CSCI 3333 Practice Midterm #1

- 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 the number of elements in a BST (containing strings) that come lexicographically after s.

```c
int after(Node* root, string s)
{
    if (root->s < s)
        return after(root-> _____ , s);
    int lc = after(root->left, _____ );
    int rc = after(root->right, _____ );
    if (root->s _____ s)
        return lc + rc + _____;
    return lc + rc;
}
```

**Problem 2.** Determine the truth of the following statements about binary trees with \( n \) nodes.

- Every binary tree is a BST. \( \square \) True \( \square \) False
- A binary tree can have height \( \lfloor \log_2(n) \rfloor \). \( \square \) True \( \square \) False
- Inserting into a BST takes \( \Theta(\log(n)) \) best-case time. \( \square \) True \( \square \) False
- In a BST, the node with the smallest value is always a leaf. \( \square \) True \( \square \) False
Problem 3. Determine the truth of the following statements about AVL trees.

Every AVL tree is balanced. □ True □ False

The erase operation on an AVL tree with \( n \) nodes has \( \Theta(\log(n)) \) worst-case running time. □ True □ False

The maximum number of rotations done in an AVL tree insert is \( \Theta(1) \). □ True □ False

The minimum number of nodes in an AVL tree of height 5 is 20. □ True □ False

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

If insert(22) were called, the right child of _________ would change to _________ during the first rotation.

If insert(2) were called, _________ rotations would occur during the call.

If insert was called 4 times, at least _________ rotations would occur during the calls.

If insert was called twice, up to _________ rotations would occur during the calls.

![AVL Tree Diagram](image)

Figure 1: The AVL tree for Problem 4.
Problem 5. Determine the truth of the following statements about the running times of linear probing hash tables with load factor $\alpha$.

If $\alpha > 1/2$, then a collision has occurred.  \hspace{1cm} \square \text{True} \hspace{1cm} \square \text{False}

The worst-case running time of $\text{insert}$ is $\Theta(\alpha)$.  \hspace{1cm} \square \text{True} \hspace{1cm} \square \text{False}

The best-case running time of $\text{search}$ is $\Theta(1/(1 - \alpha))$.  \hspace{1cm} \square \text{True} \hspace{1cm} \square \text{False}

The worst-case running time of $\text{search}$ is $\Theta(1/(1 - \alpha))$.  \hspace{1cm} \square \text{True} \hspace{1cm} \square \text{False}

Problem 6. Suppose you’re given a class $\text{HashTable}$ implementing a chaining hash table and containing the following instance variables:

- An array $A$ containing $\text{list<int>}$s.
- A capacity variable storing the length of $A$.

Note: there is no instance variable that stores how many elements are in the table.

Write a method that returns the load factor of the hash table and has the following prototype:

```
int HashTable :: load_factor()
```

Give the worst-case and best-case running time of this method in terms of $n$ (the number of elements in the table) and $L$ (the length of $A$).
Problem 7. Suppose you’re given a class `Heap` implementing a min-heap and containing the following instance variables:

- An array `A` containing floats.
- A `capacity` variable storing the length of `A`.
- A `count` variable storing the number of elements in the heap.

Write a method that returns the largest element in the heap and has the following prototype:

```java
float Heap :: max()
```

Give the worst-case and best-case running times of this method in terms of `n` (the number of elements in the heap). *Aim for O(1) best-case.*