The following derivation examples illustrate some of the ways in which general grammars can work.

Find a derivation for the string *caabcb* according to the grammar shown.

$$
\begin{aligned}
S &\longrightarrow SABC \\
S &\longrightarrow \varepsilon \\
AB &\longrightarrow BA \\
BA &\longrightarrow AB \\
AC &\longrightarrow CA \\
CA &\longrightarrow AC \\
BC &\longrightarrow CB \\
CB &\longrightarrow BC \\
A &\longrightarrow a \\
B &\longrightarrow b \\
C &\longrightarrow c
\end{aligned}
$$

Answer:

$$
\begin{aligned}
S &\Longrightarrow SABC \\
&\Longrightarrow SABCABC \\
&\Longrightarrow ABCABC \\
&\Longrightarrow ACBABC \\
&\Longrightarrow CABABC \\
&\Longrightarrow CAABBC \\
&\Longrightarrow CAABCB \\
&\Longrightarrow cAABCB \\
&\Longrightarrow caABCB \\
&\Longrightarrow caaBCB \\
&\Longrightarrow caabCB \\
&\Longrightarrow caabcB \\
&\Longrightarrow caabcb
\end{aligned}
$$

Discussion: Observe first that the only way to get terminals is from the rules $A \longrightarrow a$, $B \longrightarrow b$, $C \longrightarrow c$ and that the only way to generate more copies of the non-terminals is $S \longrightarrow SABC$. So the first step is to generate enough As, Bs, Cs, and then get rid of the S.

$$\begin{aligned} S &\Longrightarrow SABC & S &\longrightarrow SABC \\ &\Longrightarrow SABCABC & S &\longrightarrow SABC \\ &\Longrightarrow ABCABC & S &\longrightarrow \epsilon \end{aligned}$$

Now observe that the rest of the productions rearrange the order of the As, Bs, Cs. We need $CAABCB$.

$$\begin{aligned} &\Longrightarrow ACBABC & BC &\longrightarrow CB \\ &\Longrightarrow CABABC & AC &\longrightarrow CA \\ &\Longrightarrow CAABBC & AB &\longrightarrow BA \\ &\Longrightarrow CAABCB & BC &\longrightarrow CB \end{aligned}$$

Finally, generate the terminals.

$$\begin{aligned} &\Longrightarrow cAABCB & C &\longrightarrow c \\ &\Longrightarrow caABCB & A &\longrightarrow a \\ &\Longrightarrow caaBCB & B &\longrightarrow b \\ &\Longrightarrow caabCB & A &\longrightarrow a \\ &\Longrightarrow caabcB & C &\longrightarrow c \\ &\Longrightarrow caabcb & B &\longrightarrow b \end{aligned}$$

Find a derivation for the string $aabbcc$ according to the grammar shown.

$$\begin{aligned}
S &\longrightarrow SABC \\
S &\longrightarrow X \\
BA &\longrightarrow AB \\
CA &\longrightarrow AC \\
CB &\longrightarrow BC \\
XA &\longrightarrow aX \\
X &\longrightarrow Y \\
YB &\longrightarrow bY \\
Y &\longrightarrow Z \\
ZC &\longrightarrow cZ \\
Z &\longrightarrow \varepsilon
\end{aligned}$$

Answer:

$$\begin{aligned}
S &\Longrightarrow SABC \\
&\Longrightarrow SABCABC \\
&\Longrightarrow XABCABC \\
&\Longrightarrow XABACBC \\
&\Longrightarrow XAABCBC \\
&\Longrightarrow XAABBCC \\
&\Longrightarrow aXABBCC \\
&\Longrightarrow aaXBBCC \\
&\Longrightarrow aaYBBCC \\
&\Longrightarrow aabYBCC \\
&\Longrightarrow aabbYCC \\
&\Longrightarrow aabbZCC \\
&\Longrightarrow aabbcZC \\
&\Longrightarrow aabbccZ \\
&\Longrightarrow aabbcc
\end{aligned}$$

Discussion: This derivation starts in a similar way — get enough As, Bs, and Cs to produce the necessary terminals.

$$\begin{aligned}
S &\Longrightarrow SABC & S &\longrightarrow SABC \\
&\Longrightarrow SABCABC & S &\longrightarrow SABC \\
&\Longrightarrow XABCABC & S &\longrightarrow X
\end{aligned}$$

Next, get the non-terminals in the right order:

$$
\begin{aligned}
&\Longrightarrow XABACBC && CA \longrightarrow AC \\
&\Longrightarrow XAABCBC && BA \longrightarrow AB \\
&\Longrightarrow XAABBCC && CB \longrightarrow BC
\end{aligned}
$$

Finally, generate the terminals:

$$
\begin{aligned}
&\Longrightarrow aXABBCC && XA \longrightarrow aX \\
&\Longrightarrow aaXBBCC && XA \longrightarrow aX \\
&\Longrightarrow aaYBBCC && X \longrightarrow Y \\
&\Longrightarrow aabYBCC && YB \longrightarrow bY \\
&\Longrightarrow aabbYCC && YB \longrightarrow bY \\
&\Longrightarrow aabbZCC && Y \longrightarrow Z \\
&\Longrightarrow aabbcZC && ZC \longrightarrow cZ \\
&\Longrightarrow aabbccZ && ZC \longrightarrow cZ \\
&\Longrightarrow aabbcc && Z \longrightarrow \epsilon
\end{aligned}
$$

In this last phase, X (and then Y and Z) sweeps across, transforming the non-terminals into terminals. This is necessary in order to delay the transformation of non-terminals into terminals until after the non-terminals have been arranged in the right order — if there were simply rules $A \longrightarrow a$ (etc), then strings like *abcabc* could be generated.

Find a derivation for the string $aaaa$ according to the grammar shown.

$$\begin{aligned}
S &\longrightarrow DTE \\
T &\longrightarrow BTA \\
T &\longrightarrow \varepsilon \\
BA &\longrightarrow AaB \\
Ba &\longrightarrow aB \\
BE &\longrightarrow E \\
DA &\longrightarrow D \\
Da &\longrightarrow aD \\
DE &\longrightarrow \varepsilon
\end{aligned}$$

Answer:

$$\begin{aligned}
S &\Longrightarrow DTE \\
&\Longrightarrow DBTAE \\
&\Longrightarrow DBBTAAE \\
&\Longrightarrow DBBAAE \\
&\Longrightarrow DBAaBAE \\
&\Longrightarrow DAaBaBAE \\
&\Longrightarrow DAaaBBAE \\
&\Longrightarrow DAaaBAaBE \\
&\Longrightarrow DAaaAaBaBE \\
&\Longrightarrow DAaaAaaBBE \\
&\Longrightarrow DAaaAaaBE \\
&\Longrightarrow DAaaAaaE \\
&\Longrightarrow DaaAaaE \\
&\Longrightarrow aDaAaaE \\
&\Longrightarrow aaDAaaE \\
&\Longrightarrow aaDaaE \\
&\Longrightarrow aaaDaE \\
&\Longrightarrow aaaaDE \\
&\Longrightarrow aaaa
\end{aligned}$$

Discussion: There is only one first step, but then the question is how many times to use the $T \longrightarrow BTA$ rule. With rules $BA \longrightarrow AaB$, $Ba \longrightarrow aB$, and $Da \longrightarrow aD$, both B and D sweep left-to-right like X did in the previous grammar, and only B (or, actually, BA) leads to more as, but it's still not entirely clear how many As and Bs to start with. Let's try a single application of $T \longrightarrow BTA$ to see what happens.

$$\begin{aligned} S &\Longrightarrow DTE & S &\longrightarrow DTE \\ &\Longrightarrow DBTAE & T &\longrightarrow BTA \\ &\Longrightarrow DBAE & T &\longrightarrow \epsilon \end{aligned}$$

Now there is only one choice:

$$\Longrightarrow DAaBE \quad BA \longrightarrow AaB$$

With A on the left and B on the right, they can be eliminated:

$$\begin{aligned} &\Longrightarrow DaBE & DA &\longrightarrow D \\ &\Longrightarrow DaE & BE &\longrightarrow E \end{aligned}$$

And finally, D is moved to the end and all of the non-terminals eliminated:

$$\begin{aligned} &\Longrightarrow aDE & Da &\longrightarrow aD \\ &\Longrightarrow a & DE &\longrightarrow \epsilon \end{aligned}$$

$1 = 1^2$, so a is a legal string and it seems like n applications of $T \longrightarrow BTA$ yield a^{n^2}. Since we want $aaaa$ (a^{2^2}), let's try two:

$$\begin{aligned} S &\Longrightarrow DTE & S &\longrightarrow DTE \\ &\Longrightarrow DBTAE & T &\longrightarrow BTA \\ &\Longrightarrow DBBTAAE & T &\longrightarrow BTA \\ &\Longrightarrow DBBAAE & T &\longrightarrow \epsilon \end{aligned}$$

Now there is only one choice:

$$\Longrightarrow DBAaBAE \quad BA \longrightarrow AaB$$

Continue with the same rule, then shift the a forward:

$$\begin{aligned} &\Longrightarrow DAaBaBAE & BA &\longrightarrow AaB \\ &\Longrightarrow DAaaBBAE & Ba &\longrightarrow aB \end{aligned}$$

$Ba \longrightarrow aB$ wasn't the only option for the previous step, but observe what the last three steps have accomplished: the BB has been moved one A to the right and two as have been produced.

Let's try a similar sequence again:

$$\begin{aligned} &\Longrightarrow DAaaBAaBE & BA &\longrightarrow AaB \\ &\Longrightarrow DAaaAaBaBE & BA &\longrightarrow AaB \\ &\Longrightarrow DAaaAaaBBE & Ba &\longrightarrow aB \end{aligned}$$

And there's a similar outcome — BB has been moved another A to the right and two as have been produced.

Now there are no more BAs, but the Bs can be cancelled:

$$\begin{aligned} &\Longrightarrow DAaaAaaBE && BE \longrightarrow E \\ &\Longrightarrow DAaaAaaE && BE \longrightarrow E \end{aligned}$$

Sweep the D forward to clean up the As:

$$\begin{aligned} &\Longrightarrow DaaAaaE && DA \longrightarrow D \\ &\Longrightarrow aDaAaaE && Da \longrightarrow aD \\ &\Longrightarrow aaDAaaE && Da \longrightarrow aD \\ &\Longrightarrow aaDaaE && DA \longrightarrow D \\ &\Longrightarrow aaaDaE && Da \longrightarrow aD \\ &\Longrightarrow aaaaDE && Da \longrightarrow aD \end{aligned}$$

And finally, clean up the remaining non-terminals:

$$\Longrightarrow aaaa \quad DE \longrightarrow \epsilon$$

So, how does this grammar work? D and E denote the end of the string and, effectively, the workspace. The group of Bs moves forward one A at a time, producing an a for each B in the group — this is where the multiplication computation is happening. The rest is bookkeeping, to clean up the As when they are no longer needed. The sweep approach with D is similar to the X in the previous grammar — it ensures that As aren't eliminated until after the Bs have passed by.