For each statement below, check “Yes” if the statement is *always* true and “No” otherwise. Each correct answer is worth +1 point; each incorrect answer is worth −½ point; checking “I don’t know” is worth +¼ point; and flipping a coin is (on average) worth +¼ point.

<table>
<thead>
<tr>
<th>Yes</th>
<th>No</th>
<th>IDK</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Image" /></td>
<td><img src="image2" alt="Image" /></td>
<td><img src="image3" alt="Image" /></td>
</tr>
</tbody>
</table>

**Solution:** No infinite language is regular.

θ* is both infinite and regular.

<table>
<thead>
<tr>
<th>Yes</th>
<th>No</th>
<th>IDK</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image4" alt="Image" /></td>
<td><img src="image5" alt="Image" /></td>
<td><img src="image6" alt="Image" /></td>
</tr>
</tbody>
</table>

**Solution:** If L is regular, then for every string w ∈ L, there is a DFA that rejects w.

For every string w—in particular, for every string w ∈ L—there is a one-state DFA that rejects w.

<table>
<thead>
<tr>
<th>Yes</th>
<th>No</th>
<th>IDK</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image7" alt="Image" /></td>
<td><img src="image8" alt="Image" /></td>
<td><img src="image9" alt="Image" /></td>
</tr>
</tbody>
</table>

**Solution:** If L is context-free and L has a finite fooling set, then L is not regular.

∅ is a finite fooling set for every language, regular or not.

θ* is both context-free and regular.

<table>
<thead>
<tr>
<th>Yes</th>
<th>No</th>
<th>IDK</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image10" alt="Image" /></td>
<td><img src="image11" alt="Image" /></td>
<td><img src="image12" alt="Image" /></td>
</tr>
</tbody>
</table>

**Solution:** If L is regular and L′ ∩ L = ∅, then L′ is not regular.

Consider L′ = ∅, which is regular.

<table>
<thead>
<tr>
<th>Yes</th>
<th>No</th>
<th>IDK</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image13" alt="Image" /></td>
<td><img src="image14" alt="Image" /></td>
<td><img src="image15" alt="Image" /></td>
</tr>
</tbody>
</table>

**Solution:** The language {θ^i1θ^k | i + j + k ≥ 374} is regular.

This language is finite and therefore regular.

<table>
<thead>
<tr>
<th>Yes</th>
<th>No</th>
<th>IDK</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image16" alt="Image" /></td>
<td><img src="image17" alt="Image" /></td>
<td><img src="image18" alt="Image" /></td>
</tr>
</tbody>
</table>

**Solution:** The language {θ^i1θ^k | i + j − k ≥ 374} is regular.

Consider the fooling set F = θ1*.

<table>
<thead>
<tr>
<th>Yes</th>
<th>No</th>
<th>IDK</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image19" alt="Image" /></td>
<td><img src="image20" alt="Image" /></td>
<td><img src="image21" alt="Image" /></td>
</tr>
</tbody>
</table>

**Solution:** Let M = (Q, {θ, 1}, s, A, δ) be an arbitrary DFA, and let M′ = (Q, {θ, 1}, s, A, δ′) be the DFA obtained from M by changing every θ-transition into a 1-transition and vice versa. More formally, M and M′ have the same states, input alphabet, starting state, and accepting states, but δ′(q, θ) = δ(q, 1) and δ′(q, 1) = δ(q, θ). Then L(M) ∪ L(M′) = {θ, 1}∗.

If L(M) = ∅, then L(M′) = ∅ as well.
Let \( M = (Q, \Sigma, s, A, \delta) \) be an arbitrary NFA, and \( M' = (Q', \Sigma, s', A', \delta') \) be any NFA obtained from \( M \) by deleting some subset of the states. More formally, we have \( Q' \subseteq Q \), \( A' = A \cap Q' \), and \( \delta'(q, a) = \delta(q, a) \cap Q' \) for all \( q \in Q' \). Then \( L(M') \subseteq L(M) \).

**Solution:** Every path from \( s \) to an accepting state in \( M' \) is also a path from \( s \) to an accepting state in \( M \). Thus, every string accepted by \( M' \) is also accepted by \( M \). ■

For every non-regular language \( L \), the language \( \{ 0^{|w|} \mid w \in L \} \) is also non-regular.

**Solution:** If \( L = \{ 0^{2n} \mid n \geq 0 \} \), then \( L^* = 0^* \). ■

For every context-free language \( L \), the language \( \{ 0^{|w|} \mid w \in L \} \) is also context-free.

**Solution:** Replace each terminal symbol in the context-free grammar for \( L \) with a 0. ■
Prove that if $L$ is a regular language, then $\text{STRIPFINAL}\emptyset s(L)$ is also a regular language.

**Solution:** Let $M = (Q, s, A, \delta)$ be a DFA that accepts $L$. We construct an NFA $M' = (Q', s', A', \delta')$ with $\epsilon$-transitions that accepts $\text{STRIPFINAL}\emptyset s(L)$ as follows:

$Q' = Q \cup \{\text{pass}, \text{strip}\}$

$s' = (s, \text{pass})$

$A' = A \times \{\text{pass}\}$

$\delta'((q, \text{pass}), \epsilon) = \{(q, \text{strip})\}$

$\delta'((q, \text{pass}), a) = \{\delta(q, a), \text{pass}\}$

$\delta'((q, \text{strip}), \epsilon) = \{\delta(q, \emptyset), \text{strip}\}$

$\delta'((q, \text{strip}), a) = \emptyset$

Less formally: Add a shadow copy of $M$ with $\emptyset$-transitions replaced by $\epsilon$-transitions and $1$-transitions deleted, which guesses the deleted suffix of $\emptyset$s.

*Rubric:* 10 points =
+ 4 for a formal, complete, and unambiguous description of the output automaton.
− 2 for omitting $\emptyset$ transitions without declaring that omission.
+ 6 for correctness =
  + 1½ for accepting $\epsilon$ when $L$ contains at least one string in $\emptyset^*$
  + 1½ for accepting every non-empty string in $\text{STRIPFINAL}\emptyset s(L)$
  + 1½ for rejecting $\epsilon$ when $L \cap \emptyset^* = \emptyset$
  + 1½ for rejecting every non-empty string not in $\text{STRIPFINAL}\emptyset s(L)$
  − 1 for a single typo or similar mistake

• English explanation is not required, but may help us give you partial credit.
CS/ECE 374 A ♦ Spring 2018
Midterm 1st Problem 3

For each of the following languages $L$ over the alphabet $\Sigma = \{0, 1\}$, give a regular expression that represents $L$ and describe a DFA that recognizes $L$.

(a) \( \{ 0^n1^n \mid n \geq 1 \text{ and } w \in \Sigma^+ \} \)

**Solution:**

\[
0(0 + 1)(0 + 1)^+1 \]

**Solution:**

\[
0(0 + 1)^+1 \]

(b) All strings in $0^n 1^n 0^n$ whose length is even.

**Solution:**

\[
(00)^*(11)^*(00)^* + (00)^*(0 + 1)(11)^*(0 + 1)(00)^* \]

**Rubric:**

- 10 points = 2½ for each regular expression + 2½ for each DFA. Grade as four separate subproblems, each with IDK partial credit available.
- ½ for one typo
- 1 for incorrectly including/excluding a finite number of strings
- 2½ for incorrectly including/excluding an infinite number of strings
- 1 for omitting fail state from DFA without declaring that omission explicitly.

Explanations are not required for full credit. No penalty for leaving DFA states unlabeled. These are not the only correct answers.
### Midterm 1st Problem 4

(a) Give a self-contained, formal, recursive definition of the \textit{parity} function. (In particular, do not refer to \# or other functions defined in class.)

\textbf{Solution:}\  \( \text{parity}(w) = \begin{cases} \emptyset & \text{if } w = \varepsilon \\ a \oplus \text{parity}(x) & \text{if } w = ax \end{cases} \)

Here \( \oplus \) is exclusive-or: \( \emptyset \oplus \emptyset = 1 \oplus 1 = 0 \) and \( \emptyset \oplus 1 = 1 \oplus 0 = 1. \)

(b) Let \( L \) be an arbitrary regular language. Prove that \( \text{OddParity}(L) := \{ w \in L \mid \text{parity}(w) = 1 \} \) is also regular.

\textbf{Solution:}\  \( \text{OddParity}(L) = L \cap (\emptyset^*10^*10^*1)^* \emptyset^*10^* \), where the latter regular expression describes all binary strings with odd parity.

\textbf{Rubric:}\ 2 points = \( \frac{1}{2} \) for base case + \( 1 \frac{1}{2} \) for recursive case. These are not the only correct solutions. No penalty for using arithmetic or logical functions (such as \( \sim a \) or \( \neg a \) or \( \overline{a} \) or \( 1 - a \) or \( (a + 1) \mod 2 \) for logical negation). No credit for solutions that directly refer to “the number of 1s”.}

\textbf{Solution:}\  We can construct a DFA for \( \text{OddParity}(L) \) as the product of an arbitrary DFA for \( L \) and the following two-state DFA for all strings with odd parity:

\begin{center}
\begin{tikzpicture}
\node[state,initial] (q0) at (0,0) {0};
\node[state,accepting,draw=red] (q1) at (1,0) {1};
\path[->] (q0) edge[loop above] node {0} (q0)
(q0) edge[above] node {1} (q1)
(q1) edge[loop below] node {1} (q1);
\end{tikzpicture}
\end{center}

A state \((p, q)\) in the product DFA is accepting if and only if both \( p \) and \( q \) are accepting states of their respective DFAs.

\textbf{Solution:}\  Given a DFA \( M = (Q, s, A, \delta) \) for \( L \), we construct a new DFA \( M' = (Q', s', A', \delta') \) for \( \text{EvenParity}(L) \) as follows:

\( Q' = Q \times \{\emptyset, 1\} \)
\( s' = (s, \emptyset) \)
\( A' = A \times \{1\} \)
\( \delta'(q, p, a) = (\delta(q, a), p \oplus a) \)

In each state \((q, p)\) of \( M' \), \( q \) is the current state of \( M \) and \( p \) is the parity of the input read so far.
Rubric: 4 points:

- For closure argument: −1 for single mistake in “even” regular expression
- For product construction: −1 for not specifying the accepting states of the final DFA.
- For explicit DFA construction: 1 for complete explicit formal DFA description + 1½ for correct acceptances + 1½ for correct rejections

(c) Let $L$ be an arbitrary regular language. Prove that $AddParity(L) := \{\text{parity}(w) \cdot w \mid w \in L\}$ is also regular.

Solution: $AddParity(L) = ((\emptyset + 1) \cdot L \setminus OddParity((\emptyset + 1) \cdot L)$.

The string $\text{parity}(w) \cdot w$ always has even parity.

Solution: $AddParity(L) = ((\emptyset + 1) \cdot L) \cap (0^*10^*1)^*0^*10^*.$

Solution: $AddParity(L) = 1 \cdot OddParity(L) + \emptyset \cdot (L \setminus OddParity(L)) \cdot 1.$

Solution: $AddParity(L) = 1 \cdot OddParity(L) + \emptyset \cdot EvenParity(L)$, where $EvenParity(L) = L \setminus OddParity(L)$

Solution: $AddParity(L) = 1 \cdot OddParity(L) + \emptyset \cdot EvenParity(L)$, where $EvenParity(L) = L \cap (0^*10^*1)^*0^*$

Solution: $AddParity(L) = 1 \cdot OddParity(L) + \emptyset \cdot EvenParity(L)$, where $EvenParity(L)$ is accepted by the same product DFA as $OddParity(L)$, but with accepting states $A \times \{\emptyset\}$ instead of $A \times \{1\}$.

Rubric: Max 4 points. These are not the only correct solutions. A correct solution for (c) that assumes (b) is worth full credit, even if the solution to (b) is incorrect.
Let $L$ be the language \( \{0^i1^j0^k \mid 2i = k \text{ or } i = 2k\} \).

(a) **Prove** that $L$ is not a regular language.

**Solution:** Consider the set $F = 0^*$. Let $x$ and $y$ be arbitrary distinct strings in $F$. Then $x = 0^i$ and $y = 0^j$ for some integers $i \neq j$. Let $z = 0^i10^j$. Then $xz = 0^{2i}10^i \in L$, but $yz = 0^{i+i}10^i \notin L$, because $j + i \neq 2i$ and $2(j + i) \neq i$. Thus, $F$ is a fooling set for $L$. Because $F$ is infinite, $L$ cannot be regular.

**Solution:** Consider the set $F = 0^*$. Let $x$ and $y$ be arbitrary distinct strings in $F$. Then $x = 0^i$ and $y = 0^j$ for some integers $i \neq j$. Without loss of generality, assume $i < j$. Let $z = 10^{2j}$. Then $xz = 0^i10^{2j} \notin L$, because $2i < 2j$ and $i < j < 4j$. But $yz = 0^i10^{2j} \in L$. Thus, $F$ is a fooling set for $L$. Because $F$ is infinite, $L$ cannot be regular.

**Solution:** Consider the set $F = (00)^*$. Let $x$ and $y$ be arbitrary distinct strings in $F$. Then $x = 0^{2i}$ and $y = 0^{2j}$ for some positive integers $i \neq j$. Without loss of generality, assume $i < j$. Let $z = 10^i$. Then $xz = 0^{2i}10^i \in L$. But $yz = 0^{2j}10^i \notin L$ because $2j > 2i$ and $4j > i$. Thus, $F$ is a fooling set for $L$. Because $F$ is infinite, $L$ cannot be regular.

**Rubric:** 5 points: standard fooling set rubric (see HW2). These are not the only correct answers.
(b) Describe a context-free grammar for $L$.

**Solution:**

$$
S \rightarrow A \mid B \\
A \rightarrow 0A00 \mid C \quad [2i = k] \\
B \rightarrow 00B0 \mid C \quad [i = 2k] \\
C \rightarrow 1C \mid \epsilon \quad [1^*]
$$

**Rubric:** 5 points:
- 1 for single typo or similar mistake
- 1 for incorrectly omitting $\epsilon$
- 2 for incorrectly omitting/including a finite number of non-empty strings
- 5 for incorrectly omitting/including a infinite number of strings

This is not the only correct answer. Explanations of nonterminals are not required, but they may help us give you partial credit.