You can look up material on the web and books, but you cannot look up solutions to the given problems. You can work in groups, but must write up the answers individually. As usual, any claims must be justified, and any algorithms you give must come with arguments for correctness.

Problem 1: Chernoff bounds (5pt)
Lets say at some polling location the probability that any voter votes for the democratic presidential candidate is .6. Assuming all votes are independent, how many voters n do I need to be sure with 99 percent probability that no less than $.55n$ votes are cast for the democrat? Please use Chernoff bounds.

Problem 2: Well Separated Pair Decompositions (10pt)
The following questions are related to the Well Separated Pair Decompositions (WSPD) described in class and appearing in the paper "A Decomposition of Multidimensional Point Sets with Applications to k-Nearest-Neighbors and n-Body Potential Fields" by Callahan and Kosaraju.

1. If a set of points P in Euclidean space has expansion constant c, can you use it to bound the depth of the decomposition tree. The expansion constant is defined in the paper on cover sets.
2. For a set of points P let $B_P(p, r) = \{q \in P | d(p, q) \leq r\}$. The approximate ball counting problem ACP(P, ϵ, r) is to return for every $p \in P$ a count n_p such that $|B_P(p, r(1-\epsilon))| < n_p < |B_P(p, r(1+\epsilon))|$. Outline an algorithm to do this using a WSPD. Please no more than 1/2 page.

Problem 3: Variants of SAT (10pt)
Let Φ be a boolean formula over n variables $X = \{x_1, x_2, \ldots, x_n\}$.

1. Consider the following problem. Given Φ, is there an assignment $f : X \rightarrow \{0, 1\}$ such that Φ evaluates to true under f, and also, each clause in Φ has at least one literal set to false. Prove that this variant of SAT is **NP**-hard.
2. Consider the variant of the above problem where Φ is only allowed to contain unnegated literals. Either prove this variant is also **NP**-hard or provide a polynomial time algorithm for it.

Problem 4: Termination of DPLL with clause learning (10pt)
Have a look at the slide on SAT solvers entitled "High level: DPLL w/Clause Learning." Consider the (underspecified) variant of DPLL with clause learning that always backtracks to the root of the tree (i.e., the empty partial assignment) after learning a conflict clause. Suppose it deduces conflicts using iterated unit-propogation, and learns a conflict clause from the implication graph from an arbitrary graph cut, and uses an arbitrary branching heuristic. Assume the algorithm has unlimited time and memory. Is it guaranteed to correctly determine if a boolean formula is satisfiable or unsatisfiable in a finite amount of time?