Another example of a CFG

$L = \{ a^n b^m \mid n \neq m \}$

What's a grammar for this?

\[
S \to aSb \mid A \mid B \\
A \to aA \mid a \\
B \to bB \mid b
\]

or

\[
S \to A \mid B \\
A \to aAb \mid aA \mid a \\
B \to aBb \mid Bb \mid b
\]

Moral: When you have 2 disjoint possibilities for your strings, you can branch to 2 variables.
Parsing

i.e. finding a derivation tree for a string given a grammar.

The brute-force method (exhaustive parsing):

We have a tree of possible derivation trees:

\[ S \rightarrow aSb | A | B, \quad A \rightarrow aA | a, \quad B \rightarrow bB | b \]
Nodes are possible partial derivation trees—one node is a child of another if we get it by expanding (i.e., doing a production rule with) the leftmost variable.

A leaf of this tree of derivation trees is a complete derivation tree for some string.

Exhaustive parsing is depth-first search through this tree of trees. (Stop when I match the string I am looking for.)

Do not do depth-first search; this is an infinite tree and you might go down one infinite path.

If your string can be generated by the grammar, you will eventually find it by this breadth-first search.
Problem: How do we know when to stop if our string is not generated by the grammar?

Easy sol'n: Ensure that our grammar has no productions that look like $A \Rightarrow \gamma \leftarrow$ not having these means your string can't get shorter.

or $A \Rightarrow B \leftarrow$ not having these means you can't have a loop of productions getting nowhere.

The only productions that don't make your string longer are those of the form $A \Rightarrow a$.

Note—every grammar can be transformed into an equivalent one another grammar generating the same language that does not have these kinds of productions. (unless $\gamma$ is in the language)
In this case, the largest possible number of steps to produce a string of length $k$ is $2k-1$.

So you know you are done when you have searched through $2k-1$ levels of the tree (where your string has length $k$).

There are better methods — if we have time I might spend an hour on one (after converting the grammar to a special form).

Special kinds of grammars that have better parsing methods:

1. Right-linear grammars $\leftrightarrow$ DFAs
   NFA$\leftrightarrow$ DFA

2. S-grammars: A grammar is an S-grammar if
   a) Every prod looks like $A \rightarrow a \_ \_ \_ \_ \_ a$

First thing is a left
b) Given any variable $A$ and any letter $a$, there is at most one prod. of the form

$$A \rightarrow a \_ \text{ stuff}$$

that particular var that particular letter

E.g. $S \rightarrow aSb | bSa | abSb | b$

is an $S$-grammar

but $S \rightarrow aSb | bSa | abSb | b$

is not an $S$-grammar

With an $S$-grammar, parsing is deterministic— you can always know the production to do by looking at the target string.

🤔 - Many context-free languages have no $S$-grammars.