**Lab 8 – Graphs**

**CSC 3302**

|  |  |
| --- | --- |
|  |  |
| **Introduction:** | Graphs are used in everything we use today. They are a way to represent networks of objects. In our case we use graphs to represent internet and internal network topology. Being able to read graph information is important for designing these systems. |
|  |  |
| **Program:** | Write a program called **p8.java**. This program will read a graph configuration from a file (as an adjacency matrix). You must also create a separate source file called **Graph.java** that will store this adjacency matrix and perform certain actions on it.  You program should create a Graph object for each adjacency matrix read in (as explained below) and you must print out the order vertices are visited from both a Breadth First and Depth First search. You may use any discussed implementations of data structures (Lists, stacks, queues) to perform these searches. |
|  |  |
| **Input:** | The adjacency matrix will be a series of rows corresponding to the relationship of vertices. For this assignment we are dealing with unweighted, undirected graphs.  Read from **p8.dat** and perform the searches on each matrix. The file can contain any number of matrices and is separated with a -1 between each line. Example:  6  0 1 0 0 1 0 1 0 1 1 1 1 0 1 0 1 1 0 0 1 1 0 0 1 1 1 1 0 0 0 0 1 0 1 0 0 -1  8 0 1 1 0 0 0 0 0 1 0 0 0 1 0 1 0 1 0 0 0 1 0 1 0 0 0 0 0 0 0 0 1 0 1 1 0 0 1 1 0 0 0 0 0 1 0 0 0 0 1 1 0 1 0 0 1 0 0 0 1 0 0 1 0  -1  In the above example each matrix is a separate graph and is separated with a -1. The matrix is a series of numbers of 0 or 1 separated with a space. The non -1 number above each matrix is the number of vertices in the graph.  Remember that each row corresponds to the edges from one vertex to another. Assume that each vertex is numbered from 0 to n-1. So in the first matrix, you have vertices 0,1,2,3,4,5 and a 0 represents no edge with a 1 representing an edge. |

|  |  |
| --- | --- |
|  |  |
| **Output:** | For each matrix in the file you should create a graph and display the order the nodes appear from both a Breadth First Search and a Depth First Search. Sample output:  Graph 1:  Breadth First Search: 0, 1, 4, 2, 3, 5  Depth First Search: 0, 1, 2, 3, 5, 4  Graph 2:  Breadth First Search: 0, 1, 2, 4, 6, 5, 7, 3  Depth First Search: 0, 1, 4, 2, 6, 7, 3, 5 |
|  |  |
|  |  |
|  |  |
| **Data:** | On the server, you will need to use the following absolute path and append the filename to it. "/home/courses/csci3302-002/datafiles/" |
|  |  |
|  | Remember that you must pass the data file name in as a command line argument. |