Note: Please attempt all problems. You can ask me for hints, but please avoid the temptation to just search the internet for solutions! Working through the problems by yourself will help you understand the material better, and help prepare you for the quizzes.

1. Prove that the following problems are in \( P \), by outlining a polytime algorithm to solve them (and arguing the correctness of your algorithm).

   (a) A Horn clause is a disjunction of literals that contains at most one positive literal (and may contain an arbitrary number of negative literals). A Horn formula is a conjunction of Horn clauses. (Equivalently, a Horn formula is a propositional formula that can be written as a conjunction of implications of the form \( x_1 \land x_2 \land \cdots \land x_k \rightarrow x_{k+1} \). Horn formulas are the basis for the language PROLOG.) Show that deciding if a given Horn formula is satisfiable is in \( P \).

   (b) Show that 2-SAT is in \( P \), by using the fact that the Resolution proof system (defined in class) is sound and complete.

2. Give a polytime search-to-decision reduction for the decision problem 3-COL = \( \{ G \mid G \text{ is a 3-colorable undirected graph} \} \), whose search version is to find a 3-coloring, if it exists.

3. Let \( \text{Size}(s) \) denote the class of languages that can be decided by a family of Boolean circuits of size \( s(n) \). While it is not known whether there is a language in \( \text{PH} \) that has superpolynomial circuit complexity, it is known that, for every fixed \( k \), \( \text{PH} \not\subset \text{Size}(n^k) \). You are asked to prove this, by showing the following.

   **Theorem** For every constant \( k \), there is a language \( L_k \in \Sigma^p_2 \) such that \( L_k \not\in \text{Size}(n^k) \).

   Prove this theorem by following these steps.

   (a) Using the fact that for every sufficiently large \( n \) there is a Boolean \( n \)-variable function of circuit complexity at least \( 2^n/n \), show that for every \( k \), there is a Boolean \( n \)-variable function \( f_n \) that requires circuit size at least \( n^k \) and such that \( f_n \) depends on only \( O(\log n) \) of its \( n \) variables. Conclude that such a \( f_n \) can be described using only \( \text{poly}(n) \) bits.
(b) Using a constant number of alternating quantifiers, express the property of being the lexicographically first poly(n)-bit string that describes a Boolean function $f_n$ of circuit complexity at least $n^k$. Conclude that $\text{PH}$ contains a language $L_k$ of circuit complexity at least $n^k$.

(c) Use the Karp-Lipton theorem to argue that, for every $k$, even $\Sigma^p_2$ contains some language of circuit complexity at least $n^k$.

4. Show that $\text{MAJORITY}_n$ is in $\text{NC}^1$ with circuit size $O(n)$ (and depth $O(\log n)$). (Recall that $\text{MAJORITY}_n(x_1, \ldots, x_n)$ is 1 iff $\sum_{i=1}^n x_i \geq n/2$.) (HINT: Use ideas of the construction of circuits for adding $n$ $n$-bit numbers we did in class, via a “3-for-2” trick.)

5. Prove that Parity requires $\Omega(2^n)$ size $\text{AC}^0$ circuits of depth 2 (i.e., both CNFs and DNFs).

6. Show that Multiplication is not computable in $\text{AC}^0$ (i.e., circuits of polynomial size, and arbitrary constant depth). Here Multiplication is the problem: given two $n$-bit integers, compute their product; the circuit for Multiplication will have $2n$ output gates. You need to use the fact that Parity is not in $\text{AC}^0$, where Parity is the Boolean function mapping an $n$-bit string $w$ into bit $b$, where $b = 0$ if $w$ contains an even number of 1’s, and 1 otherwise. (Hint: Give a reduction showing that if Multiplication were computable in $\text{AC}^0$, then so would be Parity.)