# 5. (a) Use induction. Let \( P(n) \) be the proposition: "If \( L_1, \ldots, L_n \) are regular, then so is \( \bigcup_{i=1}^{n} L_i \)." Then \( P(2) \) is true by the Closure theorem. Now suppose \( P(k) \) is true.

Let \( L_1, \ldots, L_k, L_{k+1} \) be any \( k+1 \) regular languages. Then
\[
\bigcup_{i=1}^{k+1} L_i = \left( \bigcup_{i=1}^{k} L_i \right) \cup L_{k+1}
\]

is regular by ind. hyp.

So \( P(k) \Rightarrow P(k+1) \). Hence \( P(k) \) is true for all \( k \geq 2 \). So the results follow.

(b) The situation for intersection is entirely similar.

# 6. \( S_1 \cap S_2 = (S_1 - S_2) \cup (S_2 - S_1) \)

Now if \( S_1 \) & \( S_2 \) are regular, then \( S_1 \cap S_2 \) will also be regular because \( (S_1 - S_2) \) & \( (S_2 - S_1) \) will be regular by the Closure theorem and so \( (S_1 - S_2) \cup (S_2 - S_1) \) will then be regular by the Closure theorem. Hence reg. lang. are closed under symmetric differences.

# 7. **Hint:** \( \overline{L_1 \cap L_2} = \overline{L_1} \cap \overline{L_2} \).

Now use the Closure theorem.

# 12. Yes. First observe that \( L_2 = (L_1 \cup L_2) - (L_1 - L_2) \)

Now \( L_1 - L_2 \) is finite, so \( L_1 - L_2 \) is regular. And \( L_1 \cup L_2 \) is given as regular. \( \therefore (L_1 \cup L_2) - (L_1 - L_2) \) is reg. by Clos. Thm.
#12. Hence $L_2$ is regular.

$$L_2 = (L_1 L_2) - (L_1 - L_2)$$

$L_1 - L_2 \subseteq L_1$. So $L_1 - L_2$ is finite.

#13. Let $\Sigma$ be the alphabet on which $L$ is based. Then

$$L_1 = \{uv : u \in L \land |v| = 2\} = L \cdot \Sigma \cdot \Sigma$$

Now any alphabet is finite, so $\Sigma$ is regular. Hence $L \cdot \Sigma \cdot \Sigma$ is regular by the Closure Theorem. So $L_1$ is regular.

#14. $\{uv : u \in L, v \in L^R\} = L \cdot (L^R)$

Now use the closure theorem.

#15. No. Let $L_1 = \{a^n : n \geq 0\}$ and $L_2 = \{a^p : p \text{ is prime}\}$.

Then $L_1 \cdot L_2 = \{a^n \cdot a^p : n \geq 0\}$

$$= \{a^{n+p} : n \geq 0\} = a^*$$

So $L_1$ & $L_1 \cdot L_2$ are both regular. It will be shown later that $L_2$ is non-regular by using various means. (See supplementary problems also)

#26. See class notes.
#1 \( L_1 \subseteq L_2 \iff L_1 - L_2 = \emptyset \). Since \( L_1 \) \& \( L_2 \) are regular we can find dfa's \( M_1 \) \& \( M_2 \) for \( L_1 \) \& \( L_2 \). Using \( M_1 \) \& \( M_2 \) we can algorithmically get a dfa \( M \) for \( L_1 - L_2 = (L_1 \cup L_2)^c \). Now \( L_1 - L_2 = \emptyset \iff L(M) = \emptyset \iff \) there is no path from the initial state of \( M \) to an accepting state of \( M \) (and this can be checked algorithmically).

#2 Since \( L \) is regular, we can find a dfa \( M \) such that \( L(M) = L \). Now \( x \in L \iff q_0 \in F(M) \) (i.e., if the initial state of \( M \) is an accepting state also). Since this can be algorithmically checked, there is an algorithm to tell if \( x \in L \).

#5 Since \( L \) is regular, we can find a dfa \( M \) such that \( L(M) = L \). Let \( M^R \) be the machine obtained by making all accepting states in \( M \) into initial states of \( M^R \) & all initial states in \( M \) into accepting states of \( M^R \). Then it can be algorithmically checked by Thm 4.7 if \( L(M) = L(M^R) \). \( L \) is palindromic \( \iff L = L^R \iff L(M) = L(M^R) \). Oh, you also have to reverse the arrows in \( M \) to get \( M^R \).

#8. Make the machine \( M^R \) as above. Then \( L \) has a string \( w \) such that \( w^R \in L \iff w \in L(M) \& w^R \in L(M) \iff w \in L(M) \& w \in L(M^R) \iff w \in L(M) \cap L(M^R) \). So check algorithmically if \( L(M) \cap L(M^R) \neq \emptyset \).
**Section 4.3 p. 123**

**#1** Hint: The idea is to look for the last occurrence of a repetition, in an accepting sequence. Following the proof on p. 119, look at the sequence 
\[ q_0, q_i, q_j, \ldots, q_f. \]
At least one state must be repeated, and such a repetition must start no later than the \( n \)-th move from the end. From this the proof will follow.

**#3(a)** Let \( L = \{ w \in \{a, b\}^* : n_a(w) = n_b(w) \} \). Then \( L \) is infinite. Suppose \( L \) is regular.
Let \( m \) be as in the Pumping Lemma.
Choose \( u = a^m b^m \). Then \( u = xyz \) as in the lemma.
\((xy) \leq m \implies y \) consists only of \( a \)'s
\( n_a(xyyz) > m \) but \( n_b(xyyz) = m \)
\( \vdots \)
\( xyyz \in L \). But \( xynz \in L \) for all \( n \geq 0 \) by the lemma. Hence we have a contradiction. So \( L \) cannot be regular.

(b) \( L^* = L \). So \( L^* \) is also non-regular.

**#4(a)** Use the Pumping Lemma with \( u = a^m b^m a^{m+1} \).

(b) If \( L \) is regular, so is \( \bar{L} \). And if \( \bar{L} \) is regular so is \( \bar{L} \cap a^* b^* a^* = \{ a^k b^k a^k : k = n+1 \} = L_a \)
But \( L_a \) is not reg. by (a). So \( L \) is not regular.
SECTION 4.3 p. 123

#4 (c) Use the Pumping Lemma with $\mu = a^mb^na^m$

(d) Use the Pumping Lemma with $\mu = a^mb^n$

(e) If $L$ is regular, then $I$ will be regular
But $L$ is non-regular from Prob. #3. So $L$ is not regular.

(f) Using the Pumping Lemma with $\mu = a^mb^na^b$
will give you the result.

5. (a) $L = \{a^p : p \text{ is prime}\}$
primes = $\{2, 3, 5, 7, 11, \ldots\}$
Apply the Pumping Lemma with $\mu = a^p$ where $p$ is a prime $\geq m$.

Then $\mu = xyz$ as in the PL
So $\mu = a^{x+y}a^z = a^{x+y}a^y$
Now by the P.L. $xy^nz = a^{x+y+2}a^n(y)$ is in $L$

So $(x+y+2) + k|y|$ will always be prime.
Let $\alpha = |x| + |z|$ & $\beta = |y|$. Then $\alpha + k\beta$
will always be prime. But

If $\alpha = 0$ or $1$, we get a contradiction with $k = 0$
And if $\alpha \geq 2$, we get a contradiction with $k = \alpha$, because
because $\alpha + \alpha\beta = \frac{\alpha(1+\beta)}{2^2}$ which is not prime

$\therefore L$ is not regular.
# 5 (b) $L$ is not regular by 5(a). So $L$ is not regular also.

(c) Apply P.L. with $\mu = a^{m^2}$

(d) Apply P.L. with $\mu = 2^m$

# 14 False. Let $L_1 = \{a^n b^n : n \geq 0\}$ and $L_2 = \{a, b\}^* - L_1$. Then $L_1$ and $L_2$ are both non-regular but $L_1 \cup L_2 = (a \cup b)^*$ is regular.

# 17 Yes. $L = L_1 \cap (L_2^R)$. Now use Closure Theorem.

# 21 Let $L_n = \{a^n b^n\}$. Then $L_n$ is reg. for each $n \geq 0$. But $L = \bigcap_{n=0}^{\infty} L_n = \{a^i b^i : i \geq 0\}$ is not regular.

# 23 No. Let $L = \{a, b\}^* - \{a^n b^n : n \geq 0\}$. Then $L$ is a non-regular language. Let $L_i = \{a, b\}^* - \{a^i b^i\}$ for $i \geq 0$.
Then each $L_i$ is regular but $\bigcap_{n=0}^{\infty} L_i = L$ which is non-regular.

# 24 No. Let $L_1 = \{a, b\}^*$ & $L_2 = \{a^n b^n : n \geq 1\}$. Then $L_1$ is reg. & $L_1 \cup L_2 = \{a, b\}^*$ is also regular. But $L_2$ is not regular. (See #11 Sec. 4.1)
# 2 The derivation tree for $aabbbaaa$ is given on the right.

# 7 (a) $S \to A S b | A A A$, $A \to a | \lambda$

(b) $S \to A | B$, $A \to a A b | a A | \lambda$, $B \to a B b | B b | a b b$
   \{a^n b^m : n \geq m^2\} \quad \{a^n b^m : n \leq m-2\}

(c) $S \to A | B b$, $A \to a a A b | a A | a$
   $B \to D D B b | D$, $D \to a | \lambda$
   \{a^n b^m : n \geq 2m+1\} \quad \{a^n b^m : n \leq 2(m-1)+1\}

d) $S \to a S b b B | \lambda$, $B \to b | \lambda$.

# 8 (a) $S \to A | B$, $A \to A c | D$, $D \to a D b | \lambda$
   $n = m$
   $B \to a B | E$, $E \to G E c | \lambda$, $G \to b | \lambda$
   $m \leq k$

(b) $S \to A | B$, $A \to A c | D$, $D \to a D b | \lambda$
   $n = m$
   $B \to a B | E$, $E \to b E c | F | G$, $F \to b F | b$
   $m > k$
   $G \to G c | c$
   $m < k$

c) $S \to a S c | T$, $T \to b T c | \lambda$

d) $S \to a S c | T$, $T \to b T c c | \lambda$.
#13
(a) \( L^2 \) is generated by \( S \rightarrow AA, \ A \rightarrow aAb/\lambda \)
(b) \( L^k \) is generated by \( S \rightarrow \underbrace{AA \ldots A}_{k \text{ times}}, \ A \rightarrow aAb/\lambda \)
(c) \( L^* \) is generated by \( S \rightarrow AS/\lambda, \ A \rightarrow aAb/\lambda \)

#20 Any derivation of \( aab.bab.a \) would have to start with
\[ S \rightarrow aaB \rightarrow aaAa \rightarrow aabBba \rightarrow aabAaba \]
and this can never lead to \( aab.bab.ba \)

#22 \( S \rightarrow [S] | (S) | SS | \lambda \)

#23 \( S \rightarrow \lambda | \emptyset | a | b | (S+S) | (S.S) | (S^*) \)

#24 Let \( V = \{X, Y\} \) and \( T = \{A, B, C, a, b, \rightarrow\} \)
The productions are given below:
\[ S \rightarrow X \rightarrow Y, \ X \rightarrow A/B/C, \ Y \rightarrow A/B/C/a/b/YY \]

"\( \rightarrow \)" denotes arrows from CFG's
"\( \rightarrow \)" denotes arrows from our grammar.

#25 Hint: Just keep applying the appropriate productions to get the rightmost needed letter to get the rightmost derivation. Same thing for the leftmost derivation except you look for leftmost needed letter.
Section 5.2 p. 145

1. \( S \rightarrow aA/b, \quad A \rightarrow aB, \quad B \rightarrow aB/b \)

2. \( S \rightarrow aA, \quad A \rightarrow aAB/b, \quad B \rightarrow b \)

#4 Hint: If \( G \) is an S-grammar, then each string \( q \) in \( L(G) \) will have a unique leftmost derivation.

#5 \( A \rightarrow ax, \quad x \in V^* \)

\( |V| \) chooses \( |T| \) choices

Maximum size of \( |P| = |V| \cdot |T| \)

#6 The string \( aab \) has two left-most derivations. So the grammar is ambiguous.

1. \( S \Rightarrow aAB \Rightarrow aab \)
2. \( S \Rightarrow AB \Rightarrow AaB \Rightarrow aab \Rightarrow aab \)

These are the two different derivation trees.
9. If $L$ is a regular language then we can find a DFA which accepts $L$. Now if we convert this DFA into a RLG there will never be a choice of productions because the DFA was deterministic. So we will get an unambiguous RLG for $L$. This means $L$ is not inherently ambiguous.

11. Yes. Let $G$ be the grammar with productions $S \rightarrow aA$, $S \rightarrow ab$, $A \rightarrow b$.
Then $ab$ has two leftmost derivations in $G$. So $G$ is ambiguous.

$G: S \rightarrow aSbS|bSaS|\lambda$

12.(a) Consider the string $w = a6ab6$

We have two derivation trees. So grammar is ambiguous.

(b) Consider $w = ab$.

$G: S \rightarrow aSbS|SS|\lambda$... $G$ is ambiguous