COMP 310 Problems
I’ve thrown together many problems from previous sections of COMP310. I may add a small number of additional
problems later. I’m certain there are typos and other minor errors. Please mention any you notice in the errors and
omissions page.

1. Let \( X = \{a, b, c, d\} \), \( Y = \{b, d, e, f\} \), and \( Z = \{a, c, e\} \). Find each of the following sets.
   - \((X \cup Y) \cap Z = \)
   - \((X \cap Y) \cup Z = \)
   - \((X - Y) - Z = \)
   - \(X - (Y - Z) = \)
   - \(|X \cup Y \cup Z| = \)
   - \(2^Z = \)
   - \(X \times Y = \)
   - \(Y \triangle Z = \)

2. Let \( X = \{1, 2, 3, 4\} \), \( Y = \{2, 4, 6, 7\} \), and \( Z = \{1, 3, 4\} \). Find each of the following sets.
   - \((X \cup Y) \cap Z \)
   - \((X \cap Y) \cup Z \)
   - \((X - Y) - Z \)
   - \(X - (Y - Z) \)
   - \(|X \cup Y \cup Z| \)
   - \(X \times Y \)
   - \(X \triangle Z \)
   - \(2^Z \)

3. Let \( A = \{3 \cdot n : n \in \mathbb{Z}^+\} \) and \( B = \{9^m : m \in \mathbb{Z}^+\} \). Prove \( B \subset A \).

4. Show that \( A \triangle (B \triangle C) = (A \triangle B) \triangle C \).

5. Recall that a tree is a connected acyclic graph. Define tree1 as a graph which has exactly 1 simple path
   between any two nodes. Define tree2 as a graph that is connected and has one less edge than vertex. Prove
   that the concepts of tree, tree1, and tree2 are equivalent.

6. Pigeonhole Principle: If \( p \) pigeons fly into \( h \) holes then at least one hole must contain at least \( \lceil p/h \rceil \) pigeons.
   Prove the pigeonhole principle (hint use contradiction).

7. Assume that at the end of the semester there will be 15 students receiving grades for this class. Prove that at
   least 2 students will get exactly the same letter grade. Prove that it is possible that no set of 3 students will
   get the same grade.

8. Assume that you have a directed graph with \( n \) vertices. Show that any path with \( n \) or more edges cannot be
   a simple path.

9. Assume that \( G \) is a connected, undirected graph with 23 nodes and that no node has degree higher than 3.
   Give a convincing argument that for any vertex \( v \) there is another vertex \( w \) such that the shortest path from
   \( v \) to \( w \) has at least 4 edges.

10. For each of the lists below either draw an connected, undirected graph with eight nodes having one node of
    each degree listed or give a convincing argument why it is impossible.

    - 7, 7, 3, 3, 3, 3, 1
    - 7, 4, 3, 3, 3, 3, 2
    - 6, 5, 4, 3, 3, 3, 2
11. Prove that every directed graph without a directed cycle has at most \( \frac{n(n-1)}{2} \) edges where \( n \) is the number of vertices. Show that it is possible to have \( \frac{n(n-1)}{2} \) edges.

12. The RST corporation has hundreds of programmers using the techniques of eXtreme Programming. One tenent of XP is pair programming (Code is written by pairs: one types while the other watches, makes suggestions, and catches errors. After a while the two people switch roles.) From time to time the pairs are broken up and new pairs are made. Manager Bob has noticed that in every group of 6 programmers there are always 3 programmers for which no pair has ever worked together or 3 programmers for which every possible pair has worked together. Bob wonders what is so special about his company. Convince Bob that there is nothing special about this property.

13. Prove that \( \sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6} \).

14. Prove that \( \sum_{i=1}^{n} i = \frac{n(n+1)}{2} \).

15. Prove that \( \sum_{i=1}^{n} \frac{1}{i} = \ln(n) + \gamma + \frac{1}{n} + \Theta(1) \).

16. Construct a DFA which accepts \( L = \{a^ib^j : i \cdot j \mod 2 = 1\} \).

17. What is \( L(M_1) \)? Explain.

18. Construct a DFA for \( L = \{w \in \{0, 1\}^* : \text{every 0 is followed by a 1}\} \).

19. Construct a DFA which accepts \( L = \{a^ib^j : i + j < 5\} \).

20. Construct a DFA which accepts \( L = \{a^ib^j : i + j < 5\} \).

21. Give a DFA that accepts each of the following languages.
   (a) \( L_1 = \{w \in \{0, 1\}^* : w \text{ contains at least three 0's}\} \)
   (b) \( L_2 = \{w \in \{a, b, c\}^* : w \text{ does not have two consecutive a's}\} \)
   (c) \( L_3 = \{w \in \{a, b\}^* : w \text{ starts with an a and contains at least 3 b's}\} \)
   (d) \( L_4 = \{w \in \{a, b\}^* : w \text{ starts and ends with the same letter}\} \)

22. Construct a DFA which accepts \( L = \{w \in \{a, b\}^* : \text{every odd position must be a “b”}\} \).

23. Construct (and explain your construction) a DFA which accepts \( L = \{w \in \{a, b\}^* : \text{the length of w is odd}\} \).

24. Construct a DFA that accepts \( L = \{a^ib^j : i > 1, \ j \text{ even, } & k \text{ odd}\} \). Briefly explain your construction.

25. What language is accepted by the DFA below? Briefly explain your answer.

26. Construct a DFA that accepts \( L = \{a^ib^j : i \cdot j \text{ is even}\} \).
27. What is $L(M_5)$? Explain.

28. Construct an NFA which accepts $L = \{a^i b^j : i \text{ is even or } j \text{ is odd}\}$.

29. Construct an NFA which accepts $L = \{w \in \{a, b\}^* : \#a \text{ is even or } \#b \text{ is even}\}$.

30. Construct an NFA which accepts $L = \{w \in \{a, b, c\}^* : w \text{ has } aaaa \text{ and } bbbb \text{ substrings}\}$.

31. Construct an NFA which accepts $L = \{w \in \{a, b, c\}^* : w \text{ has } aaaa \text{ xor } bbbb \text{ as a substring}\}$ (xor is “exclusive or” i.e. one or the other, but not both).

32. Construct an NFA that accepts $L = \{w \in \{a, b\}^* : w \text{ contains the string } abab \text{ and ends with the string } baaa\}$. Briefly explain your construction.

33. What language is accepted by the NFA below? Briefly explain your answer.

34. Convert $M_5$ to an equivalent DFA.

35. Convert $M_9$ to an equivalent DFA.

36. Convert the NFA below to an equivalent DFA.
37. For the NFA below construct a DFA which accepts the same language.

38. Convert the NFA below into an equivalent DFA. In other words, construct a DFA which accepts the same language as the NFA.

39. Minimize the DFA $M_{10}$. 
40. Minimize the DFA $M_{11}$.

![DFA Diagram](image)

41. Minimize the DFA below.

![DFA Diagram](image)

42. Give a regular expression which describes all strings of $a$s and $b$s in which all $a$s come in runs of at least 3 (eg aaabbaaabb is in the language, but aabaaabb is not). Briefly explain your answer.

43. Give a regular expression which describes the language $L_5 = \{w \in \{a, b\}^* : w \text{ doesn’t contain } aba\}$. Briefly explain your expression.

44. Give a regular expression for each of the following languages.

- $\{a^{2i}b^{3j}\}$.
- $\{w \in \{a, b\}^* : \text{where } |w| \text{ is even}\}$.
- $\{w \in \{a, b\}^* : \text{where } w \text{ starts and ends with } a\}$.
- $\{w \in \{a, b\}^* : \text{where every } a \text{ is followed by a } b\}$.
- $\{a^ib^j : i + j \text{ is even}\}$.
- $\{w \in \{a, b\}^* : w \text{ ends with } 3 \text{ identical letters}\}$

45. Give a regular expression associated with the following languages.

(a) $L_1 = \{w \in \{0, 1\}^* : \text{w contains at least two 0’s}\}$
(b) $L_2 = \{w \in \{a, b, c\}^* : \text{w does not have two consecutive a’s } \}$
(c) $L_3 = \{w \in \{a, b\}^* : \text{w has an even number of both a’s and b’s} \}$

46. For each of the following regular expressions: list 5 words in the language associated with the regular expression, list 5 words not in the language associated with the regular expression, and describe the language in English (Mathematical terminology/symbolism is also acceptable).

(a) $(a + b)^*abbbab(a + b)^*$
(b) \((a + b)(a + b)(a + b)a(a + b)\)
(c) \((00 + 11)^*\)
(d) \(2^*1^*0^*\)
(e) \(0^*(01^*0 + 10^*1)01^*\)

47. Give an English description of the language described by the regular expression \(r = (a + b)(a + b)^*(aaaaa + bbbba)(a + b)^*(a + b)\).

48. Let \(L_1\) be the language described by the regular expression \((ab + b)^*(bba + ab)a^* + (ab)^*\). Give an NFA which accepts \(L_1\).

49. Give an NFA which accepts the language described by the regular expression \((a + aba)(a^*b + bbb)^*\).

50. Give an NFA which accepts the language described by the regular expression \(ab + ba^* + b^*a^*\).

51. Give an NFA that accepts the language described by the regular expression \(r = (ab + ba)^*(ab^*a + ba^*b)^*(abb + baa)\).

52. Let \(L_2\) be the language accepted by the DFA below. Give a regular expression for \(L_2\) (to ease grading remove the states in the order S, A, B, C, D, E).

53. Give a regular expression for the language accepted by the DFA below.

54. Give a regular expression for the language accepted by the DFA below.

55. Give a regular expression that describes the same language as accepted by the DFA below (remove the states in the order S, A, B, C).
56. Give a regular expression describing the language accepted by the DFA below.

57. Convert the DFA below to a right linear grammar. In other words, construct a RLG which generates the language accepted by the DFA.

58. Convert the following DFA to a regular expression (remove the states in the order S,A,B,C). In other words, give a regular expression that describes the language accepted by the DFA.
59. Give a right-linear grammar for \( L_2 \) (the language from the previous problem).

60. Let \( L_4 \) be the language generated by the right-linear grammar below. Give an NFA which accepts \( L_4 \).

\[
\begin{align*}
S & \rightarrow aS | bA | aB \\
A & \rightarrow aS | aB | bC | a \\
B & \rightarrow bB | aB | bA | b \\
C & \rightarrow aS | bA | aC | a \\
\end{align*}
\]

61. Give a NFA which accepts the same language generated by the right linear grammar below.

\[
\begin{align*}
S & \rightarrow abaS | baA | ab \\
A & \rightarrow bB | bbB | aC | b \\
B & \rightarrow aS | babA \\
C & \rightarrow bA | abaS | abbbB \\
\end{align*}
\]

62. Give a right linear grammar that generates the same language as the DFA below.

63. Let \( L_5 \) be the language accepted by the DFA below. Give a regular expression for \( L_5 \) (to ease grading remove the states in the order S, A, B, C, D, E).
64. Let $L_6 = \{ww : w \in \{a, b\}^*\}$. Prove that $L_6$ is not regular.

65. Let $L_7 = \{a^ib^j : |i - j| < 100\}$. Determine whether or not $L_7$ is regular. Prove that you are correct.

66. Let $L_8 = \{w \in \{a, b\}^* : \#a \cdot \#b \text{ is even}\}$. Determine whether or not $L_8$ is regular. Prove that you are correct.

67. Show that $L_{10} = \{a_i^4\}$ is not regular.

68. Show that $L_{11} = \{w \in \{a, b\}^* : w \text{ is a palindrome}\}$ is not regular.

69. Prove that $L_1 = \{w \in \{a, b\}^* : \#a - \#b < 10\}$ is not regular.

70. Prove that $L_3 = \{a^ib^j : i + j = k\}$ is not regular.

71. Show that $L_4 = \{w \in \{a, b, c\}^* : \#a = \#b \cdot \#c\}$ is not regular.

72. Show that $L_{10} = \{a^2\}$ is not regular.

73. Let $L_9$ be a regular language and $\text{Suffix}(L_9)$ be the set of all suffixes of all words in $L_9$. Must $L_9$ also be regular? Prove that $\text{Suffix}(L_9)$ is regular or show a counterexample.

74. Let $L_{10}$ be a regular language and $L' \subseteq L_{10}$. Must $L'$ also be regular? Prove that $L'$ is regular or show a counterexample.

75. Show that regular languages are closed under intersection.

76. If $L_5$ and $L'_5$ are regular prove that $L_5 - L'_5$ must be regular.

77. If $L_6$ is regular prove that $\text{Prefix}(L_7)$ is also regular (where $\text{Prefix}(L)$ is the set of all prefixes of all languages in $L$).

78. Assume $L_7$ is regular and $L'_7$ is not regular. Can $L_7 \cap L'_7$ be regular? Can $L_7 \cap L'_7$ be non-regular? For each part either provide a proof that it cannot occur or give an example to show that it is possible.

79. What language is generated by the CFG below? Explain your answer.

$$
S \rightarrow aSd \mid T \\
T \rightarrow aTc \mid U \\
U \rightarrow bUc \mid \lambda
$$

80. Give a context free grammar for $L_7 = \{w \in \{a, b\}^* : w = w^R\}$ (palindromes).

81. Create a context free grammar which generates the language

$$L_2 = \{w \in \{a, b\}^* : \#a = \#b \text{ or } \#a = 2\#b\}.$$

82. Give a context free grammar for $L_8 = \{a^ib^j : i + j = k + \ell\}$.

83. Give a context free grammar for $L_9 = \{w \in \{a, b, c\}^* : \#a + \#b = \#c\}$.

84. Give a context free grammar for $L_{10} = \{a^ib^j : 2i = 3j + 4\}$.
85. Construct (and explain your construction) a CFG which generates
$$L_5 = \{w \in \{a, b\}^*: w \text{ does not contain 2 consecutive a's}\}.$$ 

86. Give a CFG that generates the language $$L = \{a^ib^j: i > 2, j > 3\}$$
87. Give a CFG that generates the language $$L = \{a^{3i}b^{2j}: i, j \in \mathbb{Z}\}$$
88. Give a CFG which generates $$L_1 = \{a^ib^j: i \neq j\}.$$ 
89. Give a CFG which generates $$L_2 = \{w \in \{a, b\}^*: \#a \neq \#b\}.$$ 
90. Give a CFG which generates $$L_3 = \{a^ib^j: i \neq 3j + 1\}.$$ 
91. Give a CFG which generates $$L_4 = \{a^ib^j: i \geq 2, j \geq 3\}.$$ 
92. Construct a CFG which generates $$L_4 = \{w \in \{a, b\}^*: w \text{ contains at least two a's}\}.$$ 
93. What language is generated by the following grammar? Explain.
$$S \rightarrow abS \mid T$$
$$T \rightarrow cTdd \mid \lambda$$

94. What language is generated by
$$S \rightarrow aSbT$$
$$T \rightarrow aTa\lambda$$

95. What language is generated by
$$S \rightarrow aTbU$$
$$T \rightarrow aTb|ab$$
$$U \rightarrow dU|dc$$

96. Give 5 words generated by the CFG below. What language is generated by the CFG? Explain.
$$S \rightarrow aSbS \mid bSaS \mid \lambda$$

97. Draw a parse tree and give a leftmost derivation for some word (with length at least 6) which is generated by this grammar,
$$S \rightarrow aAA \mid AbB \mid SaB$$
$$A \rightarrow AB \mid BBA \mid aB$$
$$B \rightarrow aBAB \mid aBSB \mid ab$$

98. Is the following grammar ambiguous? Prove your claim.
$$S \rightarrow aAB \mid bBA \mid ab$$
$$A \rightarrow Aab \mid BB \mid a \mid \lambda$$
$$B \rightarrow AC \mid aBAa \mid CC$$
$$C \rightarrow SAb \mid b \mid \lambda$$

99. Show that the following grammar is ambiguous.
$$S \rightarrow aB \mid bA \mid \lambda$$
$$A \rightarrow aS \mid bAA \mid a$$
$$B \rightarrow bS \mid aBB \mid b$$
100. Use the CYK algorithm to show which of the following words is generated by the grammar below.
\( w_1 = aaaaab, w_2 = aaab, w_3 = aba \)

\[
S \rightarrow AB | CD | AD \\
A \rightarrow BA | DC | CB \\
B \rightarrow SS | AC | b \\
C \rightarrow AS | CC | a \\
D \rightarrow SA | a
\]

101. Use the CYK algorithm to determine which of \( acac, bc cacac, \) and \( acacac \) are generated by the grammar below.

\[
S \rightarrow AA | AC | CC \\
A \rightarrow AC | BC | CA | a \\
B \rightarrow BB | AB | b \\
C \rightarrow CA | c
\]

102. Use the CYK algorithm to determine which of \( abba, bbab, \) and \( ab babba \) are generated by the grammar below.

\[
S \rightarrow AB|BA|SS \\
A \rightarrow CC | a \\
B \rightarrow CC | b \\
C \rightarrow AA|SS|a
\]

103. Use the CYK algorithm to determine whether \( w = abbbcb \) is generated by the grammar below?

\[
S \rightarrow CB | NN \\
U \rightarrow BB | CB \\
B \rightarrow UB | b \\
C \rightarrow UC | SS | a | c \\
N \rightarrow UU
\]

104. Remove useless productions from the following grammar.

\[
S \rightarrow aAC | BF | aFaa \\
A \rightarrow aDE | aEb | AaB \\
B \rightarrow SAB | SaA | \lambda \\
C \rightarrow AaB | SaS | a \\
D \rightarrow aA | EDa | bE \\
E \rightarrow EaE | aA | bD \\
F \rightarrow aFB | Fab | aBBb | bEC
\]

105. Remove lambda productions from the following grammar.

\[
S \rightarrow aAB | BaaB \\
A \rightarrow aAb | bB | DD \\
B \rightarrow aCa | ADD \\
C \rightarrow aDD | a \\
D \rightarrow bAS | \lambda
\]
106. Remove unit productions from the following grammar.

\[
S \rightarrow aAB \mid BaaB \mid D \\
A \rightarrow aAb \mid B \mid DD \\
B \rightarrow aCa \mid ADD \parallel D \\
C \rightarrow aDD \mid a \\
D \rightarrow bAS \mid ab
\]

107. Convert the following grammar to CNF.

\[
S \rightarrow aAC \mid aBaa \\
A \rightarrow aDE \mid aEb \mid AaB \mid b \\
B \rightarrow SAB \mid SaA \mid a \\
C \rightarrow AaB \mid SaS \mid a \\
D \rightarrow aA \mid EDa \mid b \\
E \rightarrow aSB \mid Sab \mid aBBb \mid bEC
\]

108. Give a grammar equivalent to the grammar below which contains no \(\lambda\)-productions.

\[
S \rightarrow aBb \mid aAB \mid bBA \\
A \rightarrow AA \mid baA \mid \lambda \\
B \rightarrow SBA \mid CDA \mid \lambda \\
C \rightarrow aCC \mid CAa \mid \lambda \\
D \rightarrow AA \mid a
\]

109. Give a CFG with no \(\lambda\) productions which generates the same language as the grammar below (ie remove \(\lambda\) productions from the following grammar).

\[
S \rightarrow aAB \mid aBC \mid aC \\
A \rightarrow bA \mid BBB \\
B \rightarrow aB \mid AB \mid \lambda \\
C \rightarrow a \mid b \mid ACb \mid \lambda
\]

110. Give a grammar equivalent to the grammar below which contains no useless productions.

\[
S \rightarrow aSbaS\mid FBa\mid aA \\
A \rightarrow AC\mid a \\
B \rightarrow BB\mid b \\
C \rightarrow DFA\mid a \\
D \rightarrow DDB\mid CC\mid a \\
E \rightarrow EE\mid FD \\
F \rightarrow EF\mid DDE\mid aE
\]

111. Give a grammar equivalent to the grammar below which contains no unit productions.

\[
S \rightarrow aSbaS\mid FBa\mid aA \\
A \rightarrow C\mid a \\
B \rightarrow BB\mid b
\]
112. Give a grammar equivalent to the grammar below which is in CNF.

\[ \begin{align*}
C & \rightarrow D|ABaAB|a \\
D & \rightarrow DD|CC|a \\
E & \rightarrow EE|FD|b \\
F & \rightarrow EF|DDE|aE
\end{align*} \]

113. Remove \( \lambda \) productions from the following grammar. In other words, give a grammar which generates the same language, but does not contain any \( \lambda \) productions.

\[ \begin{align*}
S & \rightarrow aSbaS|FBa|aA \\
A & \rightarrow B|AC|a \\
B & \rightarrow BB|bC|\lambda \\
C & \rightarrow DF|a \\
D & \rightarrow DD|CC|a \\
E & \rightarrow EE|FD \\
F & \rightarrow EF|DDE|aE
\end{align*} \]

114. Give a NPDA which accepts \( L_1 = \{a^i b^j : i \neq j\} \).

115. Give a NPDA which accepts \( L_2 = \{w \in \{a, b\}^* : \#a \neq \#b\} \).

116. Give a NPDA which accepts \( L_3 = \{a^i b^j : i \neq 3j + 1\} \).

117. Give a NPDA which accepts \( L_4 = \{a^i b^j c^k : i = 2j + k\} \).

118. Give a NPDA which accepts \( L_5 = \{a^i b^j c^k : i = j \text{ or } j = k\} \).

119. Construct an NPDA which accepts \( L_1 = \{a^i b^j c^k : i + j + \ell = k\} \).

120. Construct an NPDA which accepts \( L_2 = \{w \in \{a, b, c\}^* : w \text{ is a palindrome}\} \).

121. Construct an NPDA which accepts \( L_3 = \{w \in \{a, b\}^* : \#a = 3\#b\} \).

122. Create NPDA which accepts the language \( L_3 = \{a^i b^j c^k : i = j + k = 2\} \).

123. Construct a NPDA for the language of all palindromes over the alphabet \{a, b\} which contain exactly one b and briefly explain your construction. A more formal way to describe the language is

\[ L_5 = \{w \in \{a, b\}^* : w = w^R \& \#b = 1\} \]

124. Show that \( L_1 = \{a^p : p \text{ is prime}\} \) is not context free.

125. Show that \( L_2 = \{a^i b^j c^k : i \cdot k = j\} \) is not context free.

126. Show that \( L_3 = \{w \in \{a, b, c\}^* : \#a = \#b \cdot \#c\} \) is not context free.

127. Show that \( L_4 = \{a^i b^j c^k : i \cdot j = k\} \) is not context free.

128. Prove that \( L_7 = \{a^{n^2}\} \) is not context free.
129. If \( L \) is context free, must \( \text{EditOutCs}(L) \) be context free? Either prove that it must be or give an example to show that it need not be. (A word \( \omega \) is in \( \text{EditOutCs}(L) \) iff there is a word \( \omega' \in L \) which becomes \( \omega \) when you remove all of its c’s.)

130. If \( L \) is context free, must \( \text{EvenLengthWordsIn}(L) \) be context free? Either prove that it must be or give an example to show that it need not be. (If \( L = \{a^i ccc b^i ccc a^i\} \) then \( \text{EvenLengthWordsIn}(L) = \{a^i ccc b^i ccc a^i : j \text{ is even}\} \).)

131. If \( L \) is context free, must \( \text{AlphabetizeTheLetters}(L) \) be context free? Either prove that it must be or give an example to show that it need not be. (A word \( \omega \) is in \( \text{AlphabetizeTheLetters}(L) \) iff you can get \( \omega \) by alphabetizing the letters of some word in \( L \).)

132. Given a language \( L \), define \( \text{Suffix}(L) \) to be the set of all suffixes of all words in \( L \). In other words, \( w \in \text{Suffix}(L) \) iff there is a \( w' \in \Sigma^* \) such that \( w' w \in L \). Prove that if \( L \) is context free then \( \text{Suffix}(L) \) is context free.

133. If \( L \) is not context free, can \( \bar{L} \) be context free? Either give an example to show that it can be or prove that it cannot be.

134. If \( L_8 \) is context free must \( \text{Prefix}(L_8) \) also be context free? Can \( \text{Prefix}(L_8) \) be context free? Prove your assertions.

135. If \( L_9 \) is context free and \( L_9 \subseteq L \), must \( L \) be context free? Can \( L \) be context free? Prove your assertions.

136. Prove: if \( L_8 \) is context free then \( L_8^R \) is also context free (given a language \( L \), the language \( L^R \) consists of words you can get by reversing a word in \( L \)).

137. Create a TM which accepts \( \{a^i b^j : (i \cdot j) / 4! = 0\} \).

138. Create a TM which accepts \( \{a^i b^j c^k : i = j \text{ or } j = k\} \).

139. What language is accepted by the TM below? Explain your answer.

140. What language is accepted by the TM below? Explain your answer.

141. Construct a Turing Machine which when started with a binary number on the tape it finishes with a binary number which is twice as large on the tape (ie multiply the binary number by 2).
142. Construct a TM which decides \( \{ w \in \{a, b\} : w \text{ is a palindrome} \} \).

143. Give a TM which accepts
\[
L_6 = \{ w \in \{a, b, c\}^* : \#a = \#b + \#c \}.
\]

144. Construct a Turing Machine which accepts the language \( \{a^ib^ic^i : i \in \mathbb{N}\} \).

145. Construct a Turing Machine which accepts the language \( \{ w \in \{a, b, c\}^* : \#a = \#b = \#c \} \).

146. Construct a TM which accepts \( L_1 = \{a^ib^ic^i : i = 2j = 2k\} \).

147. Construct an TM which accepts \( L_2 = \{ wXw : w \in \{a, b\}^* \} \).

148. Construct a TM which subtracts 3 from a binary number. In other words, if the tape starts with a binary number then have the machine end with a binary number which is 3 smaller.

149. Construct a TM which accepts \( L_6 \) and briefly explain your construction.

\[
L_6 = \{ a^ib^j : i \geq j \}
\]

150. Assume that the input (initial string on the tape) to the TM below will always have the form \((a + b)^* X(a + b)^*\). For each of the following words state whether it would be accepted or rejected: \(abXba, abaX, aaXbb\). What language is accepted by the machine? Explain your answers.

151. Construct a TM which accepts \( L_8 \) and briefly explain your construction.

\[
L_8 = \{ w \in \{a, b\}^* : \#a = 2 \cdot \#b + 1 \}
\]

152. Assume \( L_9 \) is recursive (decidable). Give a convincing argument that \( \overline{L_9} \) is also recursive.

153. Give a convincing argument that \( \text{RUNSFOREVER} = \{(M; w) : M(w) = \not\} \) is not recursive.

154. Give a convincing argument that \( \text{EQUALSPLIT} = \{S : S \text{ can be partitioned into two sets which sum to the same value} \} \) is NP-complete. Or equivalently, give a convincing argument that if you could solve \( \text{EQUALSPLIT} \) in polynomial time then you could also solve \( \text{SUBSETSUM} \) in polynomial time.
155. Show that \( L_3 = \{ M : M \text{ halts on every possible input} \} \) is undecidable by reducing \( \text{HALTING} \) to \( L_3 \). In other words, show that if \( L_3 \) is decidable then \( \text{HALTING} \) is also decidable.

156. Solve the following instances PCP (ie for each instance either show a sequence that solves the problem or show that no such sequence exists).

\[
\begin{align*}
&\begin{pmatrix} 0 & 01 & 1 & 101 \\ 10 & 0 & 101 & 1 \end{pmatrix} \\
&\begin{pmatrix} 00 & 01 & 010 & 1 & 1 \\ 1 & 00 & 1 & 10 & 101 \end{pmatrix} \\
&\begin{pmatrix} 001 & 110 & 01 & 10 \\ 10 & 01 & 110 & 000 \end{pmatrix} \\
&\begin{pmatrix} 0 & 0 & 01 & 10 \\ 1 & 001 & 1 & 0 \end{pmatrix} \\
&\left\{ \begin{pmatrix} b \ b \ b \ b \\ b \ a \ b \end{pmatrix}, \begin{pmatrix} b \ a \ b \ b \ b \ a \ b \end{pmatrix} \right\} \\
&\left\{ \begin{pmatrix} a \ b \ a \ b \ b \ b \ a \ b \end{pmatrix}, \begin{pmatrix} b \ a \ b \ a \ b \ a \ end{pmatrix} \right\} \\
&\begin{pmatrix} 110 & 1 & 0 \\ 1 & 01 & 110 \end{pmatrix} \quad \text{(difficult)}
\end{align*}
\]

157. Solve the following instances of SAT. In other words, for each of the following boolean expressions either find a assignment of the variables which satisfies the expression or state that no satisfying assignment exists.

\[
\begin{align*}
&\left( x_1 \lor \neg x_2 \lor x_3 \right) \land \left( x_1 \lor x_2 \right) \land \left( \neg x_4 \lor x_2 \right) \land \left( x_2 \lor \neg x_3 \lor \neg x_7 \right) \land \\
&\left( x_2 \lor x_4 \lor \neg x_7 \right) \land \left( \neg x_7 \lor \neg x_4 \right) \land \left( x_2 \lor x_3 \lor x_4 \right) \\
&\left( x_4 \lor \neg x_2 \lor \neg x_7 \right) \land \left( \neg x_7 \lor x_3 \right) \land \left( \neg x_4 \lor x_1 \right) \land \left( x_2 \lor \neg x_7 \right) \land \\
&\left( x_2 \lor x_4 \lor x_3 \right) \land \left( \neg x_7 \lor \neg x_4 \right) \land \left( x_2 \lor \neg x_3 \right) \land \left( x_2 \lor x_4 \right) \\
&\left( x_1 \lor \neg x_2 \lor x_3 \right) \land \left( \neg x_1 \lor \neg x_2 \lor \neg x_7 \lor x_2 \right) \land \left( x_2 \lor \neg x_7 \lor \neg x_3 \lor x_3 \right) \\
&\left( x_1 \lor \neg x_2 \lor x_3 \right) \land \left( \neg x_1 \lor x_3 \lor \neg x_5 \lor x_5 \right) \land \left( x_2 \lor \neg x_5 \lor \neg x_3 \lor x_5 \right) \land \left( x_2 \lor x_4 \lor \neg x_5 \lor \neg x_5 \right) \land \left( x_2 \lor x_4 \lor x_5 \lor \neg x_5 \right) \land \left( x_2 \lor x_4 \lor x_5 \lor \neg x_5 \right) \\
&\left( x_1 \lor \neg x_2 \lor x_3 \right) \land \left( x_1 \lor x_3 \lor \neg x_7 \lor x_2 \right) \land \left( x_1 \lor \neg x_7 \lor x_3 \lor x_7 \right) \land \left( x_1 \lor x_2 \lor x_3 \lor \neg x_3 \lor \neg x_3 \right) \land \left( x_2 \lor x_4 \lor x_3 \lor \neg x_3 \lor x_3 \right) \land \left( x_2 \lor x_3 \lor \neg x_7 \lor \neg x_7 \right) \land \left( x_2 \lor x_3 \lor x_4 \lor \neg x_4 \right) \\
&\left( x_1 \lor x_3 \lor \neg x_7 \right) \land \left( x_1 \lor x_3 \lor \neg x_7 \lor x_3 \right) \land \left( x_1 \lor x_3 \lor x_3 \lor \neg x_7 \lor \neg x_7 \right) \\
&\left( x_1 \lor x_3 \lor \neg x_7 \right) \land \left( x_1 \lor x_3 \lor \neg x_7 \lor x_3 \right) \land \left( x_1 \lor x_3 \lor \neg x_7 \lor \neg x_7 \right) \land \left( x_1 \lor x_3 \lor x_3 \lor \neg x_7 \lor \neg x_7 \right)
\end{align*}
\]

158. Solve the following instances of \( \text{SUBSETSUM} \) (ie for each instance either show a subset that solves the instance or show that no such subset exists).

\[
\begin{align*}
&S = \{ 2, 7, 13, 19, 29, 31 \} \text{ and } T = 43. \\
&S = \{ 4, 7, 14, 21, 27, 33, 37 \} \text{ and } T = 75. \\
&S = \{ 23, 29, 39, 44, 50, 57, 63, 71, 73, 81, 99, 104 \} \text{ and } T = 361. \\
&S = \{ 24, 73, 81, 102, 113, 312, 333, 352, 414, 429 \}, \text{ } T = 861. \\
&S = \{ 4, 8, 12, 20, 28, 32, 44, 48, 58 \} \text{ and } T = 92. \\
&S = \{ 4, 8, 12, 20, 28, 32, 44, 48, 60 \} \text{ and } T = 82.
\end{align*}
\]

159. Solve the following instances of \( \text{SUBSETSUM} \).

\[
\begin{align*}
&S = \{ 13, 19, 25, 31, 47, 51, 67, 89 \}, \text{ } T = 162. \\
&S = \{ 13, 19, 25, 31, 47, 51, 67, 89 \}, \text{ } T = 161. \\
&S = \{ 13, 19, 25, 31, 47, 51, 67, 89 \}, \text{ } T = 147.
\end{align*}
\]
160. Find the maximum clique in the graph below and give a convincing argument that there is no larger clique.

161. For each of the following instances of SUBSETSUM state whether it has a solution and give a short justification for your answer.

- \( S = \{10, 21, 30, 42, 50, 61, 72, 80, 91, 102\}, T = 532 \)
- \( S = \{10, 21, 30, 42, 50, 61, 72, 80, 91, 102\}, T = 305 \)
- \( S = \{10, 21, 30, 42, 50, 61, 72, 80, 91, 102\}, T = 379 \)

162. For each of the following statements, state whether it is true or false.

- If \( A \) is a set with 4 elements and \( B \) is a set with 3 elements then \( A \times B \) is a set with \( 4^3 = 64 \) elements.
- \( \lambda \lambda b \lambda a \lambda a \lambda = baa \)
- A DFA can have an infinite number of states.
- An NFA can accept more than one language.
- A regular grammar can have at most 5 variables.
- Some infinite languages are regular.
- If a language cannot be described by a regular expression, it is not regular.
- Every regular expression must contain a *. 
- \( \{a^ib^i\} \) is not regular.
- If \( L_1 \cap L_2 = L_3 \) and both \( L_2 \) and \( L_3 \) are regular then \( L_1 \) must be regular.
- If \( L \) is a regular language then it is generated by some CFG.
- If \( L \) is generated by some CFG then it is regular.
- If \( A \) is a set with 3 elements and \( B \) is a set with 2 elements then \( A \cup B \) must contain 5 elements.
- If \( A \) is a set with 3 elements and \( B \) is a set with 2 elements then \( A \cap B \) can contain 2 elements.
- If \( A \) is a set with 3 elements then the power set of \( A \) contains 6 elements.
- All relations are functions.
- Some relations are functions.
- All functions are relations.
- All languages are regular.
- The complement of a regular language is regular.
- The complement of a CFL is CF.
- All finite languages are CF.
- There is a CFL which does not have a CFG.
- Every CFL has a DPDA which accepts it.
- The union of two CFLs is context free.
- Some context free languages are regular.
- Every language has a CFG.
- Every DPDA accepts a CFL.
- Every CFG in Chomsky normal form generates a CFL.
- \( \{wu : w \in \{a, b, c\}^* \} \) is CF.
- Every CFL has a TM which accepts it.
- Every TM accepts a CFL.
- Some CFLs have a TM which accepts it.
Some TMs accept CFLs.
All problems are decidable.
Some problems are decidable.
Every CFL has a DFA which accepts it.
Some CFLs have a DFA which accept it.
The intersection of two CFLs must be CF.
The intersection of two regular languages must be CF.
The intersection of two regular languages can be CF.
Every CFG generates a regular language.
Every regular language is generated by a CFG.
The union of a regular language and a CFL must be regular.
The union of a regular language and a CFL must be CF.
If $|A| = 4$ and $|B| = 3$ then $|A - B|$ must be 1.
If $|A| = 4$ and $|B| = 5$ then $|A \times B|$ must be 20.
All DFAs accept regular languages.
A regular language must be infinite.
$\{(ab)^{2i}\}$ is regular.
The intersection of 2 finite languages must be regular.
Some regular languages are not context free.
All grammars are context free.
A grammar in CNF can have the production $A \rightarrow aBACACABAA$
Every DFA can be converted into an NPDA.
An NPDA can pop two stack symbols in a single transition.
$\{a^i b^j c^k d^\ell : i = j \& k = \ell\}$ is context free.
Some context free grammars are ambiguous.
Every context free language is accepted by some NFA.
A parse tree must be a binary tree.
A CFG cannot generate a finite language.
It is possible for to determine whether a given word is in a context free language.
All CFGs have $\lambda$ productions.
The union of two non-context free languages can be context free.
The intersection of 2 context free languages can be regular.
SUBSETSUM can be decided by a Turing Machine.
HALTING can be decided by a Turing Machine.
$\{a^i b^j c^k d^\ell\}$ can be decided by a Turing Machine.
Every language has a Turing Machine which decides it.
NP is the set of all languages which cannot be solved by a Turing Machine in polynomial time (ie non-polynomial).
Every recursive language is context free.
$\{w \in \{a, b, c\}^* : \#a = \#b = \#c\}$ is context free.
Any NPDA can be converted into an equivalent Turing Machine.
Every Turing Machine accepts a language.
$\{w \in \{a, b\} : \#a = \#b\}$ is recursive.
HALTING is NP-Complete.
SAT is in NP.
Some recursive languages are recursively enumerable.
Every infinite language is recursively enumerable.
There is no recursively language which is also regular.
Every context free language is recursive.
$\{w \in \{a, b, c\}^* : \#a = \#b = \#c\}$ is recursive.
Any Turing Machine can be converted into an equivalent NPDA.
Deciding a language and accepting a language are the same thing.
$\{w \in \{a, b\}^* : \#a = \#b\}$ is regular.
CLIQUE is NP-Complete.
Halting is in P.
Some recursively enumerable languages are recursive.
Every recursively enumerable language is infinite.
There is no context free language which is also recursively enumerable.
Every context free language is recursive.
\{a^ib^ic^i\} is context free.
Any Turing Machine can be converted into an equivalent NPDA.
Every Turing Machine decides a language.
\{a^ib^i\} is recursive.
SAT is NP-Complete.
Halting is in P.
Every recursive language is recursively enumerable.
Every infinite language is recursive.
There is no recursively enumerable language which is also regular.
The intersection of two regular languages is recursive.
Some languages that can be described by regular expressions are not recursive.
Every regular language is in P.
Some COMP310 finals contain this T/F problem.
A context free grammar can describe a non-recursive language.
\{a^ib^ic^id^ie^mf^n\} is recursive..
Next year someone might find a TM which decides the Halting Problem.
All finite language are in P.
NP is the set of problems which are Not Polynomial on a TM.
A language can be both recursive and recursively enumerable.