1 Exercise 4.1(d)

Strings not containing $bb$ as a substring.

If you want proof, you can get it by showing two things:

1. every string generated by the grammar is in the language, and
2. every string in the language is generated by the grammar.

Proof:

1. By induction, we can see that none of the three productions will generate strings containing $bb$ as a substring.

2. If a string $x$ does not contain $bb$ as a substring, then we can write it as $x = y_0 a y_1 a ... a y_n$ for some strings $y_0, ..., y_n$, where $n$ is the number of $a$s in $x$. Each $y_i$ will be equal to either $b$ or $\Lambda$. Now it is easy to see that $S \Rightarrow^* SaSa...aS \Rightarrow^* x$ is a derivation for $x$.

2 Exercise 4.5

No.

The proof is by contradiction, so assume that the given grammar generates the given language.

Now consider the string $x = aabbcc$, which is obviously in the language.

By our assumption, we know that there must be some parse tree $T$ for $x$. Since $x$ starts with $a$, we know that the root production of $T$ must be either $S \rightarrow aSbScS$ or $S \rightarrow aScSbS$. This implies that $x$ must have a proper
substring generated by $S$, which means (by using our assumption again) that $x$ must have a proper substring in the language.

But we can easily see that none of the proper substrings of $x$ have equal numbers of $a$s, $b$s and $c$s. This means that $x$ does not have a proper substring in the language.

The previous two paragraphs are contradictory, so we can conclude that our assumption must have been false.

3 Exercise 4.10

3.1 Part b

$S \rightarrow aSb \mid Sb \mid b$

3.2 Part d

$S \rightarrow aSbb \mid aSb \mid \Lambda$

4 Exercise 4.29

4.1 Part a

The first step is to figure out what language is generated by the given grammar. It has three rules; the first one is $S \rightarrow SSS$, and this allows us to derive any odd-length string of $S$s from the initial $S$. After doing so, we can use the other two rules to turn each $S$ into either $a$ or $ab$. An equivalent regular expression would be $(a + ab)((a + ab)(a + ab))^*$.

In English, we can describe the language as

$$\{x \mid x \text{ starts with } a, \text{ and } n_a(x) \text{ is odd, and } bb \text{ is not a substring of } x\}.$$ 

A DFA for this is
and therefore, the regular grammar is

\[
\begin{align*}
S & \rightarrow aB | bD \\
A & \rightarrow aB | bS \\
B & \rightarrow aA | bC | \Lambda \\
C & \rightarrow aA | bD | \Lambda \\
D & \rightarrow aD | bD.
\end{align*}
\]

### 4.2 Part c

By reasoning similar to part (a), we determine that \((ab + ba + \Lambda)(ab + ba + \Lambda)^*(ab + aab)\) is a regular expression for the language. We can simplify this to \((ab + ba)^*(ab + aab)\). A DFA for this is
and therefore, the regular grammar is

\[
S \rightarrow aA \mid bB \\
A \rightarrow aD \mid bC \\
B \rightarrow aS \mid bE \\
C \rightarrow aA \mid bB \mid \Lambda \\
D \rightarrow aE \mid bF \\
E \rightarrow aE \mid bE \\
F \rightarrow aE \mid bE \mid \Lambda.
\]

5 Exercise 4.30

5.1 Part a

\[
S \Rightarrow S + S \\
\Rightarrow S + (S) \\
\Rightarrow S + (S \ast S) \\
\Rightarrow S + (S \ast a) \\
\Rightarrow S + (a \ast a) \\
\Rightarrow a + (a \ast a).
\]

5.2 Part b

There are 10 distinct derivations, each with length 6.

To see this, notice that every derivation starts with \( S \Rightarrow S + S \). This splits the string into two parts: the left part is \( a \), and the right part is \( (a \ast a) \). There is only one derivation of the left part, and it has length 1. There are two derivations of the right part, each with length 4. From this information, we can deduce that there are 10 derivations of the whole string because there are 5 ways to interleave a 1-step derivation with a 4-step derivation; then, since we have two 4-step derivations to choose from, we multiply 5 by 2 to get 10.

Now I just need to convince you that there are two 4-step derivations of \( (a \ast a) \). To see this, notice that every derivation starts with \( S \Rightarrow (S) \Rightarrow (S \ast S) \). This splits the string into two parts: the left part is \( a \), and the right part is \( a \).
Each part has only one derivation, and it has length 1. From this information, we can deduce that there are two derivations of \((a*a)\) because there are two ways to interleave a 1-step derivation with another 1-step derivation. Furthermore, the derivations of \((a*a)\) must have length 4 because there are 2 initial steps followed by two sub-derivations with 1 step each.

6 Exercise 4.34

To show that the grammar is ambiguous, I must simply exhibit a string with two different parse trees. Consider the string \(x = ababa\). The two parse trees are:

```
S  S
|  |
S b S
|  |
S b S
|  |
a

S  S
|  |
S b S
|  |
a
```

7 Exercise 4.38

7.1 Part b

Two parse trees for the string \(a\):

```
S
|   |
A B A
|   |
Λ Λ a A
|   |
Λ

S
|   |
A B A
|   |
Λ Λ a A
|   |
Λ
```

and an equivalent unambiguous grammar:

\[
S \rightarrow A \mid ABA \\
A \rightarrow aA \mid Λ \\
B \rightarrow bB \mid b.
\]
7.2 Part c

Two parse trees for the string $aaabb$:

and an equivalent unambiguous grammar:

\[ S \rightarrow aaSb \mid A \]
\[ A \rightarrow aAb \mid \Lambda \]

8 Exercise 4.54(c)

For full credit, you should show your work, including at least the following three stages.

Grammar after removing $\Lambda$-productions:

\[
S \rightarrow AaA \mid CA \mid BaB \mid A \mid aA \mid Aa \mid a \mid C \\
A \rightarrow aaBa \mid CDA \mid aa \mid DC \mid CA \mid C \mid DA \mid D \mid A \mid CD \\
B \rightarrow bB \mid bAB \mid bb \mid aS \mid a \\
C \rightarrow Ca \mid bC \mid D \mid b \mid a \\
D \rightarrow bD \mid b.
\]

Grammar after removing unit productions:

\[
S \rightarrow AaA \mid CA \mid BaB \mid aA \mid Aa \mid aaBa \mid CDA \\
\mid aa \mid DC \mid DA \mid CD \mid Ca \mid bC \mid b \mid bD \\
A \rightarrow aaBa \mid CDA \mid aa \mid DC \mid CA \mid DA \mid CD \mid Ca \mid bC \mid b \mid a \mid bD \\
B \rightarrow bB \mid bAB \mid bb \mid aS \mid a \\
C \rightarrow Ca \mid bC \mid b \mid a \mid bD \\
D \rightarrow bD \mid b.
\]
Final grammar after Chomskyization:

\[
S \rightarrow EA | CA | FB | XA | AX | a | HX | IA \\
| XX | DC | DA | CD | CX | YC | b | YD \\
A \rightarrow HX | IA | XX | DC | CA | DA | CD | CX | YC | b | a | YD \\
B \rightarrow YB | JB | YY | XS | a \\
C \rightarrow CX | YC | b | a | YD \\
D \rightarrow YD | b
\]

\[
E \rightarrow AX \\
F \rightarrow BX \\
G \rightarrow XX \\
H \rightarrow GB \\
I \rightarrow CD \\
J \rightarrow YA \\
X \rightarrow a \\
Y \rightarrow b.
\]