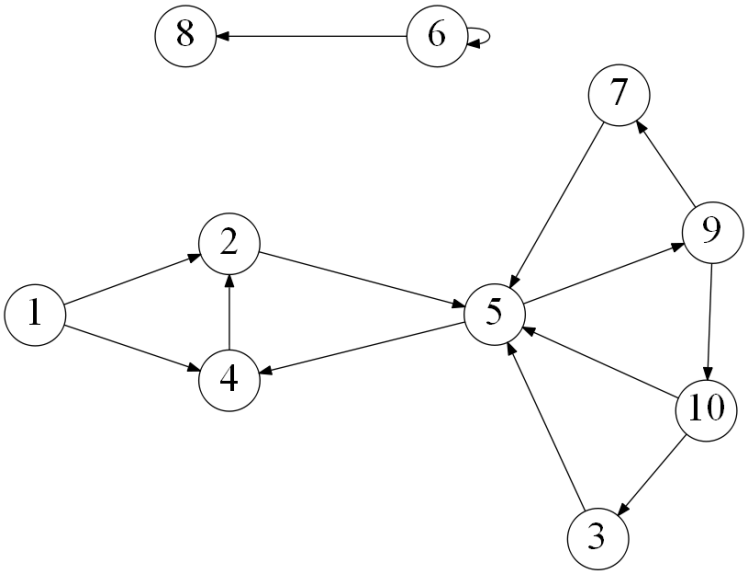
Name: __Grace Miguel__ Date: __3/21/21___

I pledge my honor that I've abided by the Stevens Honor System.

Point values are assigned for each question. Points earned: ____ / 100

Consider the following graph:




1. Draw how the graph would look if represented by an adjacency matrix. You m3,ay assume the indexes are from 1 through 10. Indicate 1 if there is an edge from vertex A -> vertex B, and 0 otherwise. (10 points)

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 3 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 4 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 6 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 |
| 7 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 9 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| 10 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |

2. Draw how the graph would look if represented by an adjacency list. You may assume the indexes are from 1 through 10. (10 points)

| | |
|---|---|
| 1→ | 2,4 |
| 2→ | 5 |
| 3→ | 5 |
| 4→ | 2 |

| | |
|---|---|
| 5→ | 9 |
| 6→ | 6,8 |
| 7→ | 5 |
| 8→ | |
| 9→ | 7,10 |
| 10→ | 3,5 |

3. List the order in which the vertices are visited with a breadth-first search. If there are multiple vertices adjacent to a given vertex, visit the adjacent vertex with the lowest value first. (10 points)
   1,2,4,5,9,7,10,3,6,8

4. List the order in which the vertices are visited with a depth-first search. If there are multiple vertices adjacent to a given vertex, visit the adjacent vertex with the lowest value first. (10 points)
   1,2,5,9,7,10,3,4,6,8

5. a) What is the running time of breadth-first search with an adjacency matrix? (5 points)
      a. O(V^2)
   b) What is the running time of breadth-first search with an adjacency list? (5 points)
      O(|V|+|E|)

6. a) What is the running time of depth-first search with an adjacency matrix? (5 points)
      a. O(V^2)
   b) What is the running time of depth-first search with an adjacency list? (5 points)
      O(|V|+|e|)

7. While an adjacency matrix is typically easier to code than an adjacency list, it is not always a better solution. Explain when an adjacency list is a clear winner in the efficiency of your algorithm? (5 points)
      a. Adjacency matrices are good for small and dense graphs whereas as adjacency lists are more efficient for large and sparse graphs. It is easier to extrapolate information from the adjacency list than a matrix. The run time will be faster because you won't be looping through a bunch of empty spots.

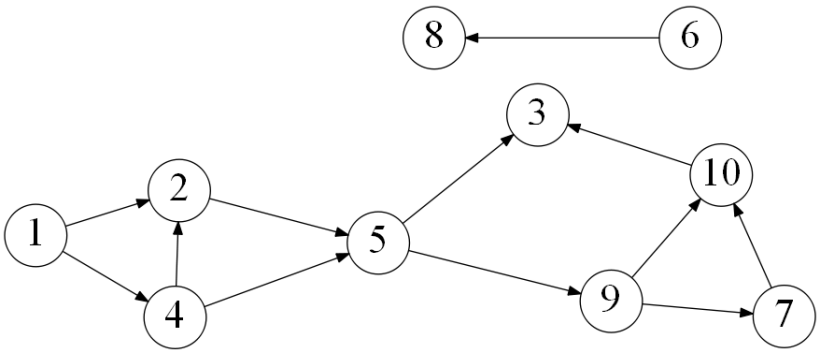
8. Explain how one can use a breadth-first to determine if an undirected graph contains a cycle. (10 points)
      a. You can use bfs to determine if an undirected graph contains a cycle because you are marking each node as visited or unvisited. If you loop through a node that you have visited then you know there is a cycle because there's a duplicate.

9. On undirected graphs, does either of the two traversals, DFS or BFS, always find a cycle faster than the other? If yes, indicate which of them is better and explain why it is the case; if not, draw two graphs supporting your answer and explain the graphs. (10 points)

a. DFS usually finds cycles faster because when you are going through the depth of one node, it is checking for visited or unvisited nodes whereas with bfs it is checking for edges and is less efficient.

10. Explain why a topological sort is not possible on the graph at the very top of this document. (5 points)
    a. In topological sorting, you are supposed to have vertex's coming one after another. In the graph above you have vertexes repeating themselves in the list and being visited again. 4 is directed towards 2 which is not the way that this is supposed to work. You are supposed to be working forward, not back.

Consider the following graph:




11. List the order in which the vertices are visited with a topological sort. Break ties by visiting the vertex with the lowest value first. (10 points)

| 1 | 2 | 4 | 5 | 3 | 9 | 7 | 10 | 6 | 8 |
|---|---|---|---|---|---|---|---|---|---|