Recitation 3: Big-O and Computational Complexity

1 Big-O Notation

True or False? Give a brief justification for each.

1. $n \log n \in O(n)$
2. $10^9 n \in \Theta(n)$
3. $\log(n^4) = O(\log n)$
4. $n^6 \cdot 2^{-n} \in O(1)$
5. $\log n \in \Omega(\sqrt{n})$.
6. $2^{\sqrt{n}} \in O(n^3)$
7. $3^n \in \Omega(n!)$
8. $n^k \in O(k^n)$ for $k \geq 2$

Formally prove or disprove the following statements:

1. If $f \in O(g)$ and $g \in O(h)$ then $f \in O(h)$.
2. $n^2 + 3/n \in O(n^2)$
3. $n^2 + 3n \log n \in O(n^2)$.
4. $n \log n \in O(n^{1+\epsilon})$ for every $\epsilon > 0$.

2 Analyzing runtime complexity

1. Analyze the running time of the following program:

```
v = 0
for i = 1 to n do
    for j = 1 to n do
        v = i · j^2
    end for
end for
```

2. Suppose the function $F(i, j)$ takes $\Theta(i)$ time to execute. Analyze the runtime of the following program, which has input n:

```
for i = 1 to n do
    for j = i to n do
        Call F(i, j)
    end for
end for
```

Hint: $\sum_{i=1}^{n} i^2 = n(n+1)(2n+1)/6$.

3. Analyze the runtime of the following binary search algorithm. Assume that it is first called as BinSearch(A, 0, n, x). Here A is the (sorted) array to be searched and the goal is to return the index of x of it's found in A, and -1 otherwise.

```
procedure BinSearch(A, ℓ, r, x)
    if ℓ > r then
        return -1
    end if
    m ← ℓ + ⌊(r − ℓ)/2⌋
    if A[m] = x then
        return m
    else if A[m] < x then
        return BinSearch(A, m + 1, r, x)
    else
        return BinSearch(A, ℓ, m − 1, x)
    end if
end procedure
```

4. Analyze the runtime of the following sorting algorithm. Assume the Merge(A_1, A_2) procedure takes time $O(n_1 + n_2)$ where A_1 has length n_1 and A_2 has length n_2.

```
procedure MergeSort(A, ℓ, r)
    if ℓ ≥ r then
        return
    end if
    m ← ℓ + ⌊(r − ℓ)/2⌋
    A_1 ← MergeSort(A, ℓ, m)
    A_2 ← MergeSort(A, m + 1, r)
    Merge(A_1, A_2)
end procedure
```