Write your answers in the separate answer booklet.

You have 180 minutes (after you get the answer booklet) to answer seven questions.

1. Recall that a ***run*** in a string $w \in \{0,1\}^*$ is a maximal substring of w whose characters are all equal. For example, the string `00011111110000` is the concatenation of three runs:

$$\texttt{00011111110000} = \texttt{000} \bullet \texttt{1111111} \bullet \texttt{0000}$$

   (a) Let L_a denote the set of all non-empty strings in $\{0,1\}^*$ where the length of the first run is equal to the number of runs. For example, L_a contains the strings `0` and `1100000` and `0001110`, but does not contain `000111` or `100011` or the empty string ε (because it has no first run).

   Prove that L_a is not a regular language.

   (b) Let L_b denote the set of all strings in $\{0,1\}^*$ that contain an even number of odd-length runs. For example, L_b contains the strings `010111` and `1111` and the empty string ε, but does not contain either `0011100` or `11110`.

   - Describe a DFA or NFA that accepts L_b ***and***
   - Give a regular expression that describes L_b.

   (You do not need to prove that your answers are correct.)

2. Aladdin and Badroulbadour are playing a cooperative game. Each player has an array of positive integers, arranged in a row of squares from left to right. Each player has a token, which starts at the leftmost square of their row; their goal is to move *both* tokens onto the rightmost squares at the same time.

   On each turn, *both* players move their tokens *in the same direction*, either left or right. The distance each token travels is equal to the number under that token at the beginning of the turn. For example, if a token starts on a square labeled 5, then it moves either five squares to the right or five squares to the left. If *either* token moves past either end of its row, then both players immediately lose.

   For example, if Aladdin and Badroulbadour are given the arrays

A:	7	5	4	1	2	3	3	2	3	1	4	2
B:	5	1	2	4	7	3	5	2	4	6	3	1

   they can win the game by moving right, left, left, right, right, left, right. On the other hand, if they are given the arrays

A:	2	3	5	1	3
B:	3	4	1	2	1

   they cannot win the game. (The first move must be to the right; then Aladdin's token moves out of bounds on the second turn.)

   Describe and analyze an algorithm to determine whether Aladdin and Badroulbadour can solve their puzzle, given the input arrays $A[1..n]$ and $B[1..n]$.

Problems 3–7 appear on the following pages.

3. Submit a solution to ***exactly one*** of the following problems. Don't forget to tell us which problem you've chosen!

   (a) Let $G = (V, E)$ be an arbitrary undirected graph. A subset $S \subseteq V$ of vertices is *mostly independent* if more than half the vertices of S have no neighbors in S.

   Prove that the following problem is NP-*complete*: Given an undirected graph G and an integer k, decide if G contains a mostly independent set of size at least k.

   (b) ***Prove*** that the following problem is NP-*complete*: Given an undirected graph G and an integer k, decide if G contains two disjoint independent sets of size k.

   (In fact, both of these problems are NP-complete, but we only want a proof for one of them.)

4. Recall that a *palindrome* is any string that is equal to its reversal, like REDIVIDER or POOP.

   (a) Describe and analyze an algorithm to find the length of the longest subsequence of a given string that is a palindrome.

   (b) A *double palindrome* is the concatenation of two *non-empty* palindromes, like REFEREE = REFER • EE or POOPREDIVIDER = POOP • REDIVIDER. Describe and analyze an algorithm to find the length of the longest subsequence of a given string that is a *double* palindrome. *[Hint: Use your algorithm from part (a).]*

   For both algorithms, the input is an array $A[1..n]$, and the output is an integer. For example, given the input string MAYBEDYNAMICPROGRAMMING, your algorithm for part (a) should return 7 (for the subsequences NMRORMN and MAYBYAM, among others), and your algorithm for part (b) should return 12 (for the subsequence MAYBYAMIRORI).

5. Suppose we are given n intervals $I = \{[a_1, b_1], \ldots, [a_n, b_n]\}$ (*not* necessarily sorted) which are pairwise disjoint (i.e., non-overlapping) along with a positive number L. Our goal is to find a maximum size set $X = \{x_1, \ldots, x_k\}$ of numbers such that each member of X belongs to an interval of I, and each pair of numbers in X at *at least* L apart, i.e.,

$$\min\left\{|x_j - x_i| \;\middle|\; 1 \le i < j \le k\right\} \ge L.$$

   For example, given the intervals $I = \{[1,4],[5,8],[11,12],[13,17],[19,23]\}$ and $L = 7$, one valid answer is $X = \{1, 8, 15, 23\}$ which has size $k = 4$. Note that in general, X is allowed to contain multiple numbers from the same interval.

   (a) ***Prove*** the following greedy strategy leads to a correct algorithm for the above problem:

   > Let ℓ be the leftmost number contained in any input interval. Add ℓ to X and compute the remainder of X by recursively working with the intervals I' made by removing all values within distance L of ℓ in each interval of I.

   [Hint: Let X^* be an optimal solution to the problem. Do an exchange argument to show there is an equally large solution X that contains ℓ.]

(b) Describe an algorithm to find a maximum size set of numbers X as described above. An algorithm based on the strategy in part (a) requires no further justification to receive full credit.

6. For each statement below, there are two boxes in the answer booklet labeled "Yes" and "No". Check "Yes" if the statement is ***always*** true and "No" otherwise, and give a ***brief*** (at most one short sentence) explanation of your answer. **Assume $P \neq NP$.** If there is any other ambiguity or uncertainty about an answer, check "No". For example:

- $x + y = 5$

  [X] Yes [ ] No — Suppose $x = 3$ and $y = 4$.

- 3SAT can be solved in polynomial time.

  [ ] Yes [X] No — 3SAT is NP-hard.

- If P = NP, then Emily can beat Super Mario Brothers in four minutes and 53 seconds.

  [X] Yes [ ] No — The hypothesis is false, so the implication is true.

Read each statement *very* carefully; some of these are deliberately subtle!

(a) Which of the following statements are true?

- The solution to the recurrence $T(n) = 4T(n/4) + O(n)$ is $T(n) = O(n \log n)$.
- The solution to the recurrence $T(n) = 4T(n/4) + O(n^2)$ is $T(n) = O(n^2 \log n)$.
- Every directed acyclic graph contains at most one source and at most one sink.
- Depth-first search explores every path from the source vertex s to every other vertex in the input graph.
- Suppose $A[1..n]$ is an array of integers. Consider the following recursive function:

$$Huh(i,j) = \begin{cases} 0 & \text{if } i < 0 \text{ or } j > n \\ \max \begin{Bmatrix} Huh(i, j+1) \\ Huh(i-1, j) \\ A[i] \cdot A[j] + Huh(i-1, j+1) \end{Bmatrix} & \text{otherwise} \end{cases}$$

We can compute $Huh(n, 0)$ by memoizing this function into an array $Huh[0..n, 0..n]$ in $O(n^2)$ time, increasing i in the outer loop and increasing j in the inner loop.

(b) Suppose we want to prove that the following language is undecidable.

$$\textsc{Duck} := \left\{ \langle M \rangle \;\middle|\; M \text{ accepts } \texttt{GRAPES} \text{ but rejects } \texttt{LEMONADE} \right\}$$

Professor Canard, your wetlands-ornithology instructor, suggests a reduction from the standard halting language

$$\textsc{Halt} := \left\{ (\langle M \rangle, w) \;\middle|\; M \text{ halts on input } w \right\}.$$

Specifically, suppose there is a Turing machine LooksLikeADuck that decides Duck. Professor Canard claims that the following algorithm decides Halt.

```
DecideHalt(⟨M⟩, w):
  Write code for the following algorithm:
      WaddleAway(x):
        run M on input w
        ⟨⟨ignore the output of M⟩⟩
        if x = LEMONADE
            return False
        else
            return True
  return LooksLikeADuck(⟨WaddleAway⟩)
```

Which of the following statements ***must be*** true ***for all*** inputs $(\langle M \rangle, w)$?

- If M accepts w, then WaddleAway accepts GRAPES.
- If M diverges on w, then WaddleAway rejects GRAPES.
- If M accepts w, then LooksLikeADuck accepts $\langle$WaddleAway$\rangle$.
- If M diverges on w, then DecideHalt rejects $(\langle M \rangle, w)$.
- DecideHalt decides the language Halt. (That is, Professor Canard's reduction is correct.)

7. **More of the same:** For each statement below, there are two boxes in the answer booklet labeled "Yes" and "No". Check "Yes" if the statement is ***always*** true and "No" otherwise, and give a ***brief*** (at most one short sentence) explanation of your answer. **Assume $P \neq NP$.**

   (a) Which of the following statements are true for ***all*** languages $L \subseteq \{0,1\}^*$?

   - $L^* = (L^*)^*$
   - If L is decidable, then L^* is decidable.
   - L is either regular or NP-hard.
   - If L is undecidable, then L has an infinite fooling set.
   - The language $\{\langle M \rangle \mid M \text{ decides } L\}$ is undecidable.

   (b) Suppose there is a ***polynomial-time*** many-one reduction from some language $A \subseteq \{0,1\}$ to some other language $B \subseteq \{0,1\}$. Which of the following statements are true, assuming P $\neq$ NP?

   - $A \cap B \neq \emptyset$.
   - There is an algorithm to transform any Python program that solves B in polynomial time into a Python program that solves A in polynomial time.
   - If B is NP-hard, then A is NP-hard.
   - If B is decidable, then A is decidable.
   - If a Turing machine M accepts every string in B, the *same* Turing machine M also accepts every string in A.