You have 90 minutes to complete this exam. You may assume without proof any statement proved in class.

(3 pts) 1  Give a simple verbal description of the language recognized by the following DFA.

2  Draw NFAs for the following languages over \{0, 1\}, taking full advantage of nondeterminism:

   a. strings such that in every prefix, the numbers of zeroes and ones differ by at most 2;
   b. strings that begin with 10 or 110, and end with 01 or 011.
3 Prove that the following languages over a given alphabet $\Sigma$ are regular:

(2 pts) a. strings in which no pair of consecutive characters are identical;

(2 pts) b. binary strings in which every even-numbered character is a 0;

(2 pts) c. the language $\{3, 6, 9, 12, 15, 18, 21, \ldots \}$ over the decimal alphabet, corresponding to natural numbers that are divisible by 3.
The symmetric difference of two languages $L'$ and $L''$, denoted $L' \Delta L''$, is the set of strings that belong to $L'$ or $L''$ but not both. Prove that regular languages are closed under symmetric difference.

Prove or argue to the contrary: adding a finite number of strings to a regular language necessarily results in a regular language.
The circular shift of a language $L$ is defined as $L^\circ = \{uv : vu \in L\}$, where $u$ and $v$ stand for arbitrary strings. For example, $\{1234\}^\circ = \{1234, 2341, 3412, 4123\}$. Prove that regular languages are closed under circular shift.

Describe an algorithm that takes as input a DFA and determines whether the automaton recognizes the empty language, $\emptyset$. 

(3 pts) 6 The circular shift of a language $L$ is defined as $L^\circ = \{uv : vu \in L\}$, where $u$ and $v$ stand for arbitrary strings. For example, $\{1234\}^\circ = \{1234, 2341, 3412, 4123\}$. Prove that regular languages are closed under circular shift.

(3 pts) 7 Describe an algorithm that takes as input a DFA and determines whether the automaton recognizes the empty language, $\emptyset$. 

(3 pts)
SOLUTIONS
You have 90 minutes to complete this exam. You may assume without proof any statement proved in class.

1. Give a simple verbal description of the language recognized by the following DFA.

```
0   1
0   0      0
0   1
1   1
```

**Solution.** Nonempty strings of even length.

2. Draw NFAs for the following languages over \{0, 1\}, taking full advantage of nondeterminism:

   a. strings such that in every prefix, the numbers of zeroes and ones differ by at most 2;
   b. strings that begin with 10 or 110, and end with 01 or 011.

```
0   0   0   0   0
1   1   1   1   1
```

**b.**
Prove that the following languages over a given alphabet $\Sigma$ are regular:

(2 pts) a. strings in which no pair of consecutive characters are identical;
(2 pts) b. binary strings in which every even-numbered character is a 0;
(2 pts) c. the language $\{3, 6, 9, 12, 15, 18, 21, \ldots \}$ over the decimal alphabet, corresponding to natural numbers that are divisible by 3.

Solution.

a. Let $L$ be the language in the problem statement. Then $\bar{L}$ is the set of all strings that contain a pair of consecutive characters that are identical, corresponding to the following NFA:

![Diagram](image)

This diagram features a branch for each symbol $\sigma \in \Sigma$. Since $\bar{L}$ is regular and regular languages are closed under complement, $L$ must be regular as well.

b. This language is regular because it is recognized by the following NFA:

![Diagram](image)

c. Recall that an integer is divisible by 3 if and only if the sum of its digits is divisible by 3. This suggests the automaton $(\{0, 1, 2\}, \{0, 1, 2, \ldots, 9\}, \delta, 0, \{0\})$ where $\delta(q, \sigma) = (q + \sigma) \mod 3$. This automaton almost works, except that it accepts syntactically invalid strings such as $\varepsilon$ or 003. To fix this, modify the automaton to only accept strings that start with a nonzero digit: $(\{\varepsilon, 0, 1, 2, \text{FAIL}\}, \{0, 1, 2, \ldots, 9\}, \delta, \epsilon, \{0\})$ where

$$
\delta(q, \sigma) = \begin{cases} 
(q + \sigma) \mod 3 & \text{if } q = 0, 1, 2, \\
\sigma \mod 3 & \text{if } q = \varepsilon \text{ and } \sigma \neq 0, \\
\text{FAIL} & \text{otherwise}.
\end{cases}
$$
4 The symmetric difference of two languages $L'$ and $L''$, denoted $L' \Delta L''$, is the set of strings that belong to $L'$ or $L''$ but not both. Prove that regular languages are closed under symmetric difference.

**Solution.** Let $L'$ and $L''$ be regular. By definition, $L' \Delta L'' = (L' \cap \overline{L''}) \cup (\overline{L'} \cap L'')$. Since regular languages are closed under complement, intersection, and union, it follows that $L' \Delta L''$ is regular.

5 Prove or argue to the contrary: adding a finite number of strings to a regular language necessarily results in a regular language.

**Solution.** As shown in class, every finite language is regular. Since regular languages are closed under union, it follows that the union of a regular language with a finite language is regular.
The circular shift of a language \( L \) is defined as \( L^\circ = \{uv : vu \in L\} \), where \( u \) and \( v \) stand for arbitrary strings. For example, \( \{1234\}^\circ = \{1234, 2341, 3412, 4123\} \). Prove that regular languages are closed under circular shift.

Let \( D = (Q, \Sigma, \delta, q_0, F) \) be a DFA for \( L \). Here is a nondeterministic procedure for deciding whether a given string \( w \) is in \( L^\circ \):

**Stage 1.** Choose a state \( q \in Q \) nondeterministically.

**Stage 2.** Launch the DFA on the input string, starting in state \( q \) rather than \( q_0 \). No need to wait for the DFA to process all of \( w \); whenever the DFA is in an accept state, you may choose to advance to the next stage.

**Stage 3.** Run the DFA on the unprocessed portion of \( w \), starting in state \( q_0 \) and accepting if you end up in state \( q \).

For any fixed \( q \in Q \), stages 2 and 3 can be implemented as an NFA \( D_q \), which consists of two copies of the original DFA: the first copy has all states marked as rejecting and has \( \epsilon \)-transitions added from any state in \( F \) to the state \( q_0 \) of the second copy, and the second copy has \( q \) marked as the only accept state. Now for each \( q \in Q \), create a copy of that composed automaton \( D_q \). To obtain an automaton for \( L^\circ \), it remains to introduce a new start state that has, for each \( q \in Q \), an \( \epsilon \)-transition to state \( q \) of the copy \( D_q \).

Describe an algorithm that takes as input a DFA and determines whether the automaton recognizes the empty language, \( \emptyset \).

We can view a DFA as a directed graph, with the DFA’s arrows and states corresponding to edges and vertices. The algorithm is simply to check if the graph contains a path from the start state to an accept state. This can be done either by trying out all candidate paths in a brute force manner, or by using an efficient graph algorithm you may have encountered in CS 180, such as depth- or breadth-first search.