CS 240: Data Structures and Data Management **Spring 2023**

Midterm Practice Problem

1. **True or False**
   (a) The midterm covers 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) All heaps satisfy AVL tree's height-balance requirement
   (e) Worst-case runtime of `QuickSelect` can be avoided by always choosing the median elements as the pivot
   (f) A binary search tree with n leaves must have height in O(n)
   (g) 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.
   (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 $o(nlogn)$
   (c) Given $T(1) = 1$, resolve $T(n) = T(\frac{3n}{4}) + n$.
   (d) Disprove following statement - if $f(n) \in o(nlogn)$, then $f(n) \in O(n)$

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)
```
s = 100;
i = 2;
while (i < n) {
    for j = 1 to n do
        s = s + 10;
    }
    i = i * i;
}
```

   (b) For this algorithm, you may assume that n is power of 3

Algorithm 1: STOOGE(A, i, j)

Input: Array A of size n, index i (initially 0), index j (initially $n-1$)
Output: No output but the subarray $A[i \dots j]$ will be sorted

```
if A[j] < A[i] then
    SWAP(A[i], A[j])
end
if j − i + 1 > 2 then
    t ← ⌊(j−i+1)/3⌋;
    STOOGE(A, i, j − t);
    STOOGE(A, i + t, j);
    STOOGE(A, i, j − t);
end
```

4. **Runtime Cases**
After midterm grading is complete, Taebin wants to make sure all the grades are okay, and have them listed in sorted order. He asks Prashanth or Robert to get help with this task. Prashanth or Robert have many sorting algorithms to choose from, so their decision is based on their mood. Their moods fluctuate while they check assignment grades, and eventually depends on the final student's grade.

   Assignment grades in array A are integers ranging from 0 to 100. There is exactly one student with a grade of 0, and exactly one student with a grade of 100.

   The function `verify(A)` runs in $\Theta(n)$ time and returns:

   - +1, if the final student of A scored 100.
   - -1, if the final student of A scored 0.
   - 0, otherwise.

```
PrashanthSort (A):
    mood := verify(A)
    if mood = +1
        MergeSort(A)
    else if mood = -1
        SelectionSort(A)
    else MSDRadixSort(A, R = 10, m = 3)


RobertSort (A):
    mood := verify(A)
    if mood = +1
        SelectionSort(A)
    else if mood = -1
        LSDRadixSort(A, R = 10, m = 3)
    else MergeSort(A)
```

   (a) What is the best-case, worst-case, and average-case runtime for `PrashanthSort`?
   (b) What is the best-case, worst-case, and average-case runtime for `RoberSort`?
   (c) Taebin's decision, which we refer to as `TaebinSort`, is to flip a coin. If it flips heads, he gets help from Prashanth (`PrashanthSort`). Otherwise, if it flips tails, he gets help from Robert (`RobertSort`). What is the expected runtime (worst-case expected) for `TaebinSort`?

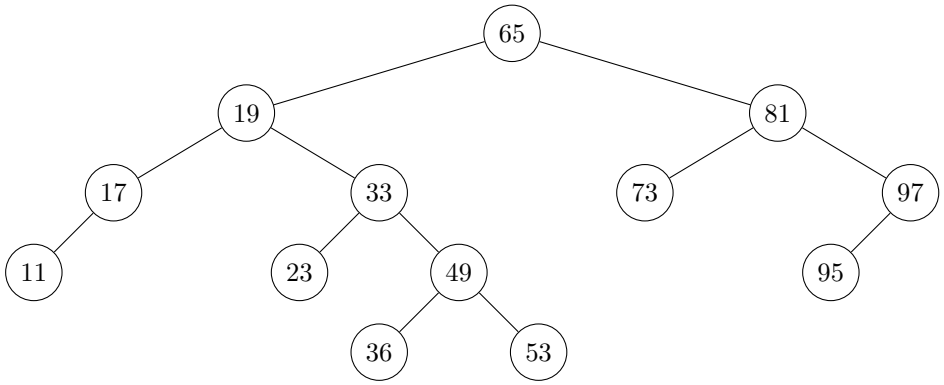
5. **Analysis**
Suppose A is an array containing n distinct elements. In addition, assume each element is in between 1 and n, inclusive. Analyze this pseudo-code to determine a tight bound on the average number of question mark (?) that are printed, rather than a runtime. You may assume n is divisible by 2.

```
????(A, n)
    count = 1
    for i = 1 to n-1
        if A[i] % A[0] == 0
            count++
    for i = 0 to count
        print("?")
```

6. **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.

7. **Stack Using Heap**
   How would you implement a stack using a heap? Analyze the complexity of the push and pop operations.

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



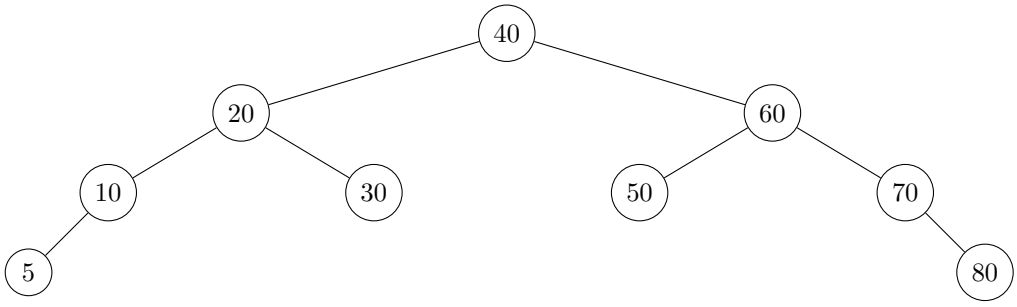
   (a) Fill out height factor of each node. For example, node 33 will have height factor of 2.

   (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)

9. **Rotations in AVL tree**
   Consider following AVL tree.



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

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

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

10. **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?