Contex-Free Languages
- Grammars- rewrite rules to generate syntactically correct sentences Natural languages- confix-sensitive
generate symacically
correct sentences
Natural languages - contix-sensitive
Programming languages - restricted expressibility more structured/ rigid
expressibility,
more structured
Example:
- Charriple
<stmt> := <if-stmt> <while-stmt> <ded -stmt=""> </ded></while-stmt></if-stmt></stmt>
<asgn-stmt></asgn-stmt>
<if-stmt>:= if <bool-expr> then <stmt-let> else <stmt)< th=""></stmt)<></stmt-let></bool-expr></if-stmt>
<pre><while_stmt>:= while < bool-expr> then <stmt_list></stmt_list></while_stmt></pre>
<asgn-stmt>:= <var> = <arth-expr></arth-expr></var></asgn-stmt>
<stmt-list> := <stmt>; <stmt-list></stmt-list></stmt></stmt-list>
<pre><stmt-list> := <stmt>; <stmt-list> <var> := <init-char> <char> <var></var></char></init-char></var></stmt-list></stmt></stmt-list></pre>
<pre><var>:= <init_char><char><var></var></char></init_char></var></pre>
<pre> <var> := <init_char> <char> <var> <init-char> := - a b z A B Z' </init-char></var></char></init_char></var></pre>
<pre><var>:= <init_char><char><var></var></char></init_char></var></pre>
<pre> <var> := <init_char> <char> <var> <init-char> := - a b z A B Z' </init-char></var></char></init_char></var></pre>

Formal defn: A CFG is a 4-tuple
(N, E, P,S) where
(ii) P- set of non-terminals Lo Symbols on the The of productions (ii) S - set of terminals Lo alphabet of the language generated by the grammar (iii) P- set of production rates
Sympas on the
(ii) 5 - set al terminals
/ d halada 1 +10 /0 0000
coaphabet of the language
generated by the grammar
(iii) P- set of production rales
PCNx (NUS!)* La rewrite rules to
generate sentences of
PSNx (NUZ)* Les reunite rules to generale sentences of -the language
(iv) S - Stoot Symbol
SEN Gnon-terminal from which production stork
production stock
5 1 C n n 1 = -7
Eg: L= {0 ⁿ 1 ⁿ n > 0}
$S \rightarrow OS1 / \epsilon$