CS/ECE 374 A ✧ Fall 2023 Conflict Midterm 1 Problem 1	Name:

For each statement below, check "Yes" if the statement is always true and check "No" otherwise, and give a brief (one short sentence) explanation of your answer. Read these statements very carefully—small details matter!

(a) Every infinite language is regular.

☐ Yes ☒ No $\{0^n1^n \mid n \geq 0\}$ is infinite but not regular.

(b) The language $(0 + 1(01^*0)^*1)^*$ is not context-free.

☐ Yes ☒ No Every regular language is context-free.

(c) Every subset of an irregular language is irregular.

☐ Yes ☒ No $\emptyset$ is a subset of $\{0^n1^n \mid n \geq 0\}$

(d) The language $\left\{0^a1^b \;\middle|\; a - b \text{ is divisible by } 374\right\}$ is regular.

☒ Yes ☐ No DFA can remember $(\#1 - \#0)$ mod 374 in its states

(e) If language L is not regular, then L has a finite fooling set.

☒ Yes ☐ No $\emptyset$ is a fooling set for *every* language.

(f) If there is a DFA that rejects every string in language L, then L is regular.

☐ Yes ☒ No A DFA that rejects everything rejects every string in $\{0^n1^n \mid n \geq 0\}$.

(g) If language L is accepted by an DFA with n states, then its complement $\Sigma^* \setminus L$ is also accepted by a DFA with n states.

☒ Yes ☐ No $A' = Q \setminus A$

(h) 1^*0^* is a fooling set for the language $\left\{1^i0^{i+j}1^j \;\middle|\; i, j \geq 0\right\}$.

☐ Yes ☒ No Can't distinguish 100000 and 111000.

(i) Every regular language is accepted by a DFA with an odd number of accepting states.

☒ Yes ☐ No We can always add an extra inaccessible accepting state.

(j) The context-free grammar $S \to \varepsilon \mid 0S1S \mid 1S0S$ generates all strings in which the number of 0s equals the number of 1s.

☒ Yes ☐ No The shortest nonempty balanced prefix is generated by $0S1$ or $1S0$.

Rubric: 10 points = ½ for each correct answer + ½ for each correct explanation of a correct answer. These are not the only correct explanations. No negative scores for incorrect answers.

CS/ECE 374 A ✧ Fall 2023 Conflict Midterm 1 Problem 2	Name:

Consider the function cycleleft defined in the question handout. Let L be an arbitrary regular language.

(a) Prove that $\textsc{CycleLeft}(L) = \{\text{cycleleft}(w) \mid w \in L\}$ is a regular language.

(b) Prove that $\textsc{CycleRight}(L) = \{w \in \Sigma^* \mid \text{cycleleft}(w) \in L\}$ is a regular language.

Solution (a): At the start, M' guesses its final input bit f, passes it to M, and remembers it. Later, whenever M' reads an f, M' guesses whether that it is the last input bit, without feeding M, and rejects all further inputs. Otherwise, M' passes its input bits to M.

$$\begin{aligned}
Q' &= Q \times \{\text{start}, \text{final}0, \text{final}1, \text{end}\} \\
s' &= (s, \text{start}) \\
A' &= A \times \{\text{end}\}
\end{aligned}$$

$$\begin{aligned}
\delta'((q, \text{start}), \varepsilon) &= \big\{(\delta(q, 0), \text{final}0),\ (\delta(q, 1), \text{final}1),\ (q, \text{end})\big\} \\
\delta'((q, \text{final}0), 0) &= \big\{(\delta(q, 0), \text{final}0),\ (q, \text{end})\big\} \\
\delta'((q, \text{final}0), 1) &= \big\{(\delta(q, 1), \text{final}0))\big\} \\
\delta'((q, \text{final}1), 0) &= \big\{(\delta(q, 1), \text{final}1)\big\} \\
\delta'((q, \text{final}1), 1) &= \big\{(\delta(q, 1), \text{final}1),\ (q, \text{end})\big\}
\end{aligned}$$

All unspecified transitions go to $\varnothing$.

State (q, start) means M is in state q and M' has not read anything. State $(q, \text{final}0)$ means M is in state q and M''s last input bit will be 0. State $(q, \text{final}1)$ means M is in state q and M''s last input bit will be 1. State (q, end) means M' has read all of its input. ■

Rubric: 5 points: standard language transformation rubric. Explanation (in gray) is ***not*** required. This is not the only correct solution. −2½ for swapping parts (a) and (b).

Solution (b): At the start M' remembers the first input bit f but does not pass it to M. Otherwise, M' passes its input bits directly to M, and then optionally sends the remembered first input bit f to M as its final input bit.

$$\begin{aligned}
Q' &= Q \times \{\text{start}, \texttt{0}\text{first}, \texttt{1}\text{first}, \text{end}\} \\
s' &= (s, \text{start}) \\
A' &= A \times \{\text{start}, \text{end}\}
\end{aligned}$$

$$\begin{aligned}
\delta((q, \text{start}), \texttt{0}) &= \big\{(q, \texttt{0}\text{first}),\ (\delta(q, \texttt{0}), \text{end})\big\} \\
\delta((q, \text{start}), \texttt{1}) &= \big\{(q, \texttt{1}\text{first}),\ (\delta(q, \texttt{1}), \text{end})\big\}
\end{aligned}$$

$$\begin{aligned}
\delta((q, \texttt{0}\text{first}), a) &= \big\{(\delta(q, a), \texttt{0}\text{first}),\ (\delta(\delta(q, a), \texttt{0}), \text{end})\big\} \\
\delta((q, \texttt{1}\text{first}), a) &= \big\{(\delta(q, a), \texttt{1}\text{first}),\ (\delta(\delta(q, a), \texttt{1}), \text{end})\big\}
\end{aligned}$$

$$\delta((s, \text{end}), a) = \varnothing$$

State $(q, \texttt{0}\text{first})$ means M is in state q and M' has not read anything. State $(q, \texttt{0}\text{first})$ means M is in state q and M''s first input bit was $\texttt{0}$. State $(q, \texttt{1}\text{first})$ means M is in state q and M''s first input bit was $\texttt{1}$. State (q, end) means M is in state q and M' has read its entire input string. ■

Rubric: 5 points: standard language transformation rubric. Explanation (in gray) is ***not*** required. This is not the only correct solution. −2½ for swapping parts (a) and (b).

CS/ECE 374 A ✧ Fall 2023 Conflict Midterm 1 Problem 3	Name:

Consider the recursive function squish defined in the question handout.

(a) ***Prove*** that $\#(1, \text{squish}(w)) \le \#(1, w)$ for every string w.

(b) ***Prove*** that $\#(1, \text{squish}(w))$ is even if and only if $\#(1, w)$ is even (or equivalently, ***prove*** that $\#(1, \text{squish}(w)) \bmod 2 = \#(1, w) \bmod 2$) for every string w.

Solution (a): Let w be an arbitrary string.
Assume that $\#(1, \text{squish}(x)) \le \#(1, x)$ for every string x shorter than w.
There are three cases to consider.

- If $|w| \le 1$, then $\#(1, \text{squish}(w)) = \#(1, w)$ by definition of squish.
- Suppose $w = aax$ for some string x and some bit a. Then

$$\begin{aligned}
\#(1, \text{squish}(w)) &= \#(1, \text{squish}(aax)) && w = aax \\
&= \#(1, 0 \cdot \text{squish}(x)) && \text{def. squish} \\
&= \#(1, \text{squish}(x)) && \text{def. } \#(1, \cdot) \\
&\le \#(1, x) && \textit{induction hypothesis} \\
&\le \#(1, aax) && \text{def. } \#(1, \cdot) \\
&= \#(1, w) && w = aax
\end{aligned}$$

- Suppose $w = abx$ for some string x and bits $a \neq b$. Then

$$\begin{aligned}
\#(1, \text{squish}(w)) &= \#(1, \text{squish}(abx)) && w = abx \\
&= \#(1, 1 \cdot \text{squish}(x)) && \text{def. squish} \\
&= 1 + \#(1, \text{squish}(x)) && \text{def. } \#(1, \cdot) \\
&\le 1 + \#(1, x) && \textit{induction hypothesis} \\
&= \#(1, abx) && \#(1, ab) = 1 \\
&= \#(1, w) && w = abx
\end{aligned}$$

■

Rubric: 5 points: standard induction rubric. Explanations for each step (in gray) are ***not*** required. No "bad style" penalties.

Solution (b): Let w be an arbitrary string.
Assume that $\#(1, \text{squish}(x)) \bmod 2 = \#(1, x) \bmod 2$ for every string x shorter than w.
There are three cases to consider:

- If $|w| \leq 1$, then $\#(1, \text{squish}(w)) \bmod 2 = \#(1, w) \bmod 2$ by definition of squish.
- Suppose $w = aax$ for some string x and some bit a. Then

$$\begin{aligned}
\#(1, \text{squish}(w)) \bmod 2 &= \#(1, \text{squish}(aax)) \bmod 2 && w = aax \\
&= \#(1, 0 \cdot \text{squish}(x)) \bmod 2 && \text{def. squish} \\
&= \#(1, \text{squish}(x)) \bmod 2 && \text{def. } \#(1, \cdot) \\
&= \#(1, x) \bmod 2 && \textit{induction hypothesis} \\
&= \#(1, aax) \bmod 2 && \text{def. } \#(1, \cdot) \\
&= \#(1, w) && w = aax
\end{aligned}$$

- Suppose $w = abx$ for some string x and bits $a \neq b$. Then

$$\begin{aligned}
\#(1, \text{squish}(w)) \bmod 2 &= \#(1, \text{squish}(abx)) \bmod 2 && w = abx \\
&= \#(1, 1 \cdot \text{squish}(x)) \bmod 2 && \text{def. squish} \\
&= (1 + \#(1, \text{squish}(x))) \bmod 2 && \text{def. } \#(1, \cdot) \\
&= (1 + \#(1, x) \bmod 2) && \textit{induction hypothesis} \\
&= \#(1, abx) \bmod 2 && \#(1, ab) = 1 \\
&= \#(1, w) \bmod 2 && w = abx
\end{aligned}$$

■

Rubric: 5 points: standard induction rubric. Explanations for each step (in gray) are ***not*** required. No "bad style" penalties.

CS/ECE 374 A ✧ Fall 2023 Conflict Midterm 1 Problem 4	Name:

Let L be the set of all strings in $\{0,1\}^*$ in which every run of 0s is followed immediately by a *shorter* run of 1s. For example, the strings 0001100000111 and 11100100000111 and 11111 are in L, but the strings 00011111 and 000110000 are not.

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

(b) Describe a context-free grammar for L.

Solution (a): Let $F = 0^*$.
Let x and y be arbitrary distinct strings in F.
Then $x = 0^i$ and $y = 0^j$ for some non-negative integers $i \neq j$.
Assume $i < j$ (otherwise swap variable names).
Let $z = 1^i$.
Then $xz = 0^i 1^i \notin L$, because the run of 1s has the same length as the run of 0s.
But $yz = 0^j 1^i \in L$, because $i < j$.
We conclude that F is a fooling set for L, and F is infinite, so L cannot be regular. ■

Rubric: 5 points: standard fooling set rubric. This is not the only correct solution. Boilerplate in gray is ***not*** required.

Solution (b):

$$
\begin{aligned}
S &\to SH \mid R \\
H &\to 0L \mid 0E && \text{run of 0s followed by s}H\text{orter run of 1s} \\
E &\to 0E1 \mid 01 && \text{run of 0s followed by }E\text{qual-length run of 1s} \\
R &\to 1R \mid \varepsilon && \text{optional initial }R\text{un of 1s}
\end{aligned}
$$

■

Solution (b):

$$
\begin{aligned}
S &\to 1S \mid A && \text{optimal initial run of 1s then } A \\
A &\to AR \mid \varepsilon && = R^* \\
R &\to 0R \mid 0R1 \mid 001 && R\text{un of 0s followed by shorter run of 1s}
\end{aligned}
$$

■

Rubric: 5 points. −1 for a minor mistake. 2 points for significant progress. This is not the only correct solution. Nonterminal explanations (in gray) are ***not*** required.

CS/ECE 374 A ✧ Fall 2023 Conflict Midterm 1 Problem 5	Name:

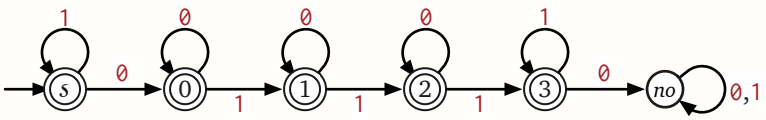
For each of the following languages L over the alphabet $\Sigma = \{0, 1\}$, describe a DFA that accepts L **and** give a regular expression that represents L. You do not need to justify your answers.

(a) Strings that do not contain the subsequence 01110.

(b) Strings that contain at least two even-length runs of 1s.

Solution (a): Equivalently, strings with at most two 1s between the first 0 and the last 0.

$$1^*0^*(\varepsilon + 1 + 10^*1)0^*1^*$$



The states have the following meanings:

- s: We have not read any 0s.
- 0: We have read at least one 0, but no later 1.
- 1: We have read one 1 after the first 0
- 2: We have read two 1s after the first 0
- 3: We have read at least three 1s after the first 0
- no: We have read the forbidden subsequence

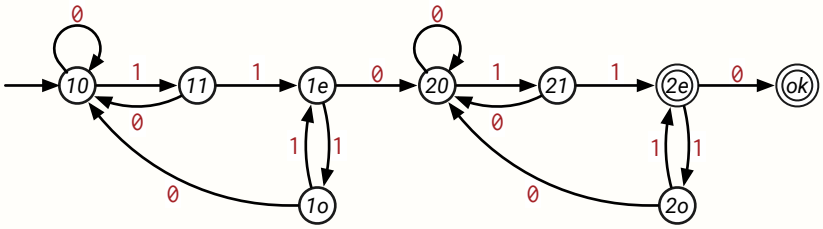
■

Rubric: 5 points = 2½ for regular expression (standard rubric) + 2½ for DFA (standard rubric). Explanations (in gray) and DFA state names are **not** required. These are not the only correct solutions.

Solution (b):

$$(\varepsilon + (0+1)^*0)\ \underline{11(11)^*}\ (0(0+1)^*0+0)\ \underline{11(11)^*}\ (0(0+1)^*+\varepsilon)$$

The bracketed subexpressions are even-length runs of 1s.



The states have the following meanings:

- 10: We have read no even run of 1s, and the last symbol was not 1.
- 11: We have read no even run of 1s, and we've just read one 1.
- 1*e*: We have read no even run of 1s, and we've just read an even number of 1s.
- 1*o*: We have read no even run of 1s, and we've just read an odd number of 1s.
- 20: We have read one even run of 1s, and the last symbol was not 1.
- 21: We have read one even run of 1s, and we've just read one 1.
- 2*e*: We have read one even run of 1s, and we've just read an even number of 1s.
- 2*o*: We have read one even run of 1s, and we've just read an odd number of 1s.
- *ko*: We have read two even runs of 1s

■

Rubric: 5 points = 2½ for regular expression (standard rubric) + 2½ for DFA (standard rubric). Explanations and DFA state names (in gray) are **not** required. These are not the only correct solutions.