1. (A) For CFG 1:
   (a) $aabb$ is in the language. The derivation tree is shown below.
   (b) $abaa$ is not in the language.
   (c) $abba$ is not in the language.
   (d) $aaaa$ is not in the language.

(B) For CFG 2:
   (a) $aabb$ is not in the language.
   (b) $abaa$ is in the language. The derivation tree is shown below.
   (c) $abba$ is in the language. The derivation tree is shown below.
   (d) $aaaa$ is in the language. The derivation tree is shown below.

(C) For CFG 3:
   (a) $aabb$ is not in the language.
   (b) $abaa$ is not in the language.
   (c) $abba$ is not in the language.
   (d) $aaaa$ is in the language. The derivation tree is shown below.

(D) For CFG 4:
   (a) $aabb$ is not in the language.
   (b) $abaa$ is not in the language.
   (c) $abba$ is not in the language.
   (d) $aaaa$ is not in the language.
Derivation tree for 1.(a):

```
S
  \--- a
    \--- S
      \--- b
```

Derivation tree for 2.(b):

```
S
  \--- a
    \--- S
      \--- b
        \--- S
          \--- a
```

Derivation tree for 2.(c):

```
S
  \--- a
    \--- S
      \--- b
        \--- S
          \--- a
```

Derivation tree for 2.(d):

```
S
  \--- a
    \--- S
      \--- a
        \--- S
          \--- a
```

Derivation tree for 3.(d):

```
S
  \--- a
    \--- S
      \--- a
        \--- S
          \--- a
            \--- X
              \--- a
```

2
2. (a) We need to put new productions where we replace instances of A, C and D with $\lambda$ symbols for every rule.

\[
S \rightarrow AbB \mid bB \mid B \\
A \rightarrow C \mid a \\
B \rightarrow S \mid b \\
C \rightarrow BbS
\]

(b) We need to replace rules $S \rightarrow B$, $A \rightarrow C$, $B \rightarrow S$ for unit production removal.

\[
S \rightarrow AbB \mid bB \mid b \\
A \rightarrow BbS \mid a \\
B \rightarrow AbB \mid bB \mid b \\
C \rightarrow BbS
\]

(c) C is unreachable, so we remove it. All rules terminate because all production rules can reach to a state where they terminate.

\[
S \rightarrow AbB \mid bB \mid b \\
A \rightarrow BbS \mid a \\
B \rightarrow AbB \mid bB \mid b
\]

3. Result after removing all useless production rules.

\[
S \rightarrow bC \\
C \rightarrow bc
\]

4. We can start by converting the language described to a context-free grammar.

\[
Q_1 \rightarrow aQ_2 \mid bQ_1 \\
Q_2 \rightarrow bQ_2 \mid Q_3 \\
Q_3 \rightarrow aQ_1 \mid Q_4 \\
Q_4 \rightarrow aQ_2 \mid bQ_4 \mid \lambda
\]

We can obtain the version without unit, $\lambda$ or useless productions by applying the same methods as we did above.

\[
Q_1 \rightarrow aQ_2 \mid bQ_1 \mid a \\
Q_2 \rightarrow bQ_2 \mid aQ_1 \mid aQ_2 \mid bQ_4 \mid a \mid b \\
Q_3 \rightarrow aQ_1 \mid aQ_2 \mid bQ_4 \mid a \mid b \\
Q_4 \rightarrow aQ_2 \mid bQ_4 \mid a \mid b
\]
We can add rules $A \rightarrow a$ and $B \rightarrow b$ to create rules of the form $A \rightarrow a$ and replace where they are not a single letter to obtain a grammar with Chomsky Normal Form.

\[
\begin{align*}
Q_1 & \rightarrow AQ_2 \mid BQ_1 \mid a \\
Q_2 & \rightarrow BQ_2 \mid AQ_1 \mid AQ_2 \mid BQ_4 \mid a \mid b \\
Q_3 & \rightarrow AQ_1 \mid AQ_2 \mid BQ_4 \mid a \mid b \\
Q_4 & \rightarrow AQ_2 \mid BQ_4 \mid a \mid b \\
A & \rightarrow a \\
B & \rightarrow b
\end{align*}
\]
We can start with a CFG of the language described that doesn’t have unit, \( \lambda \) or useless productions and convert it to Chomsky Normal Form.

\[
S \rightarrow aSb \mid aXb \mid ab \\
X \rightarrow aXa \mid bXb \mid aa \mid bb
\]

We can add rules \( A \rightarrow a \) and \( B \rightarrow b \) to create rules of the form \( A \rightarrow a \) and replace where they are not a single letter to get our intermediate grammar.

\[
S \rightarrow ASB \mid AXB \mid AB \\
X \rightarrow AXA \mid BXB \mid AA \mid BB \\
A \rightarrow a \\
B \rightarrow b
\]

Then we need to replace productions where it maps to three or more rules to create rules of the form \( A \rightarrow BC \).

\[
S \rightarrow AC_1 \mid AC_2 \mid AB \\
X \rightarrow AC_3 \mid BC_2 \mid AA \mid BB \\
C_1 \rightarrow SB \\
C_2 \rightarrow XB \\
C_3 \rightarrowXA \\
A \rightarrow a \\
B \rightarrow b
\]
6. We use the notation in the book, there will be notes to remember which transitions we took for part (b). We are trying to find whether string \( \text{bababa} \) is in the grammar \( G \) where 
\[
S \rightarrow XZ \mid ZY \\
X \rightarrow ZX \mid a \\
Y \rightarrow XZ \mid a \\
Z \rightarrow YY \mid b
\]

(a) First we need to do the direct derivations for each letter. For example \( V_{11} \) has \( Z \) in it because \( Z \rightarrow b \) is a production rule.

\[
V_{11} = \{Z\}, \quad V_{22} = \{X, Y\}, \quad V_{33} = \{Z\}, \\
V_{44} = \{X, Y\}, \quad V_{55} = \{Z\}, \quad V_{66} = \{X, Y\}
\]

Then we need to find derivations of form \( A \rightarrow BC \) for corresponding rules of first two letters.

\[
V_{12} = \{X, S\}, V_{23} = \{Y, S\}, V_{34} = \{X, S\}, V_{45} = \{Y, S\}, V_{56} = \{X, S\}
\]

For finding rules that cover first three letters, we need to rules that represent three letters by combining rules that represent one letter and two letter or vice versa. We are going to denote where the previous rules come from with subscript. For example; \( X \in V_{ij} \) will be shortened to \( X_{ij} \) instead.

\[
V_{13} = \{Y, S \mid Y \rightarrow X_{12}Z_{33}, S \rightarrow Z_{11}Y_{23}\}, \quad V_{24} = \{Z \mid Z \rightarrow Y_{23}Y_{44}\}, \\
V_{35} = \{Y, S \mid Y \rightarrow X_{34}Z_{55}, S \rightarrow Z_{33}Y_{45}\}, \quad V_{46} = \{Z \mid Z \rightarrow Y_{45}Y_{66}\}
\]

\[
V_{14} = \{Z \mid Z \rightarrow Y_{13}Y_{44}\}, V_{25} = \{Z \mid Z \rightarrow Y_{24}Y_{55}\}, \\
V_{36} = \{Z \mid Z \rightarrow Y_{35}Y_{66}\} \\
V_{15} = \{Z \mid Z \rightarrow Y_{13}Y_{45}\}, \\
V_{26} = \{X, Y, S \mid Y, S \rightarrow X_{22}Z_{36}, X \rightarrow Z_{25}X_{66}, \ldots\}, \\
V_{16} = \{X, Y, S \mid S \rightarrow Z_{11}Y_{26}, \ldots\}
\]

Because \( S \in V_{16} \), \( w = \text{bababa} \in L(G) \).
(b) When creating the derivation tree, we are going to use the rules we noted and start from $S$ production rule and move to other letters. When we see something like $S \rightarrow Z_{11}Y_{26}$, we are going to branch the tree and look to $V_{11}$ and $V_{26}$ for next rules.

We omitted any rule explanation for first two letter combinations but for the sets $V_{ii}$, it will lead to $i$'th letter. For sets $V_{i,i+1}$, it will lead to sets $V_{ii}$ and $V_{i+1,i+1}$.