1. (a) Tree:

(b) \[ A_1 \rightarrow A_1A_3 \rightarrow A_1A_3A_1A_2 \rightarrow A_1A_3A_1A_3A_1A_3 \rightarrow A_1A_3A_1A_3A_1A_3b \rightarrow A_1A_3A_1A_3ab \rightarrow A_1A_3A_1bab \rightarrow A_1bab \rightarrow abab \]

(c) \[ G \] with no left recursive productions:

\[
\begin{align*}
A_1 & \rightarrow aB \mid a \\
B & \rightarrow A_3B \mid A_3 \\
A_2 & \rightarrow A_3A_1A_3 \\
A_3 & \rightarrow bC \mid b \\
C & \rightarrow A_1A_2C \mid A_1A_2 \\
\end{align*}
\]

(d) Chomsky Normal Form version of \( G \):

\[
\begin{align*}
A_1 & \rightarrow A_1A_3 \mid a \\
A_2 & \rightarrow A_3A_4 \\
A_3 & \rightarrow A_3A_5 \mid b \\
A_4 & \rightarrow A_1A_3 \\
A_5 & \rightarrow A_1A_2 \\
\end{align*}
\]

(e) Greibach Normal Form version of \( G \):

\[
\begin{align*}
A_1 & \rightarrow aB \mid a \\
B & \rightarrow bCB \mid bB \mid bC \mid b \\
A_2 & \rightarrow bCA_1A_3 \mid bA_1A_3 \\
A_3 & \rightarrow bC \mid b \\
C & \rightarrow aBA_2C \mid aA_2C \mid aBA_2 \mid aA_2 \\
\end{align*}
\]
2. For \( L = \{ a^n b^m c^{n+m} \mid n, m \geq 0 \} \) with \( \Sigma = \{a, b, c\} \), we can define a PDA \( M = \{Q, \Sigma, \Gamma, \delta, q_0, z, F\} \) where \( Q = \{q_0, q_1, q_2, q_3\} \), \( \Gamma = \{0, z\} \), \( F = \{q_3\} \) and \( \delta \) is

\[
\begin{align*}
\delta(q_0, a, z) &= \{(q_0, 0z)\} \\
\delta(q_0, a, 0) &= \{(q_0, 00)\} \\
\delta(q_0, \lambda, z) &= \{(q_1, z)\} \\
\delta(q_0, \lambda, 0) &= \{(q_1, 0)\} \\
\delta(q_1, b, z) &= \{(q_1, 0z)\} \\
\delta(q_1, b, 0) &= \{(q_1, 00)\} \\
\delta(q_1, \lambda, z) &= \{(q_2, z)\} \\
\delta(q_1, \lambda, 0) &= \{(q_2, 0)\} \\
\delta(q_2, c, 0) &= \{(q_2, \lambda)\} \\
\delta(q_2, \lambda, z) &= \{(q_3, z)\}
\end{align*}
\]

3. For \( L = \{ a^m b^2 \mid m \geq 0 \} \) with \( \Sigma = \{a, b\} \), we can define a Deterministic PDA to show it’s a deterministic CFL. The Det.PDA \( M = \{Q, \Sigma, \Gamma, \delta, q_0, z, F\} \) where \( Q = \{q_0, q_1, q_2\} \), \( \Gamma = \{1, z\} \), \( F = \{q_0\} \) and \( \delta \) is

\[
\begin{align*}
\delta(q_0, a, z) &= \{(q_1, 11z)\} \\
\delta(q_1, a, 1) &= \{(q_1, 111)\} \\
\delta(q_1, b, 1) &= \{(q_2, \lambda)\} \\
\delta(q_2, b, 1) &= \{(q_2, \lambda)\} \\
\delta(q_2, \lambda, z) &= \{(q_0, \lambda)\}
\end{align*}
\]

4. To find whether \( L_1 \) and \( L_2 \) share a common element, we need to find \( L_1 \cap L_2 \) is equal to the empty set or \( L_1 \cap L_2 = \emptyset \). We know that intersection of a regular language and context-free language is context free from closure properties. We will construct a pushdown automata (PDA) \( M_1 \) for \( L_1 \cap L_2 \) using \( M_1 \), a PDA for \( L_1 \) and \( M_2 \), a deterministic finite automata (DFA) for \( L_2 \).

We can create \( M_3 \) for \( L_1 \cap L_2 \) by combining \( M_1 = \{Q_1, \Sigma, \Gamma, \delta_1, q_0, z, F_1\} \) and \( M_2 = \{Q_2, \Sigma, \delta_2, l_0, F_2\} \). To have a machine that accepts the words both of the machines accept, we need to create new transitions, states, and final states that simulate these two machines at the same time.

Let’s say \( M_1 \) has a transition defined as \( \delta_1(q_i, a, x) = \{(q_j, y)\} \) and \( M_2 \) has a transition defined as \( \delta_2(l_k, a) = l_i \). If we receive \( a \) as an input for \( M_1 \) at state \( q_i \) and stack element \( x \), we transition to \( q_j \) with replacing \( x \) with \( y \) on the stack. If we receive \( a \) as an input for \( M_2 \) at state \( l_k \), we transition to \( l_i \). Because they take the same inputs, we can combine the states by cartesian product and do the transition on combined states.
Therefore, we can combine $q_i$ and $l_k$ as $(q_i, l_k)$ and define the transition for $M_3$ as $\delta_3((q_i, l_k), a, x) = \{(q_j, l_i), y)\}$. New final states are going to be cartesian product of final states of both machines because if a word is in both languages, it needs to be accepted by $M_3$.

For $M_3 = \{Q_3, \Sigma, \Gamma, \delta_3, k_0, z, F_3\}$, where $Q_3 = Q_1 \times Q_2$, $F_3 = F_1 \times F_2$, $k_0 = (q_0, l_0)$ and $\delta_3((s_1, s_2), a, x) = ((s_3, s_4), y)$ if $\delta_1(s_1, a, x) = (s_3, y)$ and $\delta_2(s_2, a) = s_4$ for all $s_i, a, x, y$.

After we obtain the $M_3$ we need to prove that the language is empty or not. According to Theorem 8.2 in the book, we can create a corresponding context-free grammar for $L(M_3)$. In this grammar, after we remove the useless symbols, if $S$ is removed, then this language is empty and $L_1$ and $L_2$ doesn’t share any common elements. If there’s at least one derivation rule, then it’s non-empty and $L_1$ and $L_2$ share common elements.

5. (a) For $L(G)$ with $\Sigma = \{a, b, c\}$, we can define a PDA $M = \{Q, \Sigma, \Gamma, \delta, q_0, z, F\}$ where $Q = \{q_0, q_1, q_f\}$, $\Gamma = \{X, Y, Z, z\}$, $F = \{q_f\}$ and $\delta$ is

\[
\begin{align*}
\delta(q_0, b, z) &= ((q_1, Xz) \\
\delta(q_1, b, X) &= ((q_1, XYZ), (q_1, \lambda)) \\
\delta(q_1, a, X) &= ((q_1, Y)) \\
\delta(q_1, a, Y) &= ((q_1, \lambda)) \\
\delta(q_1, c, Z) &= ((q_1, \lambda)) \\
\delta(q_1, \lambda, z) &= ((q_f, z))
\end{align*}
\]

(b) $(q_0, bbac, z) \vdash (q_1, bbac, Xz) \vdash (q_1, bac, XYZ) \vdash (q_1, ac, YZz) \vdash (q_1, c, Zz) \vdash (q_1, \lambda, z) \vdash (q_f, \lambda, z)$.

(c) $S \rightarrow bX \rightarrow bbXYZ \rightarrow bbYZ \rightarrow bbbaZ \rightarrow bbac$

(d) Tree:

```
   S
  /\  \\
 X /  \\
/  |  \\
\ b a c
```

6. $L_1 = \{a^nb^n \mid n \geq 1\}$, $L_2 = \{w \mid w = w^R\}$
7. (a) For any grammar $G$ in Chomsky Normal Form where $\lambda \notin L(G)$ and for any word $w \in L(G)$ with length $|w| = n$, the derivation will take $2n - 1$ steps. We start with a start variable in our derivation. For each step in the derivation, we will increase number of variables by 1 because of form $A \rightarrow BC$ and stop adding new variables when we have a derivation with $n$ variables because of generating $w$. This part will take $n - 1$ steps because in each step we add one more variable, so we need $n - 1$ steps to get $n$ variables.

After we get $n$ variables, it will take one step to convert each variable to a letter with rules of the form $A \rightarrow a$, therefore this part will take $n$ steps for $n$ variables because our word is $n$ letters long.

The total number of steps are $2n - 1 = 2|w| - 1$.

(b) For any grammar $G$ in Greibach Normal Form where $\lambda \notin L(G)$ and for any word $w \in L(G)$ with length $|w| = n$, the derivation will take $n$ steps. We start the derivation with a start variable. With each derivation rule in the form of $A \rightarrow aV_1V_2...$, we will add one terminal and at least one variable to the current derivation. We will repeat this step until total number of terminals and variables will be $n$.

Let’s say the initial part took $k$ steps. Therefore there needs to be $k$ terminals and $n - k$ variables that need to be transformed to terminals with rules of the form $A \rightarrow a$. This part will take $n - k$ steps. Therefore total number of steps will be $n - k + k = n = |w|$.

8. For the word $aabbbb$ the instantaneous description for $M$ in Problem 3 can be represented as: $(q_0, aabbbb, z) \vdash (q_1, abbbb, 11z) \vdash (q_1, bbbbb, 1111z) \vdash (q_2, bbb, 111z) \vdash (q_2, bb, 11z) \vdash (q_2, b, 1z) \vdash (q_2, \lambda, z) \vdash (q_0, \lambda, \lambda)$. 