Answer all four questions with informal, but complete, proofs. You may not discuss this exam with anyone except myself and the TA, whether taking the course or not. Each question has equal weight, but some are more difficult than others. You may cite without proof any result from the Arora-Barak text or proved in class. In particular, you can use without proof the \( NP \)-completeness of any problem proved \( NP \)-complete in class, in the two texts, or on the homeworks, including: SAT, 3-SAT, Independent Set, Clique, Vertex Covering, 3-coloring, Hamiltonian Circuit, and Subset Sum. However, you may not use the list of \( NP \)-complete problems in the appendix of Garey and Johnson without proof.

**Computability of polynomial time.** Recall from the second homework that we defined a formal programming language to have two components, a validation procedure \( V(p) \) that determines whether \( p \) is a valid program, and an interpreter \( I(p, x) \) that determines what program \( p \) does on input \( x \). The expressible languages are \( L_p = \{ x | I(p, x) \text{ accepts} \} \) for \( p \) so that \( V(p) \).

Show that there is a formal programming language where both \( V \) and \( I \) are computable, so that the expressible languages are exactly those in \( P \) (Hint: add a counter, that keeps track of the number of steps taken).

Then show that, in contrast, the language \( PTM = \{ M : M \text{ is a TM that always halts on every input, and whose time function is polynomially bounded} \} \) is not computable.

**NP-Completeness.** Prove that the following problem is \( NP \)-complete:

Let \( G = (V, E) \) be an undirected graph. \( S \subseteq V \) is 3-seperated if any two distinct elements of \( S \) are at least distance 3 from each other in \( G \), i.e., not only are they not adjacent, but they share no common neighbors.

**Problem:** 3-Seperated subset (3SS)

**Instance:** An undirected graph \( G = (V, E) \), and an integer \( 1 \leq k \leq |V| \).

**Solution format:** A subset \( S \subseteq V, |S| = k \).

**Constraints:** \( S \) must be 3-separted.

**Objective** Decide whether a 3-seperated subset of size \( k \) exists.

**NP-hardness.** We showed that the problem of determining whether a polynomial is identically zero can be solved in polynomial time. Show that it is \( NP \)-hard to determine whether a multivariate polynomial \( p(x_1, ..., x_n) \),
even given explicitly as the sum of all of its non-zero terms, is ever zero, i.e., whether there are real numbers \( a_1, \ldots, a_n \) so that \( p(a_1, \ldots, a_n) = 0 \), is \( NP \)-hard. Give a short explanation of why this problem might not be in \( NP \). (Hint: Think of how to combine multiple polynomial equations into a single equation. Also, the variables here take on real values. Can you use equations to force them to take Boolean (0/1) values?)

Consequences of \( P = NP \). Let \( F(x, y) \) be a polynomially time computable function, restricted to inputs \( x, y \) so that \( |y| = |x| = n \).

We say \( V \) is an \( \epsilon \)–median value for \( F \) on \( x \) if

\[
\{|y|F(x, y) \leq V\}| \geq (1/2 - \epsilon)2^n
\]

and

\[
\{|y|F(x, y) \geq V\}| \geq (1/2 - \epsilon)2^n.
\]

(The actual median would be a 0-median, but there can be up to \( \epsilon(2^n) \) distinct \( \epsilon \)-medians.) Show that, if \( P = NP \), there is a polynomial time algorithm that, given \( x \) and any \( \epsilon = 1/poly(n) \), returns an \( \epsilon \)-median for \( F \) on \( x \).