Problem 1. (Approximate Counting)

1. Prove that a fully polynomial randomized approximation scheme for #Matchings implies a fully polynomial almost-uniform sampler for Matchings. (This is the converse of what we showed in class.)

2. Show that approximating #Independent Sets to within any constant factor is NP-hard. (In contrast, there are a fully polynomial randomized approximation schemes known for #Perfect Matchings and #Matchings.)

Problem 2. (Graph Isomorphism) Since Graph Isomorphism is in NP, it has a trivial interactive proof where the prover simply sends the NP witness (the isomorphism) to the verifier. Here, you will see how using randomness and interaction, we can obtain a different interactive proof with the additional advantage of being “zero knowledge” — the verifier learns nothing other than the fact that the graphs are isomorphic.

1. Show that the following protocol is an interactive proof for Graph Isomorphism.

 Protocol $(P, V)(G_0, G_1)$, where $G_0$ and $G_1$ are both graphs on vertex set $[n]$:

(a) $P$ finds (or gets as an auxiliary input) a permutation $\pi \in S_n$ such that $\pi(G_0) = G_1$,

(b) $P$ chooses a uniformly random permutation $\rho \overset{R}{\leftarrow} S_n$, sets $H = \rho(G_1)$, and sends $H$ to $V$.

(c) $V$ flips a coin $b \overset{R}{\leftarrow} \{0, 1\}$, and sends $b$ to $P$.

(d) If $b = 0$, $P$ sends $\psi = \rho \circ \pi$ to $V$. If $b = 1$, $P$ sends $\psi = \rho$ to $V$.

(e) $V$ accepts if $\psi(G_b) = H$. 
2. Show that the above protocol is zero knowledge in the sense that when \((G_0, G_1) \in GI\), everything \(V\) sees, it could have generated efficiently on its own. That is, there is a probabilistic polynomial-time “simulator” \(S\) such that when \((G_0, G_1) \in GI\), the output distribution \(S(G_0, G_1)\) is identical to the distribution of \(V\)’s view of the protocol \((P, V)(G_0, G_1)\) (namely the triple \((\rho, b, \psi)\)).

**Problem 3. (Random self-reducibility)** A function \(f : \{0, 1\}^* \to \{0, 1\}^*\) is random self-reducible under a sequence \(D_n\) of distributions (where \(D_n\) is a distribution on \(\{0, 1\}^n\)) if there is a probabilistic polynomial-time oracle algorithm \(M\) such that for every \(n\) and every \(x \in \{0, 1\}^n\),

1. \(M^I(x) = f(x)\), and
2. The oracle queries made by \(M^I(x)\) are each distributed according to \(D_n\).

If in addition \(M\)’s oracle calls are nonadaptive, we say that \(f\) is nonadaptively random self-reducible.

1. Show that if \(f\) is random self-reducible under \(D_n\) and \(f \notin BPP\), then there is a polynomial \(p(n)\) such that \(f\) is not \((1 - 1/p(n))\)-easy under \(D_n\).
2. Explain why there are \#P-complete, PSPACE-complete, and EXP-complete problems that are randomly self-reducible under the uniform distribution \(U_n\).
3. Show that if there were a nonadaptively random self-reducible NP-complete problem (under any distribution \(D_n\)), then \(coNP \subseteq prAM/poly\). The latter class is \(prAM\) with polynomial advice. We use the promise class rather than the language class for technical reasons that you need not worry about. (Hint: run \(M\) many times, take as advice the quantity \(Pr[D_n \in L]\).
4. (*) Show that if \(coNP \subseteq prAM/poly\), then the \(PH\) collapses. Hence NP-complete problems cannot be random self-reducible unless PH collapses.

**Problem 4. (Collapse of the AM hierarchy)**

1. For a class \(C\) of promise problems, we define \(pr\Sigma \cdot C\) to be the class of promise problems \(\Pi\) such that there exists a promise problem \(\Pi' \in C\) and a polynomial \(p\) for which

\[
x \in \Pi_Y \Rightarrow \exists y \in \{0, 1\}^{p(n)}(x, y) \in \Pi'_Y
\]

\[
x \in \Pi_N \Rightarrow \forall y \in \{0, 1\}^{p(n)}(x, y) \in \Pi'_N
\]

Similarly, we define \(prBP \cdot C\) to be the class of promise problems \(\Pi\) such that there exists a promise problem \(\Pi' \in C\) and a polynomial \(p\) for which

\[
x \in \Pi_Y \Rightarrow \Pr_{y \in \{0, 1\}^{p(n)}}[(x, y) \in \Pi'_Y] \geq 2/3
\]

\[
x \in \Pi_N \Rightarrow \Pr_{y \in \{0, 1\}^{p(n)}}[(x, y) \in \Pi'_N] \geq 2/3
\]

Show that for every integer \(k \geq 1\), \(prMA[k] = pr\Sigma \cdot prAM[k-1]\) and \(prAM[k] = prBP \cdot prMA[k-1]\), where \(prMA[0] = prAM[0] = prP\) (by definition).

2. Prove that \(prMA \subseteq prAM\). (Hint: First do error-reduction.)
3. Prove that for all \(k \geq 2\), \(prAM[k] = prAM\). Conclude that \(AM[k] = AM\).
4. Where in the above parts was it important that we were working with promise problems?