | **v3** | **v4** | **v5** | **v6** | **v7** |
| **v1** | 0 | 1 | 0 | 1 | 0 | 0 | 0 |
| **v2** | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| **v3** | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| **v4** | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| **v5** | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| **v6** | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| **v7** | 0 | 0 | 0 | 0 | 1 | 0 | 0 |

7. v5, v1, v2, v7, v6, v3, v4

8. v5, v1, v2, v7, v6, v3, v4

9. v2, v7, v5, v1, v4, v6, v3

10. v2, v7, v5, v1, v4, v6, v3