DSC 40B - Quiz 07

Name: Student ID:

About the quizzes:

- Quizzes in DSC 40B are *optional* and graded pass/fail.
- A score of 70% or higher earns a "pass" and 1.5 credits toward your final grade.
- If you don't pass, no credits are earned, but it doesn't hurt your grade.
- You have 30 minutes to complete the quiz.
- Unfortunately, we can't answer clarifying questions during the quiz. If you think a question has a bug or is unclear, please let us know in a private post on Campuswire after the quiz, and we'll take it into account when grading.

Problem 1.

An **undirected** graph has n nodes, where n is even, and every node has exactly $n/2$ neighbors. What is the time complexity of a full BFS of this graph, as a function of n?

◯ $\Theta(n)$ ◯ $\Theta(n \log n)$ ◯ $\Theta(n^2)$ ◯ $\Theta(n^3)$

Problem 2.

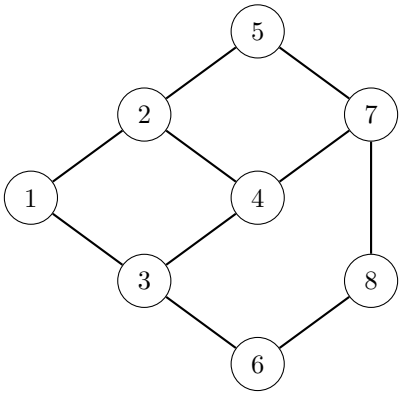
Suppose a *breadth* first search (BFS) is run on a graph $G = (V, E)$, and that node u is distance 5 from the source while node v is distance 3 from the source. True or False: it is possible that at some point during the BFS, both u and v are in the pending queue at the same time.

◯ True

◯ False

Problem 3.

Suppose a BFS is run on the undirected graph below using node 1 as the source, and using the convention that a node's neighbors are produced in ascending order by label. Which node will be the BFS predecessor of node 7?



◯ 3 ◯ 4 ◯ 5 ◯ 8

Problem 4.

`bfs_shortest_paths` (from lecture) is run on an **undirected** graph with source node s, and it returns the dictionary of BFS predecessors below. You aren't shown the graph.

```
predecessor = {
    's': None, 'a': 's', 'b': 's', 'c': 'a',
    'd': 'c', 'e': 'b', 'f': 'd', 'g': None
}
```

What is the shortest path distance from s to f?

◯ 1 ◯ 3 ◯ 4 ◯ 6

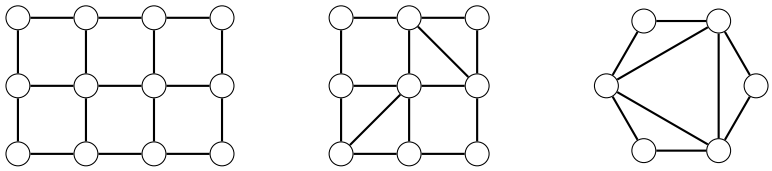
Problem 5.

The code below is `full_dfs` and `dfs` from lecture, with a `print` added at the top of `dfs`.

```
def full_dfs(graph):
    status = {node: 'undiscovered' for node in graph.nodes}
    for node in graph.nodes:
        if status[node] == 'undiscovered':
            dfs(graph, node, status)

def dfs(graph, u, status=None):
    print("Hello!")
    if status is None:
        status = {node: 'undiscovered' for node in graph.nodes}
    status[u] = 'pending'
    for v in graph.neighbors(u):
        if status[v] == 'undiscovered':
            dfs(graph, v, status)
    status[u] = 'visited'
```

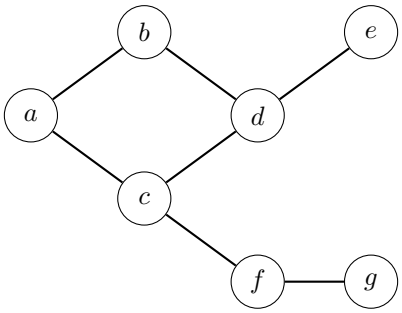
Suppose `full_dfs` is run on the **undirected** graph below, which has three connected components. How many times will `"Hello!"` be printed?



◯ 3 ◯ 27 ◯ 40 ◯ 80

Problem 6.

Suppose a DFS is run on the undirected graph below to compute start and finish times, using node a as the source. Use the convention that the start time of a is 1, and that a node's neighbors are produced in alphabetical order.



a) What will be the **start** time of node c?

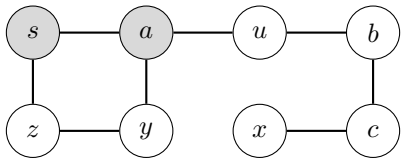
◯ 2 ◯ 3 ◯ 4 ◯ 10

b) What will be the **finish** time of node c?

◯ 7 ◯ 9 ◯ 12 ◯ 13

Problem 7.

A DFS is run on the **undirected** graph below with s as the source, and neighbors are produced in alphabetical order. At the moment `dfs(u)` is called, s and a (shaded) are pending, and every other node is undiscovered.

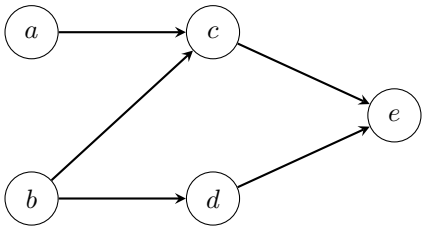


How many calls to `dfs` will start *and* finish during the call to `dfs(u)`? Don't count `dfs(u)` itself.

◯ 1 ◯ 3 ◯ 5 ◯ 7

Problem 8.

Which one of the orderings below is a valid topological sort of this DAG?



◯ a, c, e, b, d ◯ b, a, c, e, d ◯ a, b, d, c, e ◯ a, b, d, e, c

Problem 9.

BFS finds shortest paths. Does DFS find longest paths?

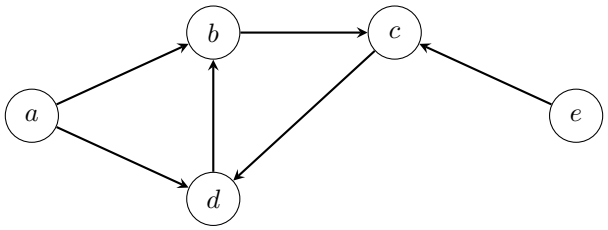
Let G be a connected, **undirected** graph, and suppose a DFS is run from a node s, recording each node's *predecessor*: the node it was discovered from. True or False: for every node v, following the DFS predecessors back from v to s gives a **longest** simple path from s to v.

◯ True

◯ False

Problem 10.

A full DFS is run on the **directed** graph below. It loops over the nodes in alphabetical order, and neighbors are also produced in alphabetical order. Which one of these edges is found to be a **back edge**?



◯ (a, d) ◯ (b, c) ◯ (d, b) ◯ (e, c)