1. **(Interactive proofs vs. instance checkers.)** Suppose languages $L$ and $\overline{L}$ have polynomial-round interactive proofs in which Merlin's strategy is implementable in $P^L$. Show that $L$ has an instance checker. You may use a slightly weaker definition of “instance checker” wherein, if the provided oracle $C$ actually computes $L$ exactly, the checker only has to output the correct answer about $x \in L$ with high probability (rather than with probability 1).

2. **(Derandomization implies circuit lower bounds.)** Suppose you wanted to prove $\text{BPP} = \text{P}$. Well, you'd better be able to at least prove $\text{coRP} = \text{P}$. And hence you'd better be able to at least prove that the PIT problem (Polynomial Identity Testing, which we know is in $\text{coRP}$) is in $\text{P}$. And hence you’d better be able to at least prove that it’s in $\text{NSUBEXP} := \bigcap_{\epsilon > 0} \text{NTIME}(2^{n^\epsilon})$. In this problem, you'll show this implies that you’d better be able to prove superpolynomial circuit lower bounds.

   
   In this problem, let $\text{AlgP}^0 / \text{poly}$ denote the class of all polynomial-degree families computable by polynomial-size algebraic circuits using $+, -, \times$ over $\mathbb{Z}$, where the only constants allowed are 0 and 1 (equivalently, where the constants must be of $\text{poly}(n)$ bit-length).

   
   (a) Show that if $\text{PERMANENT} \in \text{AlgP}^0 / \text{poly}$ and $\text{PIT} \in \text{NSUBEXP}$, then $\Sigma_2^P \subseteq \text{NSUBEXP}$.

   (You can definitely use Valiant’s Theorem on $\#P$-completeness of $\text{PERMANENT}_{0,1}$. You can also use Toda’s 1st and 2nd Theorems if you like, though you don’t need them.)

   (b) Show that if, furthermore, $\text{NEXP} \subseteq \text{P} / \text{poly}$, then $\Sigma_2^P \subseteq \text{P} \subseteq \text{SIZE}(n^c)$ for some constant $c$. (Here $\text{NE} = \text{NTIME}(2^{O(n)})$.)

   (c) Deduce that

   $\text{PIT} \in \text{NSUBEXP} \implies \left( \text{PERMANENT} \notin \text{AlgP}^0 / \text{poly} \lor \text{NEXP} \notin \text{P} / \text{poly} \right)$.

3. **(Worst-case hardness to slight hardness-on-average for EXP.)** Suppose that $L \in \text{EXP}$ but $L$ requires superpolynomial-size circuits; more precisely, for all $c$ and all sufficiently large $n$ it holds that there is no Boolean circuit of size $n^c$ computing $L_n : \{0,1\}^n \to \{0,1\}$, the indicator function for presence in $L \cap \{0,1\}^n$.

   (a) Show that there is a language $L' \in \text{E} := \text{TIME}(2^{O(n)})$ with the same property.

   (b) Let $p$ stand for the first prime larger than $n + 1$ (this can certainly be deterministically computed in $\text{poly}(n)$ time, as we’ll have $p < 2n$) and write $\mathbb{Z}_p$ for the field of integers modulo $p$. Show that there is a multilinear polynomial $f_n : \mathbb{Z}_p^n \to \mathbb{Z}_p$, agreeing with $L'_n$ on all inputs in $\{0,1\}^n$, such that the family of functions $(f_n)$ can be computed in $2^{O(n)}$ time.

   (c) Show that for every polynomial-size circuit family $(C_n)$ (where $C_n$ has $n(\log n + 1)$ inputs and $\log n + 1$ output$[1]$)

   $\Pr_{x \sim \mathbb{Z}_p^n} [C_n(x) = f_n(x)] < 1 - \frac{1}{3n}$.

\[1\] Here $\log n + 1$ is enough to encode an element of $\mathbb{Z}_p$; I’m too lazy to put ceilings/floors in the right spots here, and you may be equally lazy about this point.
Define a decision problem (language) $H$ as follows: on input $x \in \mathbb{Z}_p^n$ and integer $0 \leq j \leq \log n$, output the $j$th bit of $f_n(x)$. Show that $H \in \mathsf{E}$, and that for every polynomial-size circuit family $(D_n)$ it holds that

$$
\Pr_{x \sim \mathbb{Z}_p^n, \quad j \sim \{0, \ldots, \log n\}} \left[ D_n'(x, j) = H(x, j) \right] < 1 - \frac{1}{O(n \log n)}
$$

(where $n' = n(\log n + 1) + \log \log n$).

Remark: Thus from a language in $\mathsf{EXP}$ that is hard for polynomial-size circuits in the worst case, we may construct a language in $\mathsf{E}$ that is slightly hard-on-average for polynomial-size circuits, where “slightly” involves error at least $\frac{1}{O(n \log n)}$ on inputs of length $n'$.

(Incredibly minor notes: Strictly speaking, we have not quite shown hardness-on-average with respect to the purely uniform distribution on inputs, because of the issue of how exactly to encode the pair $\langle x, j \rangle$ by a single string. Also, strictly speaking, $H$ might be trivial for some input lengths (those not of the appropriate form $n(\log n + 1) + \log \log n$), and we’d rather have it hard for circuits at almost all input lengths. Both issues are easy and boring to fix.)