PID or Name: _____

DSC 40B - Sample Midterm 02

Note: This sample midterm is intended to give you an idea of the format of the exam, but it's not intended to be a comprehensive review of the material. Also, note that this sample exam is from a previous iteration of the course, and topics can change slightly from quarter to quarter depending on the instructor and how much was covered in lecture – you should expect Midterm 01 to cover the content from *this quarter*'s Lecture 09 through 15.

In addition to this sample exam, you should also study using the problems found at `https://dsc40b.com/practice`, as well as the labs and the homeworks.

Problem 1.

Suppose a hash table is constructed so that collisions are resolved through chaining. The hash table has 10 bins, and 100 items are inserted into the hash table.

True or False: it is possible for one of the bins to contain all 100 items.

◯ True

◯ False

Problem 2.

Consider the code below:

```
def intersection(A, B):
    common = set()
    for x in A:
        if x in B:
            common.add(x)
    return common
```

What is the expected time complexity of this code if `A` and `B` are Python `set`s (hash tables) with n elements each? State your answer as a function of n using asymptotic notation. You may assume that the hash function used is "good" in that it hashes evenly (that is, it satisfies the simple universal hashing assumption).

Problem 3.

An *undirected* graph has 5 nodes. What is the greatest number of edges it can have?

Problem 4.

A *directed* graph has 10 *edges*. What is the *smallest* number of *nodes* it can have?

Problem 5.

An undirected graph with 20 ndoes and 30 edges has three connected components: A, B, and C. Connected component A has 7 nodes. What is the smallest number of edges it can have?

Problem 6.

Let $v = \{a, b, c, d, e, f, g\}$ and $E = \{(a, g), (b, d), (d, e), (f, d)\}$. How many connected components does the undirected graph $G = (V, E)$ have?

Problem 7.

Let $G = (V, E)$ be a directed graph with $|V| > 1$. Suppose that every node in G is reachable from every other node. True or False: each node in the graph must have an out-degree of at least one and an in-degree of at least one.

◯ True

◯ False

Problem 8.

Let G be an undirected graph, and let u, v, and z be three nodes in the graph. Consider the problem of finding a shortest path from u to v which passes through node z.

True or False: a shortest path from u to v passing through z may include node u twice.

◯ True

◯ False

Problem 9.

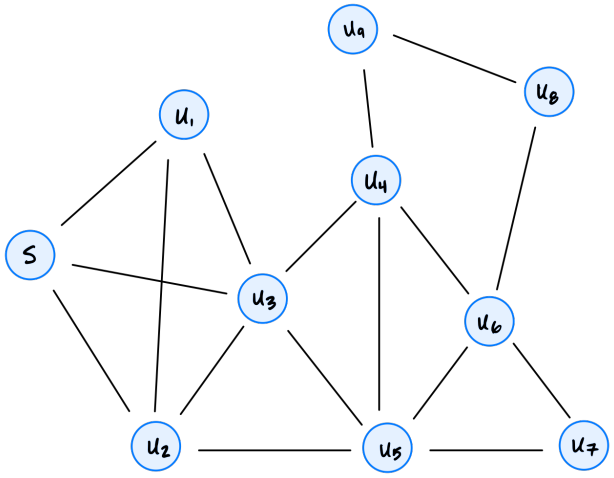
An *isolated* node in an undirected graph is a node that has no neighbors.

Suppose a graph $G = (V, E)$ is represented using an adjacency matrix. What is the best possible worst-case time complexity of an efficient algorithm for computing the number of isolated nodes in the graph? State your answer as a function of $|V|$ and $|E|$ using asymptotic notation.

Problem 10.

Consider running a breadth-first search (BFS) on the graph shown below, using node s as the source and adopting the convention that neighbors are produced in ascending order by label.

PID or Name: _____



How many nodes will be popped from the queue before node u_8 is popped?

Problem 11.

Suppose that a breadth-first search on an undirected graph G is paused and the queue is printed. Suppose that every node in the queue is either distance 5 from the source, or distance 6. At this moment, node u is undiscovered.

True or False: it is possible that node u is distance 4 from the source.

◯ True

◯ False

Problem 12.

Suppose a breadth-first search (BFS) is performed in order to calculate a dictionary of shortest path distances, `dist`, from a source node s to all nodes reachable from s in an undirected graph $G = (V, E)$.

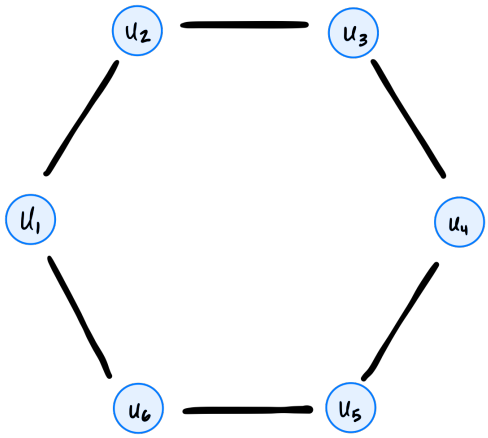
Suppose node u is a node in G, and v_1 and v_2 are two neighbors of node u. Assume that u is reachable from s.

a) What is the *largest* that `|dist[v_2] - dist[v_1]|` can be?

b) What is the *smallest* that `|dist[v_2] - dist[v_1]|` can be?

Problem 13.

Define a *circle graph* with n nodes to be an undirected graph $G = (V, E)$ with nodes $u_1, u_2, \ldots, u_n$ and edges $(u_1, u_2), (u_2, u_3), \ldots, (u_{n-1}, u_n)$, along with one additional edge (u_n, u_1) completing the cycle. An example of a circle graph with 6 nodes is shown below:



What is the time complexity of breath-first search when run on a circle graph with n nodes? State your answer as a function of n using asymptotic notation.

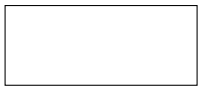
Problem 14.

Consider the code below, which is a modification of the code for `bfs` given in lecture:

```
from collections import deque

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

def modified_bfs(graph, source, status=None):
    """Start a BFS at `source`."""
    if status is None:
        status = {node: 'undiscovered' for node in graph.nodes}

    status[source] = 'pending'
    pending = deque([source])

    # while there are still pending nodes
    while pending:
        u = pending.popleft()
        for v in graph.neighbors(u):
            # explore edge (u,v)
            if status[v] == 'undiscovered':
                status[v] = 'pending'
                # append to right
                pending.append(v)
                for z in graph.nodes:    # <----- new line!
                    print(z, status[z]) # <----- new line!

        status[u] = 'visited'
```

Notice the two new lines; these lines print all node statuses any time that an undiscovered node has been found.

What is the time complexity of this modified `modified_full_bfs`? State your answer in asymptotic notation in terms of the number of nodes, $|V|$, and the number of edges, $|E|$. You may assume that the graph is represented using an adjacency list.



Problem 15.

Consider calling `full_bfs` on a *directed* graph, and recall that `full_bfs` will make calls to `bfs` until all nodes have been marked as visited.

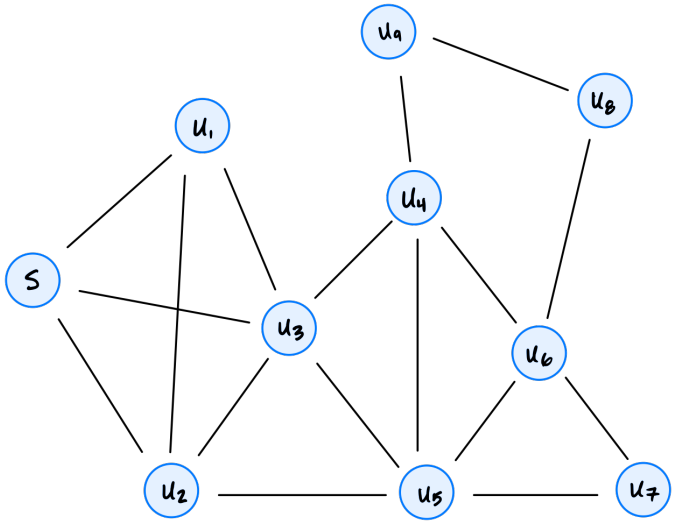
True or False: the number of calls made to `bfs` depends on the order in which `graph.nodes` produces the graph's nodes. That is, if nodes are produced in a different order, the number of calls to `bfs` may be different.

◯ True

◯ False

Problem 16.

Suppose a depth-first search (DFS) is run on the graph shown below using node s as the source and adopting the convention that a node's neighbors are produced in ascending order of their label.



a) What will be the start time of node u_7?

b) What will be the finish time of node u_7?

c) Which node will be the DFS predecessor of node u_7?

Problem 17.

Let G be a graph, and suppose s, u, and v are three nodes in the graph.

Suppose a depth-first search (DFS) is run on G using node s as the source and adopting the convention that a node's neighbors are produced in ascending order by label. During this DFS, start and finish times are computed. Suppose it is found that:

$$\texttt{start[u]} < \texttt{start[v]} < \texttt{finish[v]} < \texttt{finish[u]}.$$

Now suppose another DFS is run using node s as the source, but adopting a different convention about the order in which neighbors are produced. Suppose `new_start` and `new_finish` are the start and finish times found by this second DFS. True or False: it must be that

$$\texttt{new\_start[u]} < \texttt{new\_start[v]} < \texttt{new\_finish[v]} < \texttt{new\_finish[u]}.$$

◯ True

◯ False

PID or Name: _____

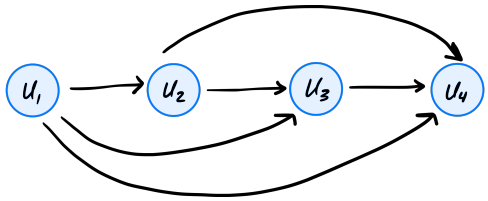
Problem 18.

Consider the graph $G = (V, E)$ with $V = \{a, b, c\}$ and $E = \{(a, b)\}$.

How many valid topological sorts of this graph are there?

Problem 19.

Suppose we define a *forward-looking* graph with n nodes to be the directed graph $G = (V, E)$ with nodes $u_1, \ldots, u_n$ and the edge (u_i, u_j) if and only if $i < j$. An example of such a graph with 4 nodes is shown below.



What is the time complexity of depth-first search when run on a forward-looking graph with n nodes, using node u_1 as the source? State your answer as a function of n using asymptotic notation.

Problem 20.

Suppose an undirected graph $G = (V, E)$ is a tree with 33 edges. Suppose a node s is used as the source of a depth-first search.

Including the root call of `dfs` on node s, how many calls to `dfs` will be made?

Problem 21.

Let $G = (V, E)$ be an undirected **tree**.

Suppose breadth-first search (BFS) and depth-first search (DFS) are each run on the graph using the same source node. True or False: for any node u in the graph, the BFS predecessor of u and the DFS predecessor of u must be the same.

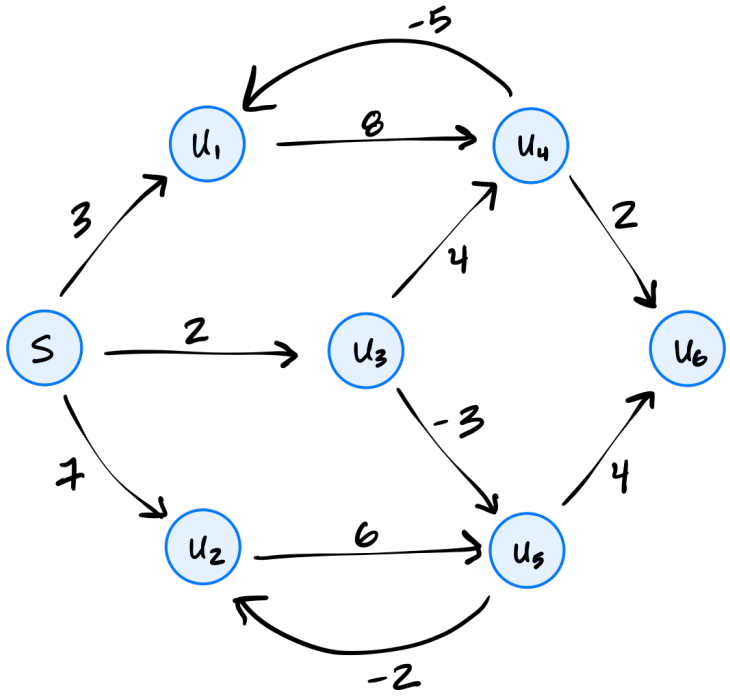
◯ True

◯ False

Problem 22.

Recall that the outer loop of the Bellman-Ford algorithm *without* early stopping loops for a fixed number of iterations, while the inner loop iterates over each edge of the graph.

Suppose Bellman-Ford *with* early stopping is run on the graph below using node s as the source:

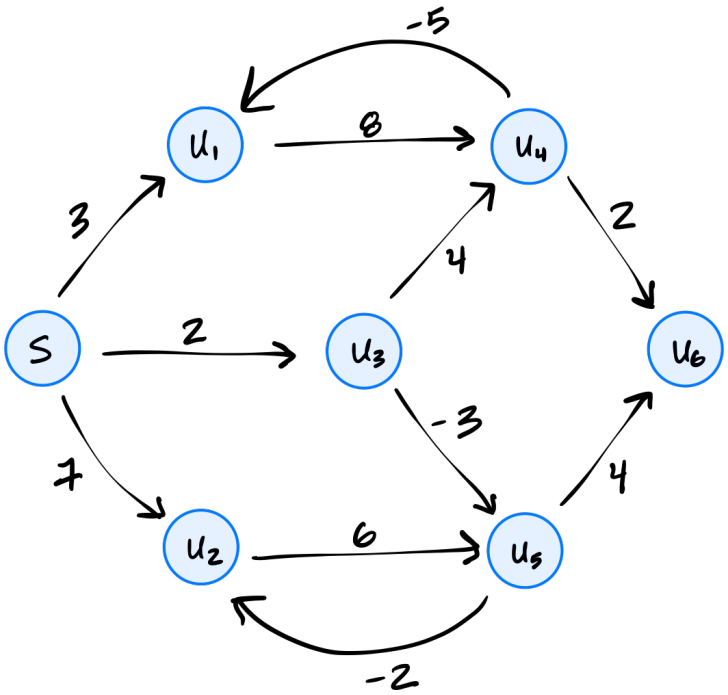


What is the fewest number of iterations of the outer loop that can possibly be performed?

Problem 23.

Recall that the Bellman-Ford algorithm keeps track of the estimated shortest path distance from the source to each node in the graph. These estimates are updated, and eventually become correct, provided that the graph has no negative cycles.

Suppose the Bellman-Ford algorithm is run on the graph below using node s as the source.



After 2 iterations of the outer loop, which of the nodes listed below are guaranteed to have correct estimated shortest path distances (no matter the order in which `graph.nodes` produces the graph's nodes)? Select all that apply.

- ☐ s
- ☐ u_1
- ☐ u_2
- ☐ u_3
- ☐ u_4
- ☐ u_5
- ☐ u_6

Problem 24.

Suppose the Bellman-Ford algorithm is run on a graph G that does not contain negative cycles. Let u be an arbitrary node in the graph, and assume that u is reachable from the source.

True or False: u's estimated distance can change multiple times during the algorithm's execution.

- ◯ True
- ◯ False

Before turning in your exam, please check that your name is on every page.

(You may use this area for scratch work.)

(You may detach and use this page for scratch work. You do not need to turn it in.)

(You may detach and use this page for scratch work. You do not need to turn it in.)