HW0: Review

COMP 221 — Suhas Arehalli

Spring 2026

Instructions

This assignment's goal is to help you recall a handful of concepts from the course's prerequisites that will be critical throughout the semester. While many of these concepts will come up as we cover background in the first week, some may require you to consult your notes from prior courses.

Keep in mind the following when completing the assignment:

1. Submissions will be accepted as typeset pdfs through Moodle. I strongly recommend using LaTeX (perhaps through a tool like Overleaf), which is what I use to write these instructions.
2. Keep in mind that this work is to be done individually — do not share or look at other students work directly. However, consulting your classmates, a preceptor, or an instructor for guidance (an idea or approach one might try) is perfectly acceptable, but **you must indicate in your submission the names of all people who have helped you.** Give credit where credit is due!
3. **Proofread your work before submission**. Like all writing, it's easy to make mistakes while you're putting together a first draft — you're doing a lot in that first pass! Luckily, you can take a second pass to revise, dedicated to making sure your writing is clear and your argument is free of holes.

Questions

1. *(Discrete Math)* Recall that an integer n is odd iff there exists an integer k such that $n = 2k+1$ and is even iff there exists an integer k such that $n = 2k$. Prove that $n(n+1)$ must be even.
2. *(Discrete Math)* Prove (by induction) that
$$\sum_{i=1}^{n} i = \frac{1}{2}n(n+1).$$
3. *(Data Structures)* Determine the growth function for the algorithm 1 in terms of n. For simplicity, you need only have the growth function count the number of times x is incremented.
4. *(Data Structures)* Provide the worst-case time complexity for algorithm 1 in Big-O notation. This should be *best possible* (i.e., slowest growing) Big-O that you can provide.[1]

[1] i.e., don't just write $O(n!)$ and call it a day.

Algorithm 1 Mysterious Algorithm

```
x ← 0
for i ← 1 to n do
    for j ← 1 to i do
        x ← x + 1
    end for
end for
return x
```

5. *(Data Structures)* Consider the following pseudocode for a purported sorting algorithm. Does this algorithm work? Why? What is it's worst-case time complexity, assuming we use a type of self-balancing BST you saw in Data Structures?

Algorithm 2 Sort Pseudocode A

```
function SORT(Array A)
    Insert all x ∈ A to a self-balancing BST T
    return an in-order traversal of T
end function
```

6. *(Discrete Math/Data Structures)* Consider a Graph $G = (V, E)$ with no self-edges. Recall that the degree of a vertex $v \in V$, notated $d(v)$, is the number of edges that are incident to/connect to v. Prove that $\sum_{v \in V} d(v)$ is even. Further, prove that the sum of the degrees of all odd-degree vertices, $\sum_{v \in V, d(v) odd} d(v)$, is even.

   HINT: Do this via induction on the number of edges.