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 handout. 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 past exams are labeled with the semester they were used, for example, ⟨⟨S14⟩⟩ or ⟨⟨F14⟩⟩. (Questions from old exams might reappear on this semester’s exams, but they might not.) Questions from this semester’s homework are labeled ⟨⟨HW⟩⟩. Questions from this semester’s labs are labeled ⟨⟨Lab⟩⟩.

1. 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 ⌊⌋ functions are formally defined as follows:

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

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

1. The reversal \( w^R \) of a string \( w \) is defined recursively as follows:

\[ w^R := \begin{cases} \epsilon & \text{if } w = \epsilon \\ x^R \cdot a & \text{if } w = ax \text{ for some } a \in \Sigma \text{ and } x \in \Sigma^* \end{cases} \]

1.A. Prove that \((w \cdot x)^R = x^R \cdot w^R\) for all strings \(w\) and \(x\). ⟨⟨lab, F14⟩⟩

1.B. Prove that \((w^R)^R = w\) for every string \(w\). ⟨⟨lab⟩⟩

1.C. Prove that \(|w| = |w^R|\) for every string \(w\). ⟨⟨lab⟩⟩

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

2. For any string \(w\) and any non-negative integer \(n\), let \(w^n\) denote the string obtained by concatenating \(n\) copies of \(w\); more formally, define

\[ w^n := \begin{cases} \epsilon & \text{if } n = 0 \\ w \cdot w^{n-1} & \text{otherwise} \end{cases} \]

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

2.A. Prove that \(w^m \cdot w^n = w^{m+n}\) for every string \(w\) and all non-negative integers \(n\) and \(m\).
2.B. Prove that \((w^m)^n = w^{mn}\) for every string \(w\) and all non-negative integers \(n\) and \(m\).

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

3. Consider the following pair of mutually recursive functions:

\[
\begin{align*}
evens(w) := & \begin{cases} 
\varepsilon & \text{if } w = \varepsilon \\
odds(x) & \text{if } w = ax 
\end{cases} \\
odds(w) := & \begin{cases} 
\varepsilon & \text{if } w = \varepsilon \\
a \cdot evens(x) & \text{if } w = ax
\end{cases}
\end{align*}
\]

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

3.A. Prove the following identity for all strings \(w\) and \(x\):

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

3.B. Prove the following identity for all strings \(w\):

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

3.C. Prove that \(|w| = |evens(w)| + |odds(w)|\) for every string \(w\).

4. Consider the following recursive function:

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

For example, \(scramble(0001101) = 0010011\).

4.A. Prove that \(|scramble(w)| = |w|\) for every string \(w\).

4.B. Prove that \(scramble(scramble(w)) = w\) for every string \(w\).

2. Regular expressions

For each of the following languages over the alphabet \(\{0, 1\}\), give an equivalent regular expression.

5. Every string of length at most 3. (Hint: Don’t try to be clever.)

6. ⟨⟨lab⟩⟩ Every string except 010. (Hint: Don’t try to be clever.)

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

8. ⟨⟨hw⟩⟩ All strings not containing the substring 010.

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

10. All strings containing either the substring 10 or the substring 01, but not both.
11. All strings containing at least two 1s and at least one 0.
12. All strings containing either at least two 1s or at least one 0.
13. All strings such that in every prefix, the number of 0s and the number of 1s differ by at most 1.
14. The set of all strings in \{0, 1\}^* whose length is divisible by 3.
15. The set of all strings in 0^*1^* whose length is divisible by 3.
16. The set of all strings in \{0, 1\}^* in which the number of 1s is divisible by 3.

3. Direct DFA construction.

Draw or formally describe a DFA that recognizes each of the following languages. If you draw the DFA you may omit transitions to a reject/junk state.

17. Every string of length at most 3.
18. Every string except 010.
19. The language \{LONG, LUG, LEGO, LEG, LUG, LOG, LINGO\}.
20. The language \textit{MOO}^* + \textit{MEOO}^*\textit{W}
21. All strings in which every run of consecutive 0s has even length and every run of consecutive 1s has odd length. \textit{⟨⟨F14⟩⟩}
22. All strings not containing the substring 010.
23. All strings containing the substring 10 or the substring 01.
24. All strings containing either the substring 10 or the substring 01, but not both.
25. The set of all strings in \{0, 1\}^* whose length is divisible by 3.
26. The set of all strings in 0^*1^* whose length is divisible by 3.
27. The set of all strings in \{0, 1\}^* in which the number of 1s is divisible by 3.
28. All strings \(w\) such that the binary value of \(w^R\) is divisible by 5.
29. All strings such that in every prefix, the number of 0s and the number of 1s differ by at most 2.

4. Fooling sets

Prove that each of the following languages is \textit{not} regular.

30. The set of all strings in \{0, 1\}^* with more 0s than 1s. \textit{⟨⟨S14⟩⟩}
31. The set of all strings in \{0, 1\}^* with fewer 0s than 1s.
32. The set of all strings in \{0, 1\}^* with exactly twice as many 0s as 1s.
33. The set of all strings in \( \{0, 1\}^* \) with at least twice as many 0s as 1s.

34. \( \{0^{2n} \mid n \geq 0\} \) (Lab)

35. \( \{0^{F_n} \mid n \geq 0\} \), where \( F_n \) is the \( n \)th 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 \( \min\{i, j\} \leq 2 \).) (Hard.)

36. \( \{0^n^3 \mid n \geq 0\} \)

37. \( \{x#y \mid x, y \in \{0, 1\}^* \text{ and } \#(0, x) = \#(1, y)\} \)

38. \( \{xx^c \mid x \in \{0, 1\}^*\} \), where \( x^c \) is the complement of \( x \), obtained by replacing every 0 in \( x \) with a 1 and vice versa. For example, 0011011 = 1110010.

39. The language of properly balanced strings of parentheses, described by the context-free grammar

\[
S \rightarrow \epsilon \mid SS \mid (S)
\]

(\( \langle \langle \text{Lab} \rangle \rangle \))

40. \( \{(01)^n(10)^n \mid n \geq 0\} \)

41. \( \{(01)^m(10)^n \mid n \geq m \geq 0\} \)

42. \( \{w#x#y \mid w, x, y \in \Sigma^* \text{ and } w, x, y \text{ are not all equal}\} \)

5. 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 alphabets are 0, 1.

43. (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.)

44. (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.)

45. (F14) \( \{www \mid w \in \Sigma^*\} \)

46. (F14) \( \{wxw \mid w, x \in \Sigma^*\} \)

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

48. 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.

49. \( \{0^m1^n \mid 0 \leq m - n \leq 374\} \)
50. \( \{0^m1^n \mid 0 \leq m + n \leq 374\} \)

51. The language generated by the following context-free grammar:

\[
S \rightarrow 0A1 \mid \epsilon \\
A \rightarrow 1S0 \mid \epsilon
\]

52. The language generated by the following context-free grammar:

\[
S \rightarrow 0S1 \mid 1S0 \mid \epsilon
\]

53. \( \{w#x \mid w, x \in \{0, 1\}^* \text{ and no substring of } w \text{ is also a substring of } x\} \)

54. \( \{w#x \mid w, x \in \{0, 1\}^* \text{ and no non-empty substring of } w \text{ is also a substring of } x\} \)

(Hard.)

55. \( \{w#x \mid w, x \in \{0, 1\}^* \text{ and every non-empty substring of } w \text{ is also a substring of } x\} \)

56. \( \{w#x \mid w, x \in \{0, 1\}^* \text{ and } w \text{ is a substring of } x\} \)

57. \( \{w#x \mid w, x \in \{0, 1\}^* \text{ and } w \text{ is a proper substring of } x\} \)

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

(Hard.)

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

6. 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. Do not just describe several smaller DFAs and write product construction!

60. \( \langle S14 \rangle \) All strings that satisfy all of the following conditions:

60.A. the number of 0s is even
60.B. the number of 1s is divisible by 3
60.C. the total length is divisible by 5

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

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

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

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

65. Other possible conditions:
65.A. The number of 0s in $w$ is odd.
65.B. The number of 1s in $w$ is not divisible by 5.
65.C. The length $|w|$ is divisible by 7.
65.D. The binary value of $w$ is divisible by 7.
65.E. The binary value of $w^R$ is not divisible by 7.
65.F. $w$ contains the substring 00
65.G. $w$ does not contain the substring 11
65.H. $ww$ does not contain the substring 101

7. NFA Construction

Let $L$ be an arbitrary regular language $\Sigma = \{0, 1\}$. Prove that each of the following languages over $\{0, 1\}$ is regular. Describe does not necessarily mean draw.

66. All strings where the 374th symbol from the end is 0.

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

67.A. The number of 0s is even
67.B. The number of 1s is divisible by 3
67.C. The total length is divisible by 5

68. $\langle\langle \text{lab} \rangle\rangle$ All strings such that in every prefix, the number of 0s and the number of 1s differ by at most 2.

69. $\langle\langle \text{lab} \rangle\rangle$ All strings such that in every substring, the number of 0s and the number of 1s differ by at most 2.

8. 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. Describe does not necessarily mean draw.

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

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

For example, $0001101^c = 1110010$.

71. $\text{OneInFront}(L) := \{1x \mid x \in L\}$

72. $\text{OnlyOnes}(L) := \{1^{\#(1,w)} \mid w \in L\}$
73. \( \text{ONLYONES}^{-1}(L) := \{ w \mid 1^{\#_1(w)} \in L \} \)

74. \( \text{MISSINGFIRST}(L) := \{ w \in \Sigma^* \mid aw \in L \text{ for some symbol } a \in \Sigma \} \)

75. The language \( \text{Prefixes}(L) := \{ x \mid xy \in L \text{ for some string } x \in \Sigma^* \} \).

76. \( \text{Suffixes}(L) := \{ y \mid xy \in L \text{ for some string } y \in \Sigma^* \} \)

77. \( \langle \langle \text{lab}, F_14 \rangle \rangle \text{ Evens}(L) := \{ \text{evens}(w) \mid w \in L \} \), where the functions \( \text{evens} \) and \( \text{odds} \) are recursively defined as follows:

\[
\text{evens}(w) := \begin{cases} 
\epsilon & \text{if } w = \epsilon \\
\text{odds}(x) & \text{if } w = ax 
\end{cases}
\]
\[
\text{odds}(w) := \begin{cases} 
\epsilon & \text{if } w = \epsilon \\
a \cdot \text{evens}(x) & \text{if } w = ax 
\end{cases}
\]

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

78. \( \langle \langle \text{lab}, F_14 \rangle \rangle \text{ Evens}^{-1}(L) := \{ w \mid \text{evens}(w) \in L \} \), where the functions \( \text{evens} \) and \( \text{odds} \) are recursively defined as above.

79. \( \text{Shuffle}(L) := \{ \text{shuffle}(w, x) \mid w, x \in L \} \), where the function \( \text{shuffle} \) is defined recursively as follows:

\[
\text{shuffle}(w, x) := \begin{cases} 
x & \text{if } w = \epsilon \\
ax \cdot \text{shuffle}(x, y) & \text{if } w = ay \text{ for some } a \in \Sigma \text{ and some } y \in \Sigma^* 
\end{cases}
\]

For example, \( \text{shuffle}(0001101, 1111) = 01011011011 \).

80. \( \text{Scramble}(L) := \{ \text{scramble}(w) \mid w \in L \} \), where the function \( \text{scramble} \) is defined recursively as follows:

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

For example, \( \text{scramble}(00010101) = 0010011 \).

9. 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.

81. All strings in \( \{0, 1\}^* \) whose length is divisible by 5.
82. All strings in which the substrings 00 and 11 appear the same number of times.
83. All strings in which the substrings 01 and 01 appear the same number of times.
84. \( \{0^n1^{2n} \mid n \geq 0\} \)
85. \( \{0^n1^n \mid n \neq 2m\} \)
86. \( \{0^i1^j2^{i+j} \mid i, j \geq 0\} \)
87. \( \{0^{i+j} \#0^j \#0^i \mid i, j \geq 0\} \)
88. \( \{0^i1^j2^k \mid j \neq i + k\} \)
89. \( \{w \# (0, w) \mid w \in \{0, 1\}^*\} \)
90. \( \{0^i1^j2^k \mid i = j \text{ or } j = k \text{ or } i = k\} \)
91. \( \{0^i1^j2^k \mid i \neq j \text{ or } j \neq k\} \)
92. \( \{0^{2i}1^i2^j \mid i, j \geq 0\} \)
93. \( \{0^i1^j2^k \mid i \neq j \text{ or } j \neq k\} \)
94. \( \{0^i1^j2^k \mid i \neq j \text{ or } j \neq k\} \)
95. \( \{0^i1^j2^k \mid i \geq 0\} \), where \( a \) and \( b \) are arbitrary natural numbers.
96. \( \{0^i1^j2^k \mid i \geq b/a\} \), where \( a \) and \( b \) are arbitrary natural numbers.

10. Context-Free Grammar Proofs

Each of the following questions describes a language \( L \) and a context-free grammar \( G \), and asks you to prove that \( L = L(G) \). As always, you must separately prove \( L \subseteq L(G) \) and \( L(G) \subseteq L \); each proof will proceed by induction.

98. Prove that the following grammar generates the language \( \{0^n1^n \mid n \geq 0\} \).
   \[
   S \rightarrow 0S1 | \varepsilon
   \]

99. Prove that the following grammar generates the language \( \{0^m1^n \mid m \leq n\} \).
   \[
   S \rightarrow 0S1 | \varepsilon | S1
   \]

100. Prove that the following grammar generates the language \( \{0^m1^n \mid n \leq 2m \text{ and } m \leq 2n\} \).
    \[
    S \rightarrow 00S1 | 0S11 | 0S1 | \varepsilon.
    \]

101. Prove that the following grammar generates the language \( \{0^m1^n \mid n \leq 2m \text{ and } m \leq 2n\} \).
    \[
    S \rightarrow 00S1 | 0S11 | 0011 | 01 | \varepsilon.
    \]

102. Prove that the following grammar generates the language \( \{0^m1^n \mid n \leq 2m \text{ and } m \leq 2n\} \).
    \[
    S \rightarrow A | B
    A \rightarrow 00A1 | 0A1 | \varepsilon
    B \rightarrow 0B11 | 0B1 | \varepsilon.
    \]
103. Prove that the following grammar generates the language \( \{0^m1^n \mid n \leq 2m \text{ and } m \leq 2n \} \).

\[
\begin{align*}
S & \rightarrow A \mid B \\
A & \rightarrow 00A1 \mid C \\
B & \rightarrow 0B11 \mid C \\
C & \rightarrow 0C1 \mid \varepsilon
\end{align*}
\]

104. Prove that the following grammar generates the language \( \{0^m+0^n=0^{m+n} \mid m, n \geq 0 \} \).

\[
\begin{align*}
S & \rightarrow +A \mid 0S0 \\
A & \rightarrow = \mid 0A
\end{align*}
\]

105. Prove that the following grammar generates the language \( \{0^{2i+2j} \mid i, j \geq 0 \} \).

\[
\begin{align*}
S & \rightarrow AB \\
A & \rightarrow 00S1 \mid \varepsilon \\
B & \rightarrow 1S00 \mid \varepsilon
\end{align*}
\]

106. Prove that the following grammar generates the language of all binary strings \( w \) such that \( \#(0, w) = \#(1, w) \).

\[
S \rightarrow \varepsilon \mid 0S1S \mid 1S0S
\]

11. 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 \(-\frac{1}{2}\) point; checking “I don’t know” is worth \(\frac{1}{4}\) point; and flipping a coin is (on average) worth \(\frac{1}{4}\) 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

107. Every language is regular.

108. For all languages \( L \), if \( L \) is regular then \( L \) can be represented by a regular expression.

109. For all languages \( L \), if \( L \) is not regular then \( L \) cannot be represented by a regular expression.

110. For all languages \( L \), if \( L \) can be represented by a regular expression then \( L \) is regular.

111. For all languages \( L \), if \( L \) cannot be represented by a regular expression then \( L \) is not regular.

112. For all languages \( L \), if there is a DFA that accepts every string in \( L \), then \( L \) is regular.

113. For all languages \( L \), if there is a DFA that accepts every string not in \( L \), then \( L \) is not regular.

114. For all languages \( L \), if there is a DFA that rejects every string not in \( L \), then \( L \) is regular.
115. For all languages $L$, if for every string $w \in L$ there is a DFA that accepts $w$, then $L$ is regular. \( \langle S14 \rangle \)

116. For all languages $L$, if for every string $w \notin L$ there is a DFA that rejects $w$, then $L$ is regular.

117. For all languages $L$, if some DFA recognizes $L$, then some NFA also recognizes $L$.

118. For all languages $L$, if some NFA recognizes $L$, then some DFA also recognizes $L$.

### Closure Properties

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

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

121. For all regular languages $L$, the language $L^*$ is regular.

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

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

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

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

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

127. \( \langle S14 \rangle \) For all languages $L$ and $L'$, the language $L \cap L'$ is regular.

128. \( \langle F14 \rangle \) For all languages $L$ and $L'$, the language $L \cup L'$ is regular.

129. For all languages $L$, the language $L^*$ is regular. \( \langle F14 \rangle \)

130. For all languages $L$, if $L^*$ is regular, then $L$ is regular.

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

132. For all languages $L$, if $L$ is finite, then $L$ is regular.

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

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

135. For all languages $L$, if $L$ contains a finite number of strings, then $L$ is regular.

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

137. \( \langle S14 \rangle \) For all languages $L \subseteq \Sigma^*$, if $L$ contains all but a finite number of strings of $\Sigma^*$, then $L$ is regular.

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

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

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

141. If $L$ and $L'$ are not regular, then $L \cup L'$ is not regular.
142. If $L$ is regular and $L \cup L'$ is regular, then $L'$ is regular. \(\langle S14 \rangle\)
143. If $L$ is regular and $L \cup L'$ is not regular, then $L'$ is not regular. \(\langle S14 \rangle\)
144. If $L$ is not regular and $L \cup L'$ is regular, then $L'$ is regular.
145. If $L$ is regular and $L \cap L'$ is regular, then $L'$ is regular.
146. If $L$ is regular and $L \cap L'$ is not regular, then $L'$ is not regular.
147. If $L$ is regular and $L'$ is finite, then $L \cup L'$ is regular. \(\langle S14 \rangle\)
148. If $L$ is regular and $L'$ is finite, then $L \cap L'$ is regular.
149. If $L$ is regular and $L \cap L'$ is finite, then $L'$ is regular.
150. If $L \subseteq L'$ and $L$ is regular, then $L'$ is regular.
151. If $L$ is regular and $L'$ is not regular, then $L \cap L' = \emptyset$.
152. If $L \subseteq L'$ and $L$ is not regular, then $L'$ is not regular.
153. If $L \subseteq L'$ and $L'$ is regular, then $L$ is regular. \(\langle F14 \rangle\)
154. If $L \subseteq L'$ and $L$ is not regular, then $L'$ is not regular.
155. If $L \subseteq L'$ and $L'$ is not regular, then $L$ is not regular. \(\langle F14 \rangle\)
156. For all languages $L \subseteq \Sigma^*$, if $L$ cannot be described by a regular expression, then some DFA accepts $\Sigma^* \setminus L$.
157. For all languages $L \subseteq \Sigma^*$, if no DFA accepts $L$, then the complement $\Sigma^* \setminus L$ can be described by a regular expression.
158. Every context-free language is regular. \(\langle F14 \rangle\)
159. Every regular language is context-free.

**Equivalence Classes.** Recall that for any language $L \subseteq \Sigma^*$, 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 $L$. We denote this equivalence by $x \equiv_L y$.

160. For all languages $L$, if $L$ is regular, then $\equiv_L$ has finitely many equivalence classes.
161. For all languages $L$, if $L$ is not regular, then $\equiv_L$ has infinitely many equivalence classes. \(\langle S14 \rangle\)
162. For all languages $L$, if $\equiv_L$ has finitely many equivalence classes, then $L$ is regular.
163. For all languages $L$, if $\equiv_L$ has infinitely many equivalence classes, then $L$ is not regular.
164. For all regular languages $L$, each equivalence class of $\equiv_L$ is a regular language. \(\text{Hard.}\)
165. For all languages $L$, each equivalence class of $\equiv_L$ is a regular language.
Fooling Sets

166. For all languages $L$, if $L$ has an infinite fooling set, then $L$ is not regular.
167. For all languages $L$, if $L$ has a finite fooling set, then $L$ is regular.
168. For all languages $L$, if $L$ does not have an infinite fooling set, then $L$ is regular.
169. For all languages $L$, if $L$ is not regular, then $L$ has an infinite fooling set.
170. For all languages $L$, if $L$ is regular, then $L$ has no infinite fooling set.
171. For all languages $L$, if $L$ is not regular, then $L$ has no finite fooling set.

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

172. $\{0^i1^j2^k \mid i + j - k = 374\}$ is regular. 〈$S14$〉
173. $\{0^i1^j2^k \mid i + j - k \leq 374\}$ is regular.
174. $\{0^i1^j2^k \mid i + j + k = 374\}$ is regular.
175. $\{0^i1^j2^k \mid i + j + k > 374\}$ is regular.
176. $\{0^i1^j \mid i < 374 < j\}$ is regular. 〈$S14$〉
177. $\{0^n1^m \mid 0 \leq m + n \leq 374\}$ is regular. 〈$F14$〉
178. $\{0^n1^m \mid 0 \leq m - n \leq 374\}$ is regular. 〈$F14$〉
179. $\{0^i1^j \mid (i - j) \text{ is divisible by } 374\}$ is regular. 〈$S14$〉
180. $\{0^i1^j \mid (i + j) \text{ is divisible by } 374\}$ is regular.
181. $\{0^n^2 \mid n \geq 0\}$ is regular.
182. $\{0^{37n+4} \mid n \geq 0\}$ is regular.
183. $\{0^n10^n \mid n \geq 0\}$ is regular.
184. $\{0^n10^n \mid m \geq 0 \text{ and } n \geq 0\}$ is regular.
185. $\{w \in \{0, 1\}^* \mid \text{w is divisible by } 374\}$ is regular.
186. $\{w \in \{0, 1\}^* \mid w \text{ represents a integer divisible by } 374 \text{ in binary}\}$ is regular.
187. $\{w \in \{0, 1\}^* \mid w \text{ represents a integer divisible by } 374 \text{ in base } 473\}$ is regular.
188. $\{w \in \{0, 1\}^* \mid |\#(0, w) - \#(1, w)| < 374\}$ is regular.
189. $\{w \in \{0, 1\}^* \mid |\#(0, x) - \#(1, x)| < 374 \text{ for every prefix } x \text{ of } w\}$ is regular.
190. $\{w \in \{0, 1\}^* \mid |\#(0, x) - \#(1, x)| < 374 \text{ for every substring } x \text{ of } w\}$ is regular.
191. $\{w0^{#(0,w)} \mid w \in \{0, 1\}^*\}$ is regular.
192. $\{w0^{#(0,w) \mod 374} \mid w \in \{0, 1\}^*\}$ is regular.
Automata Transformations

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

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

195. 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.

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

197. 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.

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

Language Transformations

199. For every regular language $L$, the language $\{w^R \mid w \in L\}$ is also regular.

200. For every language $L$, if the language $\{w^R \mid w \in L\}$ is regular, then $L$ is also regular. \(\langle F14 \rangle\)

201. For every language $L$, if the language $\{w^R \mid w \in L\}$ is not regular, then $L$ is also not regular. \(\langle F14 \rangle\)

202. For every regular language $L$, the language $\{w \mid ww^R \in L\}$ is also regular. \(\langle hw \rangle\)

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

204. For every language $L$, if the language $\{w \mid ww^R \in L\}$ is regular, then $L$ is also regular. \(\text{Hint: Consider the language } L = \{0^n1^n \mid n \geq 0\}\).

205. For every regular language $L$, the language $\{0^{\mid w \mid} \mid w \in L\}$ is also regular.

206. For every language $L$, if the language $\{0^{\mid w \mid} \mid w \in L\}$ is regular, then $L$ is also regular.