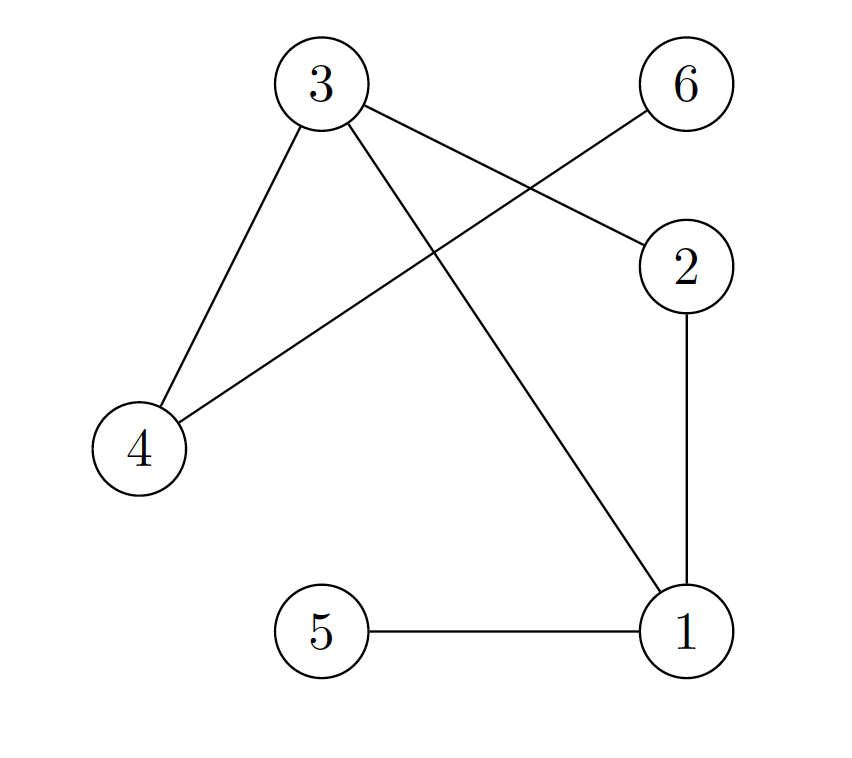
**CS 3050 Homework # 5. Name :**

**Submitted to Blackboard, due at 11:59pm on April 20, 2018.**

1. Give an adjacency-list representation for a binary tree on 6 vertices. Give an equivalent adjacency-matrix representation. Assume that vertices are numbered from 1 to 6 as in a binary heap.

2. Consider the following undirected graph G:



1. Give the adjacency matrix representation of G.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| G | 1 | 2 | 3 | 4 | 5 | 6 |
| 1 | 0 | 1 | 1 | 0 | 1 | 0 |
| 2 | 1 | 0 | 1 | 0 | 0 | 0 |
| 3 | 1 | 1 | 0 | 1 | 0 | 0 |
| 4 | 0 | 0 | 1 | 0 | 0 | 1 |
| 5 | 1 | 0 | 0 | 0 | 0 | 0 |
| 6 | 0 | 0 | 0 | 1 | 0 | 0 |

1. Give the adjacency list representation of G for every vertex.

[1] -> {2, 3, 5}

[2] -> {1, 3}

[3] -> {1, 2, 4}

[4] -> {3, 6}

[5] -> {1}

[6] -> {4}

1. Give the search ordering by running the Breadth First Search (BFS) algorithm on G with 3 as the source vertex.

Output of BFS [Node (Level)]: 3(0), 4(1), 1(1), 2(1), 6(2), 5(2)

(d) Give the search ordering generated by running the Depth First Search (DFS) algorithm on G with 3 as the source vertex. For each vertex in the DFS tree, please also indicate the start time and the finish time generated by the DFS algorithm.

Output of DFS [Node (D/F)]: 6(3/4), 4(2/5), 5(7/8), 2(9/10), 1(6/11), 3(1/12)

3. Make a 3-by-3 chart with row and column labels WHITE, GRAY, and BLACK. In each cell (i,j), indicate whether, at ANY point during a depth-first search of a directed graph, there can be an edge from a vertex of color i to a vertex of color j . For each possible edge, indicate what edge types it can be (Tree edge, Back edge, Forward edge, or Cross edge). Make a second such chart for depth-first search of an undirected graph.

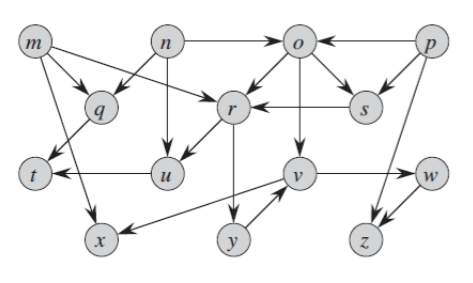
**DFS Directed Graph**

|  |  |  |  |
| --- | --- | --- | --- |
| (row i, column j) | WHITE | GRAY | BLACK |
| WHITE | Any Edge | Back / Cross | Cross |
| GRAY | Tree/ Forward | Tree/Back/Forward | Forward/Cross/Tree |
| BLACK | Not Possible | Back | Any Edge |

**DFS Undirected Graph**

|  |  |  |  |
| --- | --- | --- | --- |
| (row i, column j) | WHITE | GRAY | BLACK |
| WHITE | Tree/Back | Tree/Back | Not Possible |
| GRAY | Tree/ Back | Tree/Back | Tree/Back |
| BLACK | Not Possible | Tree/Back | Tree/Back |

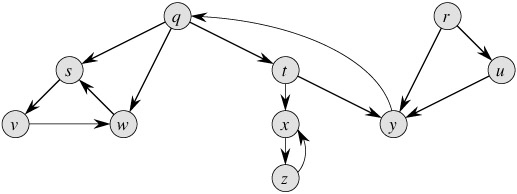
4. Show the ordering of vertices produced by topological-sort of the following graph (start the DFS at node r):



Reverse Topological Order:

p(27/28), n(23/26), o(24/25), m(19/22), q(20/21), s(17/18), r(1/16), y(6/15), v(7/14), w(10/13), z(11/12), x(8/9), u(2/5), t(3/4)

5. Show how the procedure STRONGLY-CONNECTED-COMPONENTS works on the following graph. Specifically, show the finishing times computed in line 1 and the forest produced in line 3. Assume that the loop of lines 5-7 of DFS considers vertices in alphabetical order and that the adjacency lists are in alphabetical order. (from the textbook problem 22.5-2)



Finishing times computed in line 1:

r(17/20), u(18/19), q(1/16), t(8/15), y(13/14), x(9/12), z(10/11), s(2/7), v(3/6), w(4/5)

Forests computed in line 3:

{r},{ u},{q, y, t},{x, z},{s, w, v}