CSCE 411-501 ✵ Fall 2026
Proofs Practice

1. Simplify $7^{\log_9 n^2}$ (i.e. express it in the form n^c, where c does not depend on n).

Solution:

$$7^{\log_9 n^2} = 7^{2\log_9 n} = 7^{2\log_7(n)\log_9(7)} = (7^{\log_7 n})^{2\log_9(7)} = n^{2\log_9 7}$$

2. Prove that the algorithm below correctly sorts the input array A:

```
SelectionSort(A[1,...,n])
  For i from 1 to n:
      k ← i
      For j from i+1 to n:
          If A[j] < A[k]:
              k ← j
      Swap A[i] with A[k]
```

Solution: Let A be an arbitrary array of numbers. We will prove, by induction on i, that after i iterations of the algorithm's outer for loop, $A[1,\ldots,i]$ contains the i smallest values in A in sorted order.

- Base case: Suppose $i = 0$. Vacuously, after 0 iterations of the loop, the 0 smallest values in the array are stored at the beginning in sorted order.
- Induction hypothesis: Suppose that $\forall\, k \in \mathbb{Z}_{\geq 0}$ with $k < i$, after k iterations of the for loop the smallest k elements of A are stored in positions 1 through k in sorted order.
- Induction Step: We will prove that after i iterations, the smallest i elements are stored in positions 1 through i in sorted order. By our induction hypothesis, after iteration $i-1$ of the loop, $A[1,\ldots,i-1]$ contained the $i-1$ smallest elements of the array in sorted order.

  We observe that in iteration i of the loop, the algorithm iterates through $A[i,\ldots,n]$, finds the minimum element of this subarray, and swaps it to position i. Since $A[1,\ldots,i-1]$ already contained the $i-1$ smallest elements of A, the smallest element of $A[i,\ldots,n]$ will be the i'th smallest element of the overall array. Therefore, since this loop iteration does not modify $A[1,\ldots,i-1]$, at the end of the i'th iteration we have that $A[1,\ldots,i]$ contains the i smallest values in the array and they are in sorted order.

We have now shown that after n iterations of the outer for loop (i.e. at the end of the algorithm), $A[1,\ldots,n]$ contains the smallest n elements of the array and they are in sorted order. Therefore, SelectionSort correctly sorts A.

3. We define a set of strings S recursively as follows:
   - $\varepsilon \in S$
   - $1 \in S$
   - $\forall\, x \in S$, $0x \in S$
   - $\forall\, x \in S$, $10x \in S$

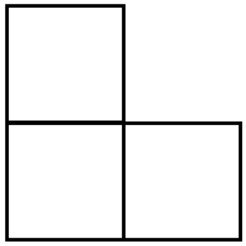
   Prove that S is the set of binary strings which do not contain two consecutive ones (Hint: How do we prove set equality?).

Solution: We want to show that for the set $A :=$ set of binary strings which do not contain two consecutive ones, that $S \subseteq A \wedge A \subseteq S$, since this implies $S = A$

- $S \subseteq A$
  Let $s \in S$ be arbitrary. We need to show $s_0 \in A$. We prove this by induction on $|s|$.
  - Base Cases: If $|s| \leq 1$ then $s = \varepsilon, s = 0$, or $s = 1$. All of these are in A.
  - Induction Hypothesis: Suppose that for all $t \in S$ with $|t| < |s|$, $t \in A$.
  - Induction Step: Since we are not in the base case, $|s| \geq 2$. Only the last two rules can generate strings of length ≥ 2, so we have two cases:
    * Suppose $s = 0x$ for some $x \in S$. Then $|x| = |s| - 1 < |s|$ so by our induction hypothesis $x \in A$, and therefore x does not contain two consecutive 1s. Adding a 0 to the front cannot create a pair of consecutive 1s, so $s \in A$.
    * Suppose $s = 10x$ for some $x \in S$. Then $|x| = |s| - 2 < |s|$ so by our induction hypothesis $x \in A$, and therefore x does not contain two consecutive 1s. When we add 10 to the front we cannot create a new pair of consecutive 1s, since the 1 we are adding is separated from every element of x by a 0. Therefore $s \in A$.

- $A \subseteq S$
  Let $s \in A$ be arbitrary. We need to show $s \in S$. We prove this by induction on $|s|$.
  - Base Case: If $|s| \leq 1$ then $s = \varepsilon, s = 0$, or $s = 1$. All of these are in S.
  - Induction Hypothesis: Suppose that for all $t \in A$ with $|t| < |s|$, $t \in S$.
  - Induction Step: Since we are not in the base case, $|s| \geq 2$. We consider two cases:
    * If the first character of s is a 0 then s is in the form $0x$, where $x \in \{0,1\}^{|s|-1}$. Since s does not contain any consecutive 1s, x (which is a substring of s) cannot either, so $x \in A$ and $|x| < |s|$. Therefore by our IH, $x \in S$, and so by the third rule, $s \in S$.
    * If the first character of s is a 1, then the second character of s must be a 0 (since $s \in A$, we know it does not contain two consecutive 1s). Therefore s is in the form $10x$ where $x \in \{0,1\}^{|s|-2}$. Since s does not contain any consecutive 1s, x (which is a substring of s) cannot either, so $x \in A$ and $|x| < |s|$. Therefore by our IH, $x \in S$, and so by the fourth rule, $s \in S$.

4. An L-triomino is a shape consisting of 3 squares in the arrangement shown below. Prove that for every positive integer n, it is possible to place L-triominos onto a 2^n by 2^n checkerboard such that each triomino covers three distinct squares of the board (no triomino hangs of the edge and no two overlap) and all but one square of the board is covered.



Solution: We will prove the strong claim that it is always possible to peform the tiling such that the uncovered square is in the corner. That is, we will prove that for every $n \in \mathbb{Z}^+$ it is possible to place L-triominos onto a 2^n by 2^n checkerboard such that each triomino covers three distinct squares of the board and all squares except a single corner square are covered.

We prove this by induction on n.

- Base case: Suppose $n = 1$. Then we can simply place a single triomino on the 2 by 2 checkerboard covering 3 of the corners and leaving one corner square uncovered.

- Induction Hypothesis: Suppose that for all positive integers $k < n$, it is possible to tile a 2^k by 2^k checkerboard with L-triominos leaving only a single corner square uncovered.

- Induction step: We need to show that we can tile a 2^n by 2^n checkerboard leaving only a single corner square uncovered.

  Divide the 2^n by 2^n checkerboard into four quadrants. Each of these quadrants is a 2^{n-1} by 2^{n-1} checkerboard, so by the induction hypothesis we can tile each of them, leaving only a single corner of each uncovered. For three of the quadrants (chosen arbitrarily) rotate the tiling so that the uncovered corner is in the middle of the 2^n by 2^n board. For the fourth quadrant, rotate the tiling so that the uncovered corner is in the corner of the 2^n by 2^n board. Finally, add one additional L-triomino to cover the 3 uncovered squares in the middle of the board (one from each of the quadrants where we left a middle corner uncovered). We have now obtained a tiling of the 2^n by 2^n board which covers all squares except a single corner.