Backus-Naur form

Canonical name	BackusNaurForm
Date of creation	2013-03-22 17:37:00
Last modified on	2013-03-22 17:37:00
Owner	CWoo (3771)
Last modified by	CWoo (3771)
Numerical id	6
Author	CWoo (3771)
Entry type	Definition
Classification	msc 68Q42
Classification	msc 68Q45
Synonym	BNF
Synonym	Backus normal form

The *Backus-Naur form* (or *BNF* as it is commonly denoted) is a convenient notation used to represent context-free grammars in an intuitive and more compact manner. In a Backus-Naur form, there are only four symbols that have special meaning:

$$\texttt{<} \qquad \texttt{>} \qquad \texttt{::=} \qquad \texttt{|}$$

Given a context-free grammar (Σ, N, P, S), a non-terminal (a symbol in the alphabet N) is always enclosed in `<` and `>` (e.g. `<expression>`). A terminal (a symbol in the alphabet Σ) is often represented as itself, though in the context of computer languages a terminal symbol is often enclosed in single quotes. A production (*non-terminal* $\to$ *symbols*) in P is then represented as

$$\texttt{<}\textit{non-terminal}\texttt{>} \ \texttt{::=} \ \textit{symbols}$$

The symbol `|` is used in BNF to combine multiple productions in P into one rule. For instance, if $P := \{S \to A, S \to B\}$, then P in BNF is

$$\texttt{<}S\texttt{>} \ \texttt{::=} \ A \ \texttt{|} \ B$$

Examples.

- Let $\Sigma = \{a, b, c\}$, $N = \{S, T, U\}$ be the terminal and non-terminal alphabets of a formal grammar, and

  $$P = \{S \to aSb, S \to TU, S \to c, T \to cUc, T \to ac, U \to bT, U \to cb\}$$

  is the set of productions. Then (Σ, N, P, S) is a context-free grammar. The BNF for P is

  $$\begin{array}{lcl} \texttt{<}S\texttt{>} & \texttt{:=} & a\texttt{<}S\texttt{>}b \ \texttt{|} \ \texttt{<}T\texttt{><}U\texttt{>} \ \texttt{|} \ c \\ \texttt{<}T\texttt{>} & \texttt{:=} & c\texttt{<}U\texttt{>}c \ \texttt{|} \ ac \\ \texttt{<}U\texttt{>} & \texttt{:=} & b\texttt{<}T\texttt{>} \ \texttt{|} \ cb \end{array}$$

- For another example, let us transform the context-free grammar specified in the `http://planetmath.org/ContextFreeLanguage`parent entry to BNF. For readability, we will call S *expression*, A *term*, B *factor*, C *number*, and D *digit*. The BNF for P is then

$$
\begin{array}{rcl}
\texttt{<}\mathit{expression}\texttt{>} & \texttt{::=} & \texttt{<}\mathit{term}\texttt{>} \mid \texttt{<}\mathit{expression}\texttt{>}\,\texttt{+}\,\texttt{<}\mathit{term}\texttt{>} \mid \texttt{<}\mathit{expression}\texttt{>}\,\texttt{-}\,\texttt{<}\mathit{term}\texttt{>} \\
\texttt{<}\mathit{term}\texttt{>} & \texttt{::=} & \texttt{<}\mathit{factor}\texttt{>} \mid \texttt{<}\mathit{term}\texttt{>}\,\texttt{*}\,\texttt{<}\mathit{factor}\texttt{>} \mid \texttt{<}\mathit{term}\texttt{>}\,\texttt{/}\,\texttt{<}\mathit{factor}\texttt{>} \\
\texttt{<}\mathit{factor}\texttt{>} & \texttt{::=} & \texttt{<}\mathit{number}\texttt{>} \mid \texttt{(<}\mathit{expression}\texttt{>)} \\
\texttt{<}\mathit{number}\texttt{>} & \texttt{::=} & \texttt{<}\mathit{digit}\texttt{>} \mid \texttt{<}\mathit{number}\texttt{>}\,\texttt{<}\mathit{digit}\texttt{>} \\
\texttt{<}\mathit{digit}\texttt{>} & \texttt{::=} & \texttt{0} \mid \texttt{1} \mid \texttt{2} \mid \texttt{3} \mid \texttt{4} \mid \texttt{5} \mid \texttt{6} \mid \texttt{7} \mid \texttt{8} \mid \texttt{9}
\end{array}
\tag{1}
$$

Remark. As the syntaxes of most programming languages are context-free grammars (or very close to it), the Backus-Naur form can be used to specify these syntaxes. In fact, BNF was invented to specify the syntax of ALGOL 60.