You have 180 minutes to answer six numbered questions.

Write your answers in the separate answer booklet.

Please return this question sheet and your cheat sheet with your answers.

1. 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 ≠ NP*.** If there is any other ambiguity or uncertainty about an answer, check "No". For example:

   - $x + y = 5$

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

   - 3SAT can be solved in polynomial time.

     Yes [✓ No] — 3SAT is NP-hard.

   - If P = NP then Jeff is the Queen of England.

     [✓ 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 $T(n) = 8T(n/2) + O(n^2)$ is $T(n) = O(n^2)$.
   - The solution to the recurrence $T(n) = 2T(n/8) + O(n^2)$ is $T(n) = O(n^2)$.
   - Every directed acyclic graph contains at least one sink.
   - Given *any* undirected graph G, we can compute a spanning tree of G in $O(V + E)$ time using whatever-first search.
   - Suppose $A[1..n]$ is an array of integers. Consider the following recursive function:

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

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

Problem 1 continues onto the next page.

1. *[continued]*

   (b) Which of the following statements are true for ***at least one*** language $L \subseteq \{\mathtt{0},\mathtt{1}\}^*$?

   - $L^* = (L^*)^*$
   - L is decidable, but L^* is undecidable.
   - L is neither regular nor NP-hard.
   - L is in P, and L has an infinite fooling set.
   - The language $\{\langle M\rangle \mid M \text{ accepts } L\}$ is undecidable.

   (c) Consider the following pair of languages:

   - $\textsc{DirHamPath} := \left\{G \mid G \text{ is a directed graph with a Hamiltonian path}\right\}$
   - $\textsc{Acyclic} := \left\{G \mid G \text{ is a directed acyclic graph}\right\}$

   (For concreteness, assume that in both of these languages, graphs are represented by their adjacency matrices.) Which of the following statements are true, assuming $\text{P} \neq \text{NP}$?

   - $\textsc{Acyclic} \in \text{NP}$
   - $\textsc{Acyclic} \cap \textsc{DirHamPath} \in \text{P}$
   - $\textsc{DirHamPath}$ is decidable.
   - A polynomial-time reduction from $\textsc{DirHamPath}$ to $\textsc{Acyclic}$ would imply P=NP.
   - A polynomial-time reduction from $\textsc{Acyclic}$ to $\textsc{DirHamPath}$ would imply P=NP.

   (d) Suppose there is a ***polynomial-time*** reduction from some language A over the alphabet $\{\mathtt{0},\mathtt{1}\}$ to some other language B over the alphabet $\{\mathtt{0},\mathtt{1}\}$. Which of the following statements are ***always*** true, assuming $\text{P} \neq \text{NP}$?

   - A is a subset of B.
   - If $B \in P$, then $A \in P$.
   - If B is NP-hard, then A is NP-hard.
   - If B is regular, then A is regular.
   - If B is regular, then A is decidable.

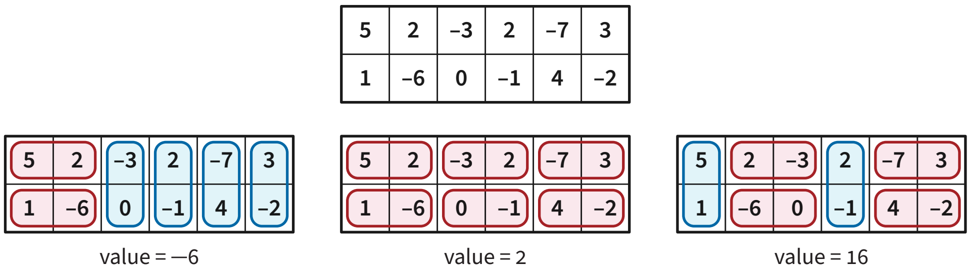
2. Describe and analyze an algorithm to determine whether the language accepted by a given DFA is finite or infinite. You can assume the input alphabet of the DFA is $\{\mathtt{0},\mathtt{1}\}$. *[Hint: DFAs are directed graphs.]*

3. Suppose you are asked to tile a $2 \times n$ grid of squares with dominos (1×2 rectangles). Each domino must cover exactly two grid squares, either horizontally or vertically, and each grid square must be covered by exactly one domino.

   Each grid square is worth some number of points, which could be positive, negative, or zero. The ***value*** of a domino tiling is the sum of the points in squares covered by vertical dominos, ***minus*** the sum of the points in squares covered by horizontal dominos.

   Describe an algorithm to compute the largest possible value of a domino tiling of a given $2 \times n$ grid. Your input is an array $Points[1..2, 1..n]$ of point values.

   As an example, here are three domino tilings of the same 2×6 grid, along with their values. The third tiling is optimal; no other tiling of this grid has larger value. Thus, given this 2×6 grid as input, your algorithm should return the integer 16.



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

   (a) Let Φ be a boolean formula in conjunctive normal form, with exactly three literals per clause (or in other words, an instance of 3SAT). ***Prove*** that it is NP-hard to decide whether Φ has a satisfying assignment in which *exactly half* of the variables are TRUE.

   (b) Let $G = (V, E)$ be an arbitrary directed graph whose edges have colors. A *rainbow Hamiltonian cycle* in G is a cycle that visits every vertex of G exactly once, in which no pair of consecutive edges have the same color. ***Prove*** that it is NP-hard to decide whether G has a rainbow Hamiltonian cycle.

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

5. Suppose you are given a height map of a mountain, in the form of an $n \times n$ grid of evenly spaced points, each labeled with an elevation value. You can safely hike directly from any point to any neighbor immediately north, south, east, or west, but only if the elevations of those two points differ by at most Δ. (The value of Δ depends on your hiking experience and your physical condition.)

   Describe and analyze an algorithm to determine the longest hike from some point s to some other point t, where the hike consists of an uphill climb (where elevations must increase at each step) followed by a downhill climb (where elevations must decrease at each step). Your input consists of an array $Elevation[1..n, 1..n]$ of elevation values, the starting point s, the target point t, and the parameter Δ.

6. 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:

$$00011111110000 = 000 \bullet 1111111 \bullet 0000$$

(a) Let L_a denote the set of all strings in $\{0,1\}^*$ where every 0 is followed immediately by at least one 1.

For example, L_a contains the strings 010111 and 1111 and the empty string ε, but does not contain either 001100 or 1111110.

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

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

(b) Let L_b denote the set of all strings in $\{0,1\}^*$ whose run lengths are increasing; that is, every run except the last is followed immediately by a *longer* run.

For example, L_b contains the strings 0110001111 and 1100000 and 000 and the empty string ε, but does not contain either 000111 or 100011.

Prove that L_b is not a regular language.