CS/ECE 374 A ✧ Fall 2019

Midterm 1 Study Questions

This is a “core dump” of potential questions for Midterm 1. This should give you a good idea of the *types* of questions that we will ask on the exam—in particular, there *will* be a series of True/False questions—but the actual exam questions may or may not appear in this list. This list intentionally includes a few questions that are too long or difficult for exam conditions; *most* of these are indicated with a *star.

Questions from Jeff’s past exams are labeled with the semester they were used: ⟨⟨*S14*⟩⟩, ⟨⟨*F14*⟩⟩, ⟨⟨*F16*⟩⟩, or ⟨⟨*S18*⟩⟩. Questions from this semester’s homework are labeled ⟨⟨*HW*⟩⟩. Questions from this semester’s labs are labeled ⟨⟨*Lab*⟩⟩. Some unflagged questions may have been used in exams by other instructors.

How to Use These Problems

Solving every problem in this handout is ***not*** the best way to study for the exam. Memorizing the solutions to every problem in this handout is the ***absolute worst*** way to study for the exam.

What we recommend instead is to work on a *sample* of the problems. Choose one or two problems at random from each section and try to solve them from scratch under exam conditions—by yourself, in a quiet room, with a 30-minute timer, *without* your notes, *without* the internet, and if possible, even without your cheat sheet. If you’re comfortable solving a few problems in a particular section, you’re probably ready for that type of problem on the exam. Move on to the next section.

Discussing problems with other people (in your study groups, in the review sessions, in office hours, or on Piazza) and/or looking up old solutions can be *extremely* helpful, but ***only after*** you have (1) made a good-faith effort to solve the problem on your own, and (2) you have either a candidate solution or some idea about where you’re getting stuck.

If you find yourself getting stuck on a particular type of problem, try to figure out *why* you’re stuck. Do you understand the problem statement? Are you stuck on choosing the right high-level approach, are you stuck on the technical details, or are you struggling to express your ideas clearly?

Similarly, if feedback suggests that your solutions to a particular type of problem are incorrect or incomplete, try to figure out what you missed. For induction proofs: Are you sure you have the right induction hypothesis? Are your cases obviously exhaustive? For regular expressions, DFAs, NFAs, and context-free grammars: Is your solution both exclusive and exhaustive? Did you try a few positive examples *and* a few negative examples? For fooling sets: Are you imposing enough structure? Are x and y really *arbitrary* strings from F? For language transformations: Are you transforming in the right direction? Are you using non-determinism correctly? Do you understand the formal notation for DFAs and NFAs?

Remember that your goal is *not* merely to “understand” the solution to any particular problem, but to become more comfortable with solving a certain *type* of problem on your own. **"Understanding" is a trap; aim for mastery.** If you can identify specific steps that you find problematic, read more *about those steps*, focus your practice *on those steps*, and try to find helpful information *about those steps* to write on your cheat sheet. Then work on the next problem!

Induction on Strings

Give complete, formal inductive proofs for the following claims. Your proofs must reply on the formal recursive definitions of the relevant string functions, not on intuition. Recall that the concatenation • and length $|\cdot|$ functions are formally defined as follows:

$$w \bullet y := \begin{cases} y & \text{if } w = \varepsilon \\ a \cdot (x \bullet y) & \text{if } w = ax \text{ for some } a \in \Sigma \text{ and } x \in \Sigma^* \end{cases}$$

$$|w| := \begin{cases} 0 & \text{if } w = \varepsilon \\ 1 + |x| & \text{if } w = ax \text{ for some } a \in \Sigma \text{ and } x \in \Sigma^* \end{cases}$$

1.1 The ***reversal*** $\boldsymbol{w^R}$ of a string w is defined recursively as follows:

$$w^R := \begin{cases} \varepsilon & \text{if } w = \varepsilon \\ x^R \bullet a & \text{if } w = ax \text{ for some } a \in \Sigma \text{ and } x \in \Sigma^* \end{cases}$$

(a) Prove that $(w \bullet x)^R = x^R \bullet w^R$ for all strings w and x. ⟨⟨***F14***⟩⟩

(b) Prove that $(w^R)^R = w$ for every string w.

(c) Prove that $|w| = |w^R|$ for every string w.

1.2 For any string w and any non-negative integer n, let $\boldsymbol{w^n}$ denote the string obtained by concatenating n copies of w; more formally, define

$$w^n := \begin{cases} \varepsilon & \text{if } n = 0 \\ w \bullet w^{n-1} & \text{otherwise} \end{cases}$$

For example, $(\texttt{BLAH})^5 = \texttt{BLAHBLAHBLAHBLAHBLAH}$ and $\varepsilon^{374} = \varepsilon$.

(a) Prove that $w^m \bullet w^n = w^{m+n}$ for every string w and all non-negative integers n and m.

(b) Prove that $(w^m)^n = w^{mn}$ for every string w and all non-negative integers n and m.

(c) Prove that $|w^n| = n|w|$ for every string w and every integer $n \geq 0$.

(d) Prove that $(w^n)^R = (w^R)^n$ for every string w and every integer $n \geq 0$.

1.3 ⟨⟨***Lab***⟩⟩ Let $\#(a, w)$ denote the number of times symbol a appears in string w. For example, $\#(\texttt{X}, \texttt{WTF374}) = 0$ and $\#(\texttt{0}, \texttt{000010101010010100}) = 12$.

(a) Give a formal recursive definition of $\#(a, w)$.

(b) Prove that $\#(a, w \bullet z) = \#(a, w) + \#(a, z)$ for all symbols a and all strings w and z.

(c) Prove that $\#(a, w^R) = \#(a, w)$ for all symbols a and all strings w.

1.4 Consider the following pair of mutually recursive functions:

$$evens(w) := \begin{cases} \varepsilon & \text{if } w = \varepsilon \\ odds(x) & \text{if } w = ax \end{cases} \qquad odds(w) := \begin{cases} \varepsilon & \text{if } w = \varepsilon \\ a \cdot evens(x) & \text{if } w = ax \end{cases}$$

For example, $evens(\texttt{0001101}) = \texttt{010}$ and $odds(\texttt{0001101}) = \texttt{0011}$.

(a) Prove the following identity for all strings w and x:

$$evens(w \bullet x) = \begin{cases} evens(w) \bullet evens(x) & \text{if } |w| \text{ is even,} \\ evens(w) \bullet odds(x) & \text{if } |w| \text{ is odd.} \end{cases}$$

(b) State and prove a similar identity for $odds(w \bullet x)$.

(c) Prove the following identity for all strings w:

$$evens(w^R) = \begin{cases} (evens(w))^R & \text{if } |w| \text{ is odd,} \\ (odds(w))^R & \text{if } |w| \text{ is even.} \end{cases}$$

(d) Prove that $|w| = |evens(w)| + |odds(w)|$ for every string w.

1.5 The ***complement*** $\boldsymbol{w^c}$ of a string $w \in \{\texttt{0}, \texttt{1}\}^*$ is obtained from w by replacing every $\texttt{0}$ in w with a $\texttt{1}$ and vice versa. The complement function can be defined recursively as follows:

$$w^c := \begin{cases} \varepsilon & \text{if } w = \varepsilon \\ \texttt{1} \cdot x^c & \text{if } w = \texttt{0}x \\ \texttt{0} \cdot x^c & \text{if } w = \texttt{1}x \end{cases}$$

(a) Prove that $|w| = |w^c|$ for every string w.

(b) Prove that $(x \bullet y)^c = x^c \bullet y^c$ for all strings x and y.

(c) Prove that $\#(\texttt{1}, w) = \#(\texttt{0}, w^c)$ for every string w.

1.6 Consider the following recursively defined function:

$$stutter(w) := \begin{cases} \varepsilon & \text{if } w = \varepsilon \\ aa \bullet stutter(x) & \text{if } w = ax \end{cases}$$

For example, $stutter(\texttt{MISSISSIPPI}) = \texttt{MMIISSSSIISSSSIIPPPPII}$.

(a) Prove that $|stutter(w)| = 2|w|$ for every string w.

(b) Prove that $evens(stutter(w)) = w$ for every string w.

(c) Prove that $odds(stutter(w)) = w$ for every string w.

(d) Prove that w is a palindrome if and only if $stutter(w)$ is a palindrome, for every string w.

1.7 Consider the following recursive function:

$$faro(w,z) := \begin{cases} z & \text{if } w = \varepsilon \\ a \cdot faro(z,x) & \text{if } w = ax \end{cases}$$

For example, $faro(\texttt{0011}, \texttt{0101}) = \texttt{00011011}$. (A "faro shuffle" splits a deck of cards into two equal piles and then perfectly interleaves them.)

(a) Prove that $|faro(x,y)| = |x| + |y|$ for all strings x and y.

(b) Prove that $faro(w,w) = stutter(w)$ for every string w.

(c) Prove that $faro(odds(w), evens(w)) = w$ for every string w.

Regular expressions

For each of the following languages over the alphabet $\{0, 1\}$, give an equivalent regular expression, and *briefly* argue why your expression is correct. (On exams, we will not ask for justifications, but you should still justify your expressions in your head.)

2.1 Every string of length at most 3. *[Hint: Don't try to be clever.]*

2.2 All strings except 010.

2.3 All strings that end with the suffix 010.

2.4 All strings that do not end with the suffix 010.

2.5 All strings that contain the substring 010.

2.6 All strings that do not contain the substring 010.

2.7 All strings that contain the subsequence 010.

2.8 All strings that do not contain the subsequence 010.

2.9 All strings containing the substring 10 or the substring 01.

2.10 *⟨⟨F16⟩⟩* All strings containing either the substring 10 or the substring 01, but not both.

2.11 All strings containing the subsequence 10 or the subsequence 01.

2.12 All strings containing either the subsequence 10 or the subsequence 01, but not both.

2.13 All strings containing at least two 1s and at least one 0.

2.14 All strings containing *either* at least two 1s *or* at least one 0.

2.15 *⟨⟨lab⟩⟩* All strings such that *in every prefix*, the number of 0s and the number of 1s differ by at most 1.

2.16 *⟨⟨F14⟩⟩* All strings in which every run of consecutive 0s has even length and every run of consecutive 1s has odd length.

2.17 The set of all strings in $\{0, 1\}^*$ whose length is divisible by 3.

2.18 *⟨⟨S14⟩⟩* The set of all strings in 0^*1^* whose length is divisible by 3.

2.19 The set of all strings in $\{0, 1\}^*$ in which the number of 1s is divisible by 3.

2.20 *⟨⟨S18⟩⟩* All strings in $0^*1^*0^*$ whose length is even.

2.21 *⟨⟨S18⟩⟩* $\left\{0^n w 1^n \mid n \geq 1 \text{ and } w \in \Sigma^+\right\}$

2.22 All strings that end with the suffix 0000000000 (ten 0s)

2.23 All strings whose last ten symbols include an odd number of 1s.

Direct DFA construction.

Draw or formally describe a DFA that recognizes each of the following languages. Don't forget to describe the states of your DFA in English.

3.1 Every string of length at most 3.

3.2 All strings except 010.

3.3 All strings that end with the suffix 010.

3.4 All strings that do not end with the suffix 010.

3.5 All strings that contain the substring 010.

3.6 All strings that do not contain the substring 010.

3.7 All strings that contain the subsequence 010.

3.8 All strings that do not contain the subsequence 010.

3.9 The language {LONG, LUG, LEGO, LEG, LUG, LOG, LINGO}.

3.10 The language $\mathtt{MOO}^* + \mathtt{MEOO}^*\mathtt{W}$

3.11 All strings in which every run of consecutive 0s has even length and every run of consecutive 1s has odd length. ⟨⟨*F14*⟩⟩

3.12 All strings containing the substring 10 or the substring 01.

3.13 All strings containing either the substring 10 or the substring 01, but not both.

3.14 All strings containing the subsequence 10 or the subsequence 01.

3.15 All strings containing either the subsequence 10 or the subsequence 01, but not both.

3.16 The set of all strings in $\{0, 1\}^*$ whose length is divisible by 3.

3.17 ⟨⟨*S14*⟩⟩ The set of all strings in 0^*1^* whose length is divisible by 3.

3.18 The set of all strings in $\{0, 1\}^*$ in which the number of 1s is divisible by 3.

3.19 All strings w such that the binary value of w^R is divisible by 5.

3.20 ⟨⟨*lab*⟩⟩ All strings such that *in every prefix*, the number of 0s and the number of 1s differ by at most 2.

3.21 ⟨⟨*S18*⟩⟩ All strings in $0^*1^*0^*$ whose length is even.

3.22 ⟨⟨*S18*⟩⟩ $\left\{0^n w 1^n \;\middle|\; n \geq 1 \text{ and } w \in \Sigma^+\right\}$

3.23 All strings that end with the suffix 0000000000 (ten 0s)

3.24 All strings whose last ten symbols include an odd number of 1s.

Fooling sets

Prove that each of the following languages is *not* regular.

4.1 The set of all strings in $\{0,1\}^*$ with more 0s than 1s. ⟨⟨*S14*⟩⟩

4.2 The set of all strings in $\{0,1\}^*$ with fewer 0s than 1s.

4.3 The set of all strings in $\{0,1\}^*$ with exactly twice as many 0s as 1s.

4.4 The set of all strings in $\{0,1\}^*$ with at least twice as many 0s as 1s.

4.5 $\left\{0^{2^n} \mid n \geq 0\right\}$ ⟨⟨*Lab*⟩⟩

4.6 $\left\{0^{F_n} \mid n \geq 0\right\}$, where F_n is the nth Fibonacci number, defined recursively as follows:

$$F_n := \begin{cases} 0 & \text{if } n = 0 \\ 1 & \text{if } n = 1 \\ F_{n-1} + F_{n-2} & \text{otherwise} \end{cases}$$

[Hint: If $F_i + F_j$ is a Fibonacci number, then either $i = j \pm 1$ or $\min\{i, j\} \leq 2$.]

4.7 $\left\{0^{n^3} \mid n \geq 0\right\}$

4.8 $\left\{x\#y \mid x, y \in \{0,1\}^* \text{ and } \#(0, x) = \#(1, y)\right\}$

4.9 $\left\{xx^c \mid x \in \{0,1\}^*\right\}$, where x^c is the *complement* of x, obtained by replacing every 0 in x with a 1 and vice versa. For example, $0001101^c = 1110010$.

4.10 The language of properly balanced strings of parentheses, described by the context-free grammar $S \to \varepsilon \mid SS \mid (S)$. ⟨⟨*Lab*⟩⟩

4.11 $\left\{(01)^n(10)^n \mid n \geq 0\right\}$

4.12 $\left\{(01)^m(10)^n \mid n \geq m \geq 0\right\}$

4.13 $\left\{w\#x\#y \mid w, x, y \in \{0,1\}^* \text{ and } w, x, y \text{ are not all equal}\right\}$

4.14 $\left\{0^i 1^j 0^k \mid i = j \text{ or } j = k\right\}$ ⟨⟨*S18*⟩⟩

4.15 $\left\{0^i 1^j 0^k \mid 2i = k \text{ or } i = 2k\right\}$ ⟨⟨*S18*⟩⟩

Regular or Not?

For each of the following languages, either prove that the language is regular (by describing a DFA, NFA, or regular expression), or prove that the language is not regular (using a fooling set argument). Unless otherwise specified, all languages are over the alphabet $\{0,1\}$.

5.1 ⟨⟨*F14*⟩⟩ The set of all strings in $\{0,1\}^*$ in which the substrings 01 and 10 appear the same number of times. (For example, the substrings 01 and 01 each appear three times in the string 1100001101101.)

5.2 ⟨⟨*F14*⟩⟩ The set of all strings in $\{0,1\}^*$ in which the substrings 00 and 11 appear the same number of times. (For example, the substrings 00 and 11 each appear three times in the string 1100001101101.)

5.3 ⟨⟨*F14*⟩⟩ $\{www \mid w \in \Sigma^*\}$

5.4 ⟨⟨*F14*⟩⟩ $\{wxw \mid w, x \in \Sigma^*\}$

5.5 The set of all strings in $\{0,1\}^*$ such that in every prefix, the number of 0s is greater than the number of 1s.

5.6 The set of all strings in $\{0,1\}^*$ such that in every *non-empty* prefix, the number of 0s is greater than the number of 1s.

5.7 $\{0^m1^n \mid 0 \le m - n \le 374\}$

5.8 $\{0^m1^n \mid 0 \le m + n \le 374\}$

5.9 The language generated by the following context-free grammar:

$$S \to 0A1 \mid \varepsilon$$
$$A \to 1S0 \mid \varepsilon$$

5.10 The language generated by the following context-free grammar:

$$S \to 0S1 \mid 1S0 \mid \varepsilon$$

5.11 $\{w\#x \mid w, x \in \{0,1\}^* \text{ and no substring of } w \text{ is also a substring of } x\}$

5.12 $\{w\#x \mid w, x \in \{0,1\}^* \text{ and no } \textit{non-empty} \text{ substring of } w \text{ is also a substring of } x\}$

5.13 $\{w\#x \mid w, x \in \{0,1\}^* \text{ and } \textit{every} \text{ non-empty substring of } w \text{ is also a substring of } x\}$

5.14 $\{w\#x \mid w, x \in \{0,1\}^* \text{ and } w \text{ is a substring of } x\}$

5.15 $\{w\#x \mid w, x \in \{0,1\}^* \text{ and } w \text{ is a proper substring of } x\}$

5.16 $\{xy \mid \#(0,x) = \#(1,y) \text{ and } \#(1,x) = \#(0,y)\}$

5.17 $\{xy \mid \#(0,x) = \#(1,y) \text{ or } \#(1,x) = \#(0,y)\}$

5.18 $\{0^a1^b0^c \mid (a \le b + c \text{ and } b \le a + c) \text{ or } c \le a + b\}$ ⟨⟨*HW*⟩⟩

5.19 $\left\{0^a 1^b 0^c \;\middle|\; a \le b + c \text{ and } (b \le a + c \text{ or } c \le a + b)\right\}$ **⟨⟨*HW*⟩⟩**

5.20 $\left\{wxw^R \;\middle|\; w, x \in \Sigma^+\right\}$ **⟨⟨*HW*⟩⟩**

5.21 $\left\{ww^R x \;\middle|\; w, x \in \Sigma^+\right\}$ **⟨⟨*HW*⟩⟩**

Product/Subset Constructions

For each of the following languages $L \subseteq \{0,1\}^*$, formally describe a DFA $M = (Q, \{0,1\}, s, A, \delta)$ that recognizes L. ***Do not attempt to draw the DFA.*** Instead, give a complete, precise, and self-contained description of the state set Q, the start state s, the accepting state A, and the transition function δ. Do ***not*** just describe several smaller DFAs and write "product construction!"

6.1 ⟨⟨*S14*⟩⟩ All strings that satisfy *all* of the following conditions:

(a) the number of 0s is even

(b) the number of 1s is divisible by 3

(c) the total length is divisible by 5

6.2 All strings that satisfy *at least one* of the following conditions: ...

6.3 All strings that satisfy *exactly one* of the following conditions: ...

6.4 All strings that satisfy *exactly two* of the following conditions: ...

6.5 All strings that satisfy *an odd number of* of the following conditions: ...

- Other possible conditions:

  (a) The number of 0s in w is odd.

  (b) The number of 1s in w is not divisible by 3. ⟨⟨***HW***⟩⟩

  (c) The length $|w|$ is divisible by 5.

  (d) The binary value of w is not divisible by 7. ⟨⟨***HW***⟩⟩

  (e) The binary value of w^R is divisible by 9.

  (f) w contains the substring 00.

  (g) w does not contain the substring 11.

  (h) w contains the substring 01 an odd number of times. ⟨⟨***HW***⟩⟩

  (i) ww does not contain the substring 101.

Regular Language Transformations

Let L be an arbitrary regular language over the alphabet $\Sigma = \{0, 1\}$. Prove that each of the following languages over $\{0, 1\}$ is regular.

7.1 $L^c := \left\{w^c \mid w \in L\right\}$, where w^c is the complement of w, defined recursively as follows:

$$w^c := \begin{cases} \varepsilon & \text{if } w = \varepsilon \\ 1 \cdot x^c & \text{if } w = 0x \text{ for some string } x \\ 0 \cdot x^c & \text{if } w = 1x \text{ for some string } x \end{cases}$$

For example, $0001101^c = 1110010$.

7.2 $\textsc{OneInFront}(L) := \{1x \mid x \in L\}$

7.3 $\textsc{OneInBack}(L) := \{x1 \mid x \in L\}$

7.4 $\textsc{OnlyOnes}(L) := \left\{1^{\#(1,w)} \mid w \in L\right\}$

7.5 $\textsc{PadWithZeros}(L) := \left\{w \mid 1^{\#(1,w)} \in L\right\}$

7.6 $\textsc{MissingFirstOne}(L) := \{w \in \Sigma^* \mid 1w \in L\}$

7.7 $\textsc{MissingLastOne}(L) := \{w \in \Sigma^* \mid w1 \in L\}$

7.8 $\textsc{Prefixes}(L) := \{x \mid xy \in L \text{ for some } y \in \Sigma^*\}$

7.9 ***⟨⟨F16⟩⟩*** $\textsc{Suffixes}(L) := \{y \mid xy \in L \text{ for some } x \in \Sigma^*\}$

7.10 $\textsc{Rotations}(L) := \{yx \mid xy \in L\}$

7.11 ***⟨⟨lab, F14⟩⟩*** $\textsc{Evens}(L) := \{evens(w) \mid w \in L\}$, where the functions *evens* and *odds* are recursively defined as follows:

$$evens(w) := \begin{cases} \varepsilon & \text{if } w = \varepsilon \\ odds(x) & \text{if } w = ax \end{cases} \qquad odds(w) := \begin{cases} \varepsilon & \text{if } w = \varepsilon \\ a \cdot evens(x) & \text{if } w = ax \end{cases}$$

For example, $evens(0001101) = 010$ and $odds(0001101) = 0011$.

7.12 ***⟨⟨lab, F14⟩⟩*** $\textsc{Evens}^{-1}(L) := \{w \mid evens(w) \in L\}$, where the functions *evens* and *odds* are recursively defined as above.

7.13 $\textsc{Faro}(L) := \left\{faro(w, x) \mid w, x \in L\right\}$, where the function *faro* is defined recursively as follows:

$$faro(w, x) := \begin{cases} x & \text{if } w = \varepsilon \\ a \cdot faro(x, y) & \text{if } w = ay \text{ for some } a \in \Sigma \text{ and some } y \in \Sigma^* \end{cases}$$

For example, $faro(0001101, 1111) = 0^1 0^1 0^1 1^1 101$

7.14 $\textsc{Scramble}(L) := \left\{scramble(w) \mid w \in L\right\}$, where the function *scramble* is defined recursively as follows:

$$scramble(w) := \begin{cases} w & \text{if } |w| \le 1 \\ ba \cdot scramble(x) & \text{if } w = abx \text{ for some } a, b \in \Sigma \text{ and } x \in \Sigma^* \end{cases}$$

For example, $scramble(00\,01\,10\,1) = 00\,10\,01\,1$.

7.15 ⟨⟨*S18*⟩⟩ $\textsc{OddParity}(L) := \{w \in L \mid \textit{parity}(w) = 1\}$, where

$$\textit{parity}(w) = \begin{cases} 0 & \text{if } \#(w, 1) \text{ is even} \\ 1 & \text{if } \#(w, 1) \text{ is odd} \end{cases}$$

7.16 $\textsc{EvenParity}(L) := \{w \in L \mid \textit{parity}(w) = 0\}$, where $\textit{parity}(w)$ is defined as above.

7.17 ⟨⟨*S18*⟩⟩ $\textsc{AddParityFront}(L) := \{\textit{parity}(w) \cdot w \mid w \in L\}$, where $\textit{parity}(w)$ is defined as above.

7.18 $\textsc{AddParityEnd}(L) := \{w \bullet \textit{parity}(w) \mid w \in L\}$, where $\textit{parity}(w)$ is defined as above.

7.19 ⟨⟨*S18*⟩⟩ $\textsc{StripInit0s}(L) = \left\{w \;\middle|\; 0^j w \in L \text{ for some } j \geq 0\right\}$

Context-Free Grammars

Construct context-free grammars for each of the following languages, and give a *brief* explanation of how your grammar works, including the language of each non-terminal. We explicitly do not want a formal proof of correctness.

8.1 All strings in $\{0, 1\}^*$ whose length is divisible by 5.

8.2 All strings in which the substrings 01 and 01 appear the same number of times.

8.3 $\{0^n 1^{2n} \mid n \geq 0\}$

8.4 $\{0^m 1^n \mid n \neq 2m\}$

8.5 $\{0^i 1^j 0^{i+j} \mid i, j \geq 0\}$ ***⟨⟨HW⟩⟩***

8.6 $\{0^{i+j} 1 0^j 1 0^i \mid i, j \geq 0\}$

8.7 $\{0^i 1^j 2^k \mid j \neq i + k\}$

8.8 $\left\{w\#0^{\#(0,w)} \;\middle|\; w \in \{0, 1\}^*\right\}$

8.9 $\{0^i 1^j 2^k \mid i = j \text{ or } j = k \text{ or } i = k\}$

8.10 $\left\{0^i 1^j 2^k \;\middle|\; i = j \text{ or } j = k\right\}$. ***⟨⟨S18⟩⟩***

8.11 $\{0^i 1^j 2^k \mid i \neq j \text{ or } j \neq k\}$

8.12 $\{0^{2i} 1^{i+j} 2^{2j} \mid i, j \geq 0\}$

8.13 $\left\{x\#y^R \;\middle|\; x, y \in \{0, 1\}^* \text{ and } x \neq y\right\}$

8.14 All strings in $\{0, 1\}^*$ that are *not* palindromes.

8.15 $\{0, 1\}^* \setminus \{ww \mid w \in \{0, 1\}^*\}$ ***⟨⟨lab⟩⟩***

8.16 $\left\{0^n 1^{an+b} \;\middle|\; n \geq 0\right\}$, where a and b are arbitrary natural numbers.

8.17 $\left\{0^n 1^{an-b} \;\middle|\; n \geq b/a\right\}$, where a and b are arbitrary natural numbers.

True or False (sanity check)

For each statement below, check "True" if the statement is ***always*** true and "False" 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.

Read each statement *very* carefully. Some of these are deliberately subtle. On the other hand, you should not spend more than two minutes on any single statement.

Definitions

A.1 Every language is regular.

A.2 Every finite language is regular.

A.3 Every infinite language is regular. ***⟨⟨S18⟩⟩***

A.4 No infinite language is regular. ***⟨⟨S18⟩⟩***

A.5 If L is regular then L can be represented by a regular expression.

A.6 If L is not regular then L cannot be represented by a regular expression.

A.7 If L can be represented by a regular expression, then L is regular.

A.8 If L cannot be represented by a regular expression, then L is not regular.

A.9 If there is a DFA that accepts every string in L, then L is regular.

A.10 If there is a DFA that accepts every string not in L, then L is not regular.

A.11 If there is a DFA that rejects every string not in L, then L is regular.

A.12 If for every string $w \in L$ there is a DFA that accepts w, then L is regular. ***⟨⟨S14⟩⟩***

A.13 If for every string $w \notin L$ there is a DFA that rejects w, then L is regular.

A.14 If L is not regular, then for every string $w \in L$, there is a DFA that accepts w. ***⟨⟨S18⟩⟩***

A.15 If L is regular, then for every string $w \in L$, there is a DFA that rejects w. ***⟨⟨S18⟩⟩***

A.16 If some DFA recognizes L, then some NFA also recognizes L.

A.17 If some NFA recognizes L, then some DFA also recognizes L.

A.18 If some NFA with ε-transitions recognizes L, then some NFA without ε-transitions also recognizes L.

Closure Properties of Regular Languages

B.1 For all regular languages L and L', the language $L \cap L'$ is regular.

B.2 For all regular languages L and L', the language $L \cup L'$ is regular.

B.3 For all regular languages L, the language L^* is regular.

B.4 For all regular languages A, B, and C, the language $(A \cup B) \setminus C$ is regular.

B.5 For all languages $L \subseteq \Sigma^*$, if L is regular, then $\Sigma^* \setminus L$ is regular.

B.6 For all languages $L \subseteq \Sigma^*$, if L is regular, then $\Sigma^* \setminus L$ is not regular.

B.7 For all languages $L \subseteq \Sigma^*$, if L is not regular, then $\Sigma^* \setminus L$ is regular.

B.8 For all languages $L \subseteq \Sigma^*$, if L is not regular, then $\Sigma^* \setminus L$ is not regular.

B.9 ⟨⟨***S14***⟩⟩ For all languages L and L', the language $L \cap L'$ is regular.

B.10 ⟨⟨***F14***⟩⟩ For all languages L and L', the language $L \cup L'$ is regular.

B.11 For all languages L, the language L^* is regular. ⟨⟨***F14, F16***⟩⟩

B.12 For all languages L, if L^* is regular, then L is regular.

B.13 For all languages A, B, and C, the language $(A \cup B) \setminus C$ is regular.

B.14 For all languages L, if L is finite, then L is regular.

B.15 For all languages L and L', if L and L' are finite, then $L \cup L'$ is regular.

B.16 For all languages L and L', if L and L' are finite, then $L \cap L'$ is regular.

B.17 For all languages $L \subseteq \Sigma^*$, if L contains infinitely many strings in Σ^*, then L is not regular.

B.18 ⟨⟨***S14***⟩⟩ For all languages $L \subseteq \Sigma^*$, if L contains all but a finite number of strings of Σ^*, then L is regular.

B.19 For all languages $L \subseteq \{0, 1\}^*$, if L contains a finite number of strings in 0^*, then L is regular.

B.20 For all languages $L \subseteq \{0, 1\}^*$, if L contains all but a finite number of strings in 0^*, then L is regular.

B.21 If L and L' are not regular, then $L \cap L'$ is not regular.

B.22 If L and L' are not regular, then $L \cup L'$ is not regular.

B.23 If L is regular and $L \cup L'$ is regular, then L' is regular. ⟨⟨***S14***⟩⟩

B.24 If L is regular and $L \cup L'$ is not regular, then L' is not regular. ⟨⟨***S14***⟩⟩

B.25 If L is not regular and $L \cup L'$ is regular, then L' is regular.

B.26 If L is regular and $L \cap L'$ is regular, then L' is regular.

B.27 If L is regular and $L \cap L'$ is not regular, then L' is not regular.

B.28 If L is regular and L' is finite, then $L \cup L'$ is regular. ⟨⟨***S14***⟩⟩

B.29 If L is regular and L' is finite, then $L \cap L'$ is regular.

B.30 If L is regular and $L \cap L'$ is finite, then L' is regular.

B.31 If L is regular and $L \cap L' = \varnothing$, then L' is regular. ⟨⟨*S18*⟩⟩

B.32 If L is regular and $L \cap L' = \varnothing$, then L' is not regular. ⟨⟨*S18*⟩⟩

B.33 If L is not regular and $L \cap L' = \varnothing$, then L' is regular. ⟨⟨*F16*⟩⟩

B.34 If L is regular and L' is not regular, then $L \cap L' = \varnothing$.

B.35 If $L \subseteq L'$ and L is regular, then L' is regular.

B.36 If $L \subseteq L'$ and L' is regular, then L is regular. ⟨⟨*F14*⟩⟩

B.37 If $L \subseteq L'$ and L is not regular, then L' is not regular.

B.38 If $L \subseteq L'$ and L' is not regular, then L is not regular. ⟨⟨*F14*⟩⟩

B.39 For every regular language L, the language $\{\mathtt{0}^{|w|} \mid w \in L\}$ is also regular. ⟨⟨*S18*⟩⟩

B.40 For every non-regular language L, the language $\{\mathtt{0}^{|w|} \mid w \in L\}$ is also non-regular. ⟨⟨*S18*⟩⟩

B.41 For all languages $L \subseteq \Sigma^*$, if L cannot be described by a regular expression, then some DFA accepts $\Sigma^* \setminus L$.

B.42 For all languages $L \subseteq \Sigma^*$, if no DFA accepts L, then the complement $\Sigma^* \setminus L$ can be described by a regular expression.

B.43 For all languages $L \subseteq \Sigma^*$, if no DFA accepts L, then the complement $\Sigma^* \setminus L$ cannot be described by a regular expression.

B.44 For all languages $L \subseteq \Sigma^*$, if L is recognized by a DFA, then $\Sigma^* \setminus L$ can be described by a regular expression. ⟨⟨*F16*⟩⟩

Properties of Context-free Languages

C.1 If L cannot be recognized by a DFA, then L is context-free.

C.2 If L cannot be recognized by a DFA, then L is not context-free.

C.3 If L can be recognized by a DFA, then L is context-free.

C.4 If L can be recognized by a DFA, then L is not context-free.

C.5 If L is not context-free, then L is regular.

C.6 If L is not context-free, then $\Sigma^* \setminus L$ is regular.

C.7 If L is not context-free, then L is not regular.

C.8 If L is not context-free, then $\Sigma^* \setminus L$ is not regular.

C.9 Every finite language is context-free.

C.10 Every context-free language is regular. ⟨⟨*F14*⟩⟩

C.11 Every regular language is context-free.

C.12 Every infinite language is context-free.

C.13 Every non-context-free language is non-regular. ⟨⟨*F16*⟩⟩

C.14 For all context-free languages L and L', the language $L \bullet L'$ is also context-free. ⟨⟨*F16*⟩⟩

C.15 For every context-free language L, the language L^* is also context-free.

C.16 For all context-free languages A, B, and C, the language $(A \cup B)^* \bullet C$ is also context-free.

C.17 For every language L, the language L^* is context-free.

C.18 For every language L, if L^* is context-free then L is context-free.

C.19 For every context-free language L, the language $\left\{0^{|w|} \mid w \in L\right\}$ is also context-free. ⟨⟨*S18*⟩⟩

Equivalence Classes. Recall that for any language $L \subset \Sigma^*$, two strings $x, y \in \Sigma^*$ are equivalent with respect to L if and only if, for every string $z \in \Sigma^*$, either both xz and yz are in L, or neither xz nor yz is in L—or more concisely, if x and y have no distinguishing suffix with respect to L. We denote this equivalence by $x \equiv_L y$.

D.1 For all languages L, if L is regular, then $\equiv_L$ has finitely many equivalence classes.

D.2 For all languages L, if L is not regular, then $\equiv_L$ has infinitely many equivalence classes. ⟨⟨*S14*⟩⟩

D.3 For all languages L, if $\equiv_L$ has finitely many equivalence classes, then L is regular.

D.4 For all languages L, if $\equiv_L$ has infinitely many equivalence classes, then L is not regular.

D.5 For all regular languages L, each equivalence class of $\equiv_L$ is a regular language.

D.6 For all languages L, each equivalence class of $\equiv_L$ is a regular language.

Fooling Sets

E.1 If L has an infinite fooling set, then L is not regular.

E.2 If L has an finite fooling set, then L is regular.

E.3 If L does not have an infinite fooling set, then L is regular.

E.4 If L is not regular, then L has an infinite fooling set.

E.5 If L is regular, then L has no infinite fooling set.

E.6 If L is not regular, then L has no finite fooling set. ⟨⟨*F14, F16*⟩⟩

E.7 If L is context-free and L has a finite fooling set, then L is regular. ⟨⟨*S18*⟩⟩

E.8 If L is context-free and L has a finite fooling set, then L is not regular. ⟨⟨*S18*⟩⟩

E.9 If L is context-free and L has an infinite fooling set, then L is not regular.

E.10 If L is not context-free and L has no infinite fooling set, then L is not regular. *[Hint: Careful!]*

Specific Languages (Gut Check). Do *not* construct complete DFAs, NFAs, regular expressions, or fooling-set arguments for these languages. You don't have time.

F.1 $\{0^i1^j0^k \mid i+j-k=374\}$ is regular. *⟨⟨S14⟩⟩*

F.2 $\{0^i1^j0^k \mid i+j-k \geq 374\}$ is regular. *⟨⟨S18⟩⟩*

F.3 $\{0^i1^j0^k \mid i+j+k=374\}$ is regular.

F.4 $\{0^i1^j0^k \mid i+j+k \geq 374\}$ is regular. *⟨⟨S18⟩⟩*

F.5 $\{0^i1^j \mid i<374<j\}$ is regular. *⟨⟨S14⟩⟩*

F.6 $\left\{0^i1^j \mid 0 \leq i+j \leq 374\right\}$ is regular. *⟨⟨F14⟩⟩*

F.7 $\left\{0^i1^j \mid 0 \leq i-j \leq 374\right\}$ is regular. *⟨⟨F14⟩⟩*

F.8 $\{0^i1^j \mid i,j \geq 0\}$ is regular. *⟨⟨F16⟩⟩*

F.9 $\{0^i1^j \mid (i-j)$ is divisible by 374$\}$ is regular. *⟨⟨S14⟩⟩*

F.10 $\{0^i1^j \mid (i+j)$ is divisible by 374$\}$ is regular.

F.11 $\left\{0^{n^2} \mid n \geq 0\right\}$ is regular.

F.12 $\left\{0^{37n+4} \mid n \geq 0\right\}$ is regular.

F.13 $\left\{0^n10^n \mid n \geq 0\right\}$ is regular.

F.14 $\left\{0^m10^n \mid m \geq 0 \text{ and } n \geq 0\right\}$ is regular.

F.15 $\{w \in \{0,1\}^* \mid |w|$ is divisible by 374$\}$ is regular.

F.16 $\{w \in \{0,1\}^* \mid w$ represents a integer divisible by 374 in binary$\}$ is regular.

F.17 $\{w \in \{0,1\}^* \mid w$ represents a integer divisible by 374 in base 473$\}$ is regular.

F.18 $\left\{w \in \{0,1\}^* \mid |\#(0,w)-\#(1,w)| < 374\right\}$ is regular.

F.19 $\left\{w \in \{0,1\}^* \mid |\#(0,x)-\#(1,x)| < 374 \text{ for every prefix } x \text{ of } w\right\}$ is regular.

F.20 $\left\{w \in \{0,1\}^* \mid |\#(0,x)-\#(1,x)| < 374 \text{ for every substring } x \text{ of } w\right\}$ is regular.

F.21 $\left\{w0^{\#(0,w)} \mid w \in \{0,1\}^*\right\}$ is regular.

F.22 $\left\{w0^{\#(0,w) \bmod 374} \mid w \in \{0,1\}^*\right\}$ is regular.

Automata Transformations

G.1 Let $M = (\Sigma, Q, s, A, \delta)$ and $M' = (\Sigma, Q, s, Q \setminus A, \delta)$ be arbitrary **DFAs** with identical alphabets, states, starting states, and transition functions, but with complementary accepting states. Then $L(M) \cap L(M') = \varnothing$. ⟨⟨*F16*⟩⟩

G.2 Let $M = (\Sigma, Q, s, A, \delta)$ and $M' = (\Sigma, Q, s, Q \setminus A, \delta)$ be arbitrary **NFAs** with identical alphabets, states, starting states, and transition functions, but with complementary accepting states. Then $L(M) \cap L(M') = \varnothing$. ⟨⟨*F16*⟩⟩

G.3 Let M be a **DFA** over the alphabet Σ. Let M' be identical to M, except that accepting states in M are non-accepting in M' and vice versa. Each string in Σ^* is accepted by exactly one of M and M'.

G.4 Let M be an **NFA** over the alphabet Σ. Let M' be identical to M, except that accepting states in M are non-accepting in M' and vice versa. Each string in Σ^* is accepted by exactly one of M and M'.

G.5 Let $M = (\Sigma, Q, s, A, \delta)$ be an arbitrary **DFA** over the alphabet $\Sigma = \{0, 1\}$, and let $M' = (\Sigma, Q, s, A, \delta')$ be the DFA obtained from M by changing every 0-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 $\delta'(q, 0) = \delta(q, 1)$ and $\delta'(q, 1) = \delta(q, 0)$ for every state q. Then $L(M) \cup L(M') = \{0, 1\}^*$. ⟨⟨*S18*⟩⟩

G.6 Let $M = (\Sigma, Q, s, A, \delta)$ be an arbitrary **DFA** over the alphabet $\Sigma = \{0, 1\}$, and let $M' = (\Sigma, Q, s, A, \delta')$ be the DFA obtained from M by changing every 0-transition into a 1-transition and vice versa, as in the previous question. Then $L(M) \cap L(M') = \varnothing$. ⟨⟨*S18*⟩⟩

G.7 Let $M = (\Sigma, Q, s, A, \delta)$ be an arbitrary **NFA** over the alphabet $\Sigma = \{0, 1\}$, and let $M' = (\Sigma, Q, s, A, \delta')$ be the NFA obtained from M by changing every 0-transition into a 1-transition and vice versa, as in the two previous questions. Then $L(M) \cap L(M') = \varnothing$.

G.8 Let $M = (\Sigma, Q, s, A, \delta)$ be an arbitrary NFA, and $M' = (\Sigma, Q', s, A', \delta')$ be any NFA obtained from M by deleting some subset of the states. More formally, $Q' \subseteq Q$ and $A' = A \cap Q'$ and $\delta'(q, a) = \delta(q, a) \cap Q'$ for every state $q \in Q'$. Then $L(M') \subseteq L(M)$. ⟨⟨*S18*⟩⟩

G.9 If a language L is recognized by a DFA with n states, then the complementary language $\Sigma^* \setminus L$ is recognized by a DFA with at most $n + 1$ states.

G.10 If a language L is recognized by an NFA with n states, then the complementary language $\Sigma^* \setminus L$ is recognized by a NFA with at most $n + 1$ states.

G.11 If a language L is recognized by a DFA with n states, then L^* is recognized by a DFA with at most $n + 1$ states.

G.12 If a language L is recognized by an NFA with n states, then L^* is recognized by a NFA with at most $n + 1$ states.

Language Transformations

H.1 For every regular language L, the language $\left\{w^R \mid w \in L\right\}$ is also regular.

H.2 For every language L, if the language $\left\{w^R \mid w \in L\right\}$ is regular, then L is also regular. *⟨⟨F14⟩⟩*

H.3 For every language L, if the language $\left\{w^R \mid w \in L\right\}$ is not regular, then L is also not regular. *⟨⟨F14⟩⟩*

H.4 For every regular language L, the language $\left\{w \mid ww^R \in L\right\}$ is also regular.

H.5 For every regular language L, the language $\left\{ww^R \mid w \in L\right\}$ is also regular.

H.6 For every language L, if the language $\left\{w \mid ww^R \in L\right\}$ is regular, then L is also regular. *[Hint: Consider the language $L = \{0^n 1^n \mid n \geq 0\}$.]*

H.7 For every regular language L, the language $\left\{0^{|w|} \mid w \in L\right\}$ is also regular.

H.8 For every language L, if the language $\left\{0^{|w|} \mid w \in L\right\}$ is regular, then L is also regular.

H.9 For every context-free language L, the language $\left\{w^R \mid w \in L\right\}$ is also context-free.