Grammars in Action

Example: A simple programming language grammar.

\[
G: \langle \text{Exp} \rangle^* ::= \langle \text{Exp} \rangle + \langle \text{Exp} \rangle \\
| \langle \text{Exp} \rangle * \langle \text{Exp} \rangle \\
| ( \langle \text{Exp} \rangle ) \\
| a \\
| b \\
| c
\]

Terminal symbols!!!

\[
S = a \\
S = a + b \\
S = a + b * c \\
S = (a + b) * c \\
S = ((a + b)) \\
S = c(a + b) \\
S = (c) + (b) \\
S = b++
\]

\[S \in L(G)\]
Grammars in Action

- The empty symbol: `<empty>`
- The only non-terminal that does not have a rule defining it.
- That is the `<empty>` symbol derives nothing.
Consider the grammar:

\[
G: \quad \langle A \rangle^* ::= a \ \langle B \rangle \mid a \\
    \langle B \rangle ::= b \ \langle B \rangle \mid b
\]

\[
G': \quad \langle A \rangle^* ::= a \ \langle B \rangle \\
    \langle B \rangle ::= b \ \langle B \rangle \mid \text{empty}
\]
Consider the following grammar fragment:

\[
\begin{align*}
\langle \text{if-stmt} \rangle & ::= \text{if} \ \langle \text{exp} \rangle \ \text{then} \ \langle \text{stmt} \rangle \ \langle \text{else-part} \rangle \\
\langle \text{else-part} \rangle & ::= \text{else} \ \langle \text{stmt} \rangle \ | \ \langle \text{empty} \rangle \\
\langle \text{exp} \rangle & ::= \ldots \\
\langle \text{stmt} \rangle & ::= \ldots
\end{align*}
\]
2.1 a) Let $L(G)$ be the language of all strings consisting of zero or more $a$’s.

2.1 i) Let $L(G)$ be the set of strings consisting of one or more $a$’s with a comma between each $a$ and the next.

2.1 d) Let $L(G)$ be the set of all strings consisting of one or more digits 0 – 9.

HW#1 – see website