This is the practice final of *COSC 39: Theory of Computation*.
Please ***read the following instructions carefully*** before you proceed.

- You have 180 minutes (3 hours) for the exam.
- This is a closed-book written exam; no external equipment and resources are allowed.
- ***Do not*** turn the pages until instructed to.
- Write your Dartmouth ID clearly in the box below. ***Do not*** write your name anywhere.

Your ID:

- You must deposit your phone to the proctor before using the bathroom. Leaving the room without informing the proctor may result in failing the exam.
- After you finish the quiz, submit all papers to the proctor.
- The answers should be *clear*, *correct*, and *concise*.
- Each problem is worth 10 points; the total score is 50 points.
- Mark the problem numbers if you are writing solutions on the scratch paper(s).

---

1. ***1d-Clickomania.*** Consider the following game on a sequence of black-white tiles: In each step, you are allowed to choose any maximal run of *at least two* single-colored tiles, and remove them from the sequence. The rest of the tiles on the two sides will rejoin into a single sequence. If there is a way to remove all the tiles from the sequence, then we say that the sequence is *winnable*. For example,

   $$■■□□□■■□ \to ■■□□□□ \to ■■ \to \varepsilon$$

   is winnable, while ■■□□□□■□ is not.

   Prove that the following language is not regular:

   $$L = \left\{ w \in \{■, □\}^* : w \text{ is winnable} \right\}.$$

2. ***Fault-tolerant computing.*** Imagine you are given a good-old Turing machine with a single-tape and a single-head, but there is one catch: exactly one of the cells on the tape is *faulty*. Any symbol written on that cell might turn into other symbols unexpectedly, while the reading head is away. Worst of all, there are no ways to tell if the cell is faulty by examination, even after the cell has turned a symbol into another. (We assume that the faulty cell is not within the input.)

   Prove that the faulty Turing machine can still simulate a standard Turing machine.

3. ***Regular language equivalence.*** Let L_0 be an arbitrary regular language over alphabet Σ. Let D_0 be a DFA deciding L_0, given by some encoding $\langle D_0 \rangle$. Prove that the following language can be decidable by a Turing machine in polynomial time:

$$L_1 \coloneqq \left\{ \langle D \rangle : L(D) = L(D_0) \right\},$$

where $L(D)$ denote the language recognized by DFA D.

4. ***Inception.*** Let M_1 be a TM deciding the language L_1 from the previous problem (given as some encoding $\langle M_1 \rangle$). Prove that the following language is undecidable:

$$L_2 \coloneqq \left\{ \langle M \rangle : L(M) = L(M_1) \right\},$$

where $L(M)$ denote the language accepted by TM M.

5. Consider the following problem:

   ZEROSUMSET
   - ***Input:*** *A set X of n integers in $[-N \,..\, N] := \{-N, \ldots, 0, 1, \ldots, N\}$.*
   - ***Output:*** *Is there a nonempty subset S of X such that numbers in S sum up to 0?*

   Prove that ZEROSUMSET is NP-complete.

(extra scratch paper)

(extra scratch paper)

(extra scratch paper)

(extra scratch paper)