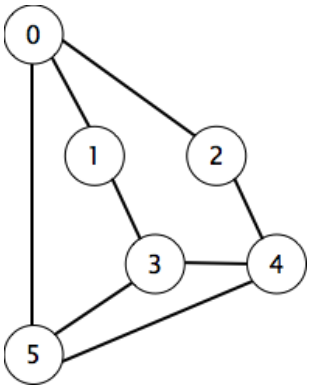
CSC 2053 Graph Traversals

1. Trace DFS on this graph:

Trace the algorithm by hand, paying attention to visit the vertices according to the order in which they are appear in the adjacency lists.



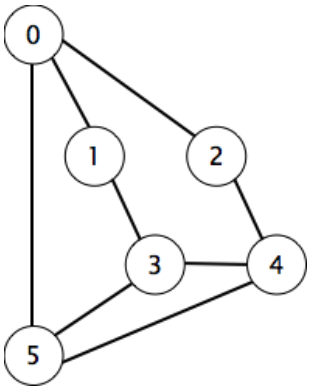
v	adjacency lists
0:	5 2 1
1:	0 3
2:	4 0
3:	4 1 5
4:	3 2 5
5:	0 4 3

v	marked[v]	edgeTo[v]
0		
1		
2		
3		
4		
5		

Draw a picture of the tree produced by depth-first search:

2. Trace BFS on this graph:

Trace the algorithm by hand, paying attention to visit the vertices according to the order in which they are appear in the adjacency lists.



v	adjacency lists
0:	5 2 1
1:	0 3
2:	4 0
3:	4 1 5
4:	3 2 5
5:	0 4 3

v	marked[v]	edgeTo[v]
0		
1		
2		
3		
4		
5		

Draw a picture of the tree produced by breadth-first search:

3. Try the depth-first and breadth-first implementations that we studied on the above graph.

We need to be careful how to input the graph, so that the adjacency lists come out the same. Use the following input file:

```
6
8
3 5
4 5
1 3
0 1
0 2
2 4
0 5
3 4
```

First, run it through `Graph.java` to verify that you got the adjacency lists as above. Then run it through `DepthFirstPaths` and `BreadthFirstPaths` to verify that you traced the algorithm correctly.

Repeat with some other vertices as sources and draw the resulting graphs here (do a quick hand-trace to verify that the answers make sense!)

4. Can you modify BreadthFirstSearch to use a Stack instead of a Queue to obtain DepthFirstSearch?