CSCI 3333 Practice Midterm #3

- Do not start until instructed to do so.
- Write your UTRGV ID only in the space provided at the top of this page.
- The midterm is closed - no books, notes, computers, cell phones, calculators, etc.
- The time allotted for the exam is 70 minutes.
- There are 7 questions worth 28 points total; each problem is worth 4 points.
- These are not the actual midterm questions.

<table>
<thead>
<tr>
<th>Problem</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td></td>
</tr>
</tbody>
</table>
**Problem 1.** Complete the following implementation of a function that returns whether the undirected graph with vertex set $V$ contains a cycle of length 3.

```cpp
bool triangle(vector<Node*> &V)
{
    for (Node* v : V)
        for (Node* vn : _____->neighs)
            for (Node* vnn : _____->neighs)
                for (Node* vnnn : _____->neighs)
                    if (vnnn == _____)
                        return _____;
    return _____;
}
```

Fill in the blanks:

The worst-case running time of `triangle` is $\Theta(\text{function of } |V|, |E|)$.

The best-case running time of `triangle` is $\Theta(\text{function of } |V|, |E|)$.

A directed graph of $|V|$ nodes containing a triangle has at least $\text{function of } |V|$ edges.

A triangle-less directed graph of $|V|$ nodes has at most $\text{function of } |V|$.
Problem 2. Complete the following implementation of the Bellman-Ford algorithm. bellman_ford should return whether the graph contains a negative-weight cycle.

```cpp
bool bellman_ford(vector<Node*> &V, Node* source, map<Node*, int> &D)
{
    D.clear();

    for (Node* v : V)
        _____[v] = numeric_limits<int>::infinity();

    D[_____] = 0;

    for (int i = 1; i < _____ .size(); ++i)
        for (Node* v : V)
            for (Node* nv : v->neighs)
                if (D[_____] + v->weights[nv] _____ D[nv])
                    D[nv] = D[_____] + v->weights[nv];

    for (Node* v : V)
        for (Node* nv : v->neighs)
            if (_____ + _____ < D[nv])
                return _____;

    return false;
}
```
Problem 3. Determine the truth of the following statements about shortest-path algorithms.

If $|E| = O(|V|)$, the running time of BFS is $O(|E|)$  □ True  □ False

If $|E| = \Theta(|V|^2)$, the running time if Dijkstra’s is $\Theta(|V|^2 \log |V|)$  □ True  □ False

Bellman-Ford is correct for disconnected input graphs.  □ True  □ False

For graphs with $|V| = \Theta(|E|)$, Dijkstra’s has lower asymptotic running time than Bellman-Ford.  □ True  □ False

Problem 4. Fill in the blanks with answers based on the graph in Figure 1.

The maximum flow from $v_1$ to $v_6$ is ______ number.

The maximum flow from $v_2$ to $v_5$ is ______ number.

The removal of the edge (__________, ________) cause the maximum flow from $v_3$ to $v_4$ to become 5.

The two distinct nodes with maximum flow between them are ______ number and ______ number.

Figure 1: The graph for Problem 4.
**Problem 5.** Draw the remaining edges of the weighted connected undirected graph below so that it has the following properties:

- All edges have positive integer weights.
- Any BFS from $v_1$ reaches $v_6$ last.
- $d(v_1, v_4) = d(v_1, v_5) = 4$.

![Figure 2: The (partially drawn) graph for Problem 5.](image)
Problem 6. Fill in the blanks with answers based on the graphs in Figure 3.

The number of edges of an MST is \( \text{number} \) and the weight of an MST is \( \text{number} \).

There are \( \text{number} \) distinct minimum spanning trees.

Removing the edge \( (\text{node}, \text{node}) \) increases the weight of an MST.

Figure 3: The graph for Problem 6.
Problem 7. Complete the labeling of the nodes in the graphs below according to the order they are “visited” (removed from the queue) during the search specified in the caption.

Figure 4: Breadth-first search ordering (Problem 7).

Figure 5: Depth-first search ordering (Problem 7).

Figure 6: Breadth-first search ordering (Problem 7).