CMP_SC 3050 SP2024

ASSIGNMENT 5: IMPLEMENT BFS AND DFS

DESCRIPTION

You are to implement both a Breadth First Search and a Depth First Search for a graph using the provided library to represent the graph. The program will accept input via stdin. Note that you can redirect stdin to come from a file via the command-line and the starter project includes an example file that can be used as input for testing. The input format should be as follows:

```
<1 or 0 to indicate directed or not>
<start node>
<# of nodes>
<# of edges>
<node 1 of edge1><node 2 of edge 1><weight of edge 1>
<node 1 of edge1><node 2 of edge 2><weight of edge 2>
......
<node 1 of edge n><node 2 of edge n><weight of edge n>
```

For example, the following input describes an undirected graph with 3 nodes (indexed 1, 2 and 3) and 2 edges ({1,2} with weight 0.5 and {2,3} with weight 2.2), and a staring node of 1:

```
0
1
3
2
1 2 0.5
2 3 2.2
```

For the output, your program should print out each node as it becomes "black". For BFS, a start node is given. For DFS, just start at the first node (node #1). For BFS, put the distance to the start node in parenthesis. For DFS, put the "finish time" in parenthesis.

You are given a main() program, a Makefile, a library, and some starter code. All you need to do is complete the actual code for the functions in BFS.c and DFS.c. If you need a Queue data structure, you can easily implement this in C (I made a separate Queue.h and Queue.c for my implementation and so I added blanks to the Makefile for you).

HOW TO GET STARTED

The starter code is available from Canvas.

You are welcome to work on your code on any platform you like. However, you should be aware that submissions will **only** be evaluated by the TAs on tc.rnet.missouri.edu. If, for example, something works on your machine but doesn't compile on tc.rnet.missouri.edu, you will get a zero.

Once you have the starter code in a directory, just type "make". This will build the code and leave you with an executable file called "test". You can run this file with the included sample input file (CLRS.4E.p557.G) this way:

```
./test <CLRS.4E.p557.G
```

SAMPLE OUTPUT

```
jimr@jimrsurfacepro9:~/CS3050/SP2024/assignments/A5/solution$ ./test <CLRS.4E.p557.G
Enter 1 for directed or 0 for undirected: Enter the node to start algorithm: Enter the number of vertices:
Enter the number of edges:

****Graph:
UNDIRECTED
Start node: 6

Vertex 1:
        1->2 (1.00)
        1->3 (1.00)
        1->4 (1.00)
        1->5 (1.00)

Vertex 2:
        2->1 (1.00)
        2->5 (1.00)
        2->9 (1.00)

Vertex 3:
        3->1 (1.00)
        3->6 (1.00)
        3->7 (1.00)

Vertex 4:
        4->1 (1.00)
        4->6 (1.00)
        4->9 (1.00)

Vertex 5:
        5->1 (1.00)
        5->2 (1.00)

Vertex 6:
        6->3 (1.00)
        6->4 (1.00)
        6->8 (1.00)

Vertex 7:
        7->3 (1.00)
        7->8 (1.00)

Vertex 8:
        8->6 (1.00)
        8->7 (1.00)
        8->9 (1.00)

Vertex 9:
        9->2 (1.00)
        9->4 (1.00)
        9->8 (1.00)


BFS:
6(0) 3(1) 4(1) 8(1) 1(2) 7(2) 9(2) 2(3) 5(3)


DFS:
5(4) 8(11) 7(12) 3(13) 6(14) 4(15) 9(16) 2(17) 1(18)
```