CS 240: Data Structures and Data Management	**Spring 2026**

Midterm Practice Problem

Note: This is a sample of problems designed to help prepare for the midterm exam. These problems do *not* encompass the entire coverage of the exam, and should not be used as a reference for its content.

1. **True or False**

   (a) The midterm covers up to and including AVL tree.

   (b) If $T_1(n) \in \Omega(f(n))$ and $T_2(n) \in O(g(n))$, then $\frac{T_1(n)}{T_2(n)} \in \Omega(\frac{f(n)}{g(n)})$

   (c) If $\lim_{n\to\infty} \frac{f(n)}{g(n)} = e^{42}$, then $f(n) \in \Theta(g(n))$

   (d) If $f(n) \in \Theta(g(n))$, then $\lim_{n\to\infty} \frac{f(n)}{g(n)} = L$ where $0 < L < \infty$

   (e) If at least one rotation was performed during AVL-delete, then the height of the AVL Tree after deletion is strictly less than the height of the AVL Tree before deletion.

   (f) All heaps satisfy AVL tree's height-balance requirement

   (g) A binary search tree with n leaves must have height in O(n)

   (h) The average-case and expected-case run-time of an algorithm must always be the same

2. **Order Notation and Recurrence Relation**

   (a) Show that $3n^2 - 8n + 2 \in \Theta(n^2)$ from first principles.

   (b) Prove from first principle that $14n + 22$ is $o(n \log n)$

   (c) Prove from first principle that $n \in \omega(2^{\sqrt{\log n}})$

   (d) Given $T(1) = 1$, resolve $T(n) = T(\frac{3n}{4}) + n$.

   (e) Disprove the following statement - if $f(n) \in o(n \log n)$, then $f(n) \in O(n)$

   (f) Prove or disprove the following claim. If $f(n) \in O(h_1(n))$ and $g(n) \in \omega(h_2(n))$, then $\frac{f(n)}{g(n)} \in o\left(\frac{h_1(n)}{h_2(n)}\right)$, assuming $f(n), g(n), h_1(n)$ are all positive $\forall n \geq 0$. You should prove the statement from first principles or provide a counter example.

3. **Pseudo-code Analysis**
   Analyze following pieces of pseudo-code and give a tight bound on the running time as a function of n.

   (a) Analyze the following piece of pseudo-code and give a tight (Θ) bound on the running time as a function of n.

   ```
   i = 2
   x = 0
   while (i < n):
       for j = 1 to n:
           for k = 1 to j:
               x = x + 1
       i = i * i
   ```

   (b) Give a tight (Θ) bound for the expected runtime of the following algorithm.

```
ArraAlg(A, n, k)
    // n = A.size()
    // A is a permuation of [0, ..., n-1]
    // k is in the set {0, ..., n-1}
    i = random(n)
    if A[i] == k then
        return i
    for j = 0 to n-1
        print("a")
    return ArrayAlg(A, n, k)
```

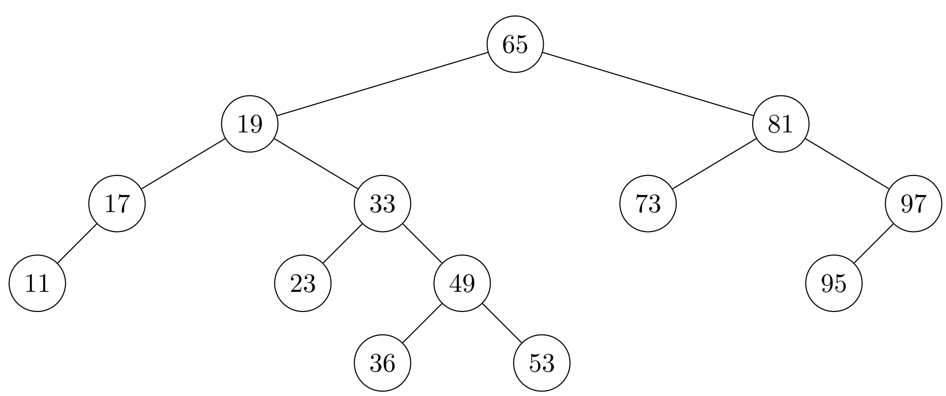
(c) Let A and B be two bi-strings of length n (modelled here as arrays where each entry is 0 or 1). A `string-compare` tests whether A is smaller, larger, or the same as B and works as follows:

```
str-cmp(A, B, n)
    for i = 0; i < n: i++ do
        if (A[i] < B[i]) then return "A is smaller"
        if (A[i] > B[i]) then return "A is bigger"
    return "They are equal"
```

Show that the average-case run-time of `str-cmp` is in $O(1)$. You may use without proof that $\sum_{i\geq 0} \frac{i}{2^i} \in O(1)$.

4. **Priority-Queue**
A student designed a data structure and named it an `almost-priority-queue`. This data structure allows two operations: `insert` and `extract_almost_Max`, where `extract_almost_Max` outputs either the largest priority or the second largest priority item and also, deletes the item it outputs. Also, `extract_almost_Max` does not tell you whether it extracted the largest or second largest priority item. In case the data structure has only one element, `extract_almost_Max` extracts that element. The student claims that the worst case running time of both `insert` and `extract_almost_Max` is $o(\log n)$. Prove that the student has made a mistake in the running time analysis of their data structure.

5. **Basics of AVL Tree**
Consider following AVL tree.



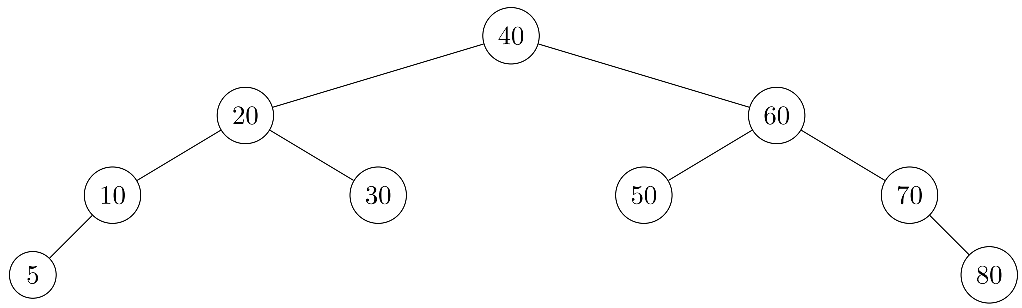
(a) Fill out height factor of each node. For example, node 33 will have height factor of 2.

(b) Fill out balance factor of each node. For example, node 33 will have height factor of 1.

(c) Insert 61 into above AVL tree.

(d) Perform `delete(73)` on resulting tree (i.e. above tree after inserting 61)

6. **Rotations in AVL tree**
Consider the following AVL tree.



Suppose we are trying to insert x into above AVL tree.

(a) What value of x would cause us to perform a left rotation?

(b) What value of x would cause us to perform a right rotation?

(c) What value of x would cause us to perform a double right rotation?

(d) What value of x would cause us to perform a double left rotation?

7. **Epsilon**
Let $0 < \epsilon < 1$. Suppose that we have an array A of n items such that the first $n - n^\epsilon$ items are sorted. Describe an $O(n)$ time algorithm to sort A.

8. **Numbers in Range**
We have an array A of n non-negative integers such that each integer is less than k. Give an $O(n + k)$ time preprocessing algorithm such that queries of the form "how many integers are there in A that are in the range $[a, b]$?" can be answered in $O(1)$ time. Note that a and b are not fixed; they are parameters given to the query algorithm.

9. **Removing Duplicates From An Array**
Assuming that we are in comparison based model. Given an array that may contain duiplicate elements, what is...

(a) The lower bound for the worst case cost of returning sorted array minus all the duplicate elements?

(b) The lower bound for the worst case cost of returning the array without duplicates, that is not necessarily sorted?