San José State University Department of Computer Engineering CMPE 152 Compiler Design

BNF Grammar for PASCAL

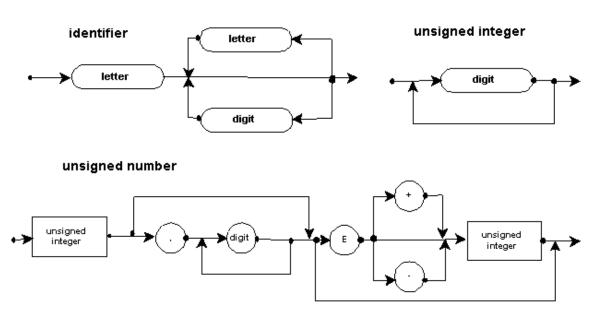
```
<letter or digit> ::= <letter> | <digit>
<br/><block> ::= <label declaration part> <constant definition part> <type definition part> <variable declaration part>
procedure and function declaration part> <statement part>
<label declaration part> ::= <empty> | label <label> {, <label>};
<label> ::= <unsigned integer>
<constant definition part> ::= <empty> | const <constant definition> { ; <constant definition> } ; <constant</pre>
definition> ::= <identifier> = <constant> <constant> ::= <unsigned number> | <sign> <unsigned number> |
<constant identifier> | <sign> <constant identifier> | <string>
<unsigned number> ::= <unsigned integer> | <unsigned real>
<unsigned integer> ::= <digit> {<digit>}
<unsigned real> ::= <unsigned integer> . <unsigned integer> | <unsigned integer> . <unsigned integer> E
<scale factor>| <unsigned integer> E <scale factor>
<scale factor> ::= <unsigned integer> | <sign> <unsigned integer>
<sign> ::= + | -
<constant identifier> ::= <identifier>
<string> ::= '<character> {<character>}'
<type definition part> ::= <empty> | type <type definition> {;<type definition>};
<type definition> ::= <identifier> = <type>
<type> ::= <simple type> | <structured type> | <pointer type>
<simple type> ::= <scalar type> | <subrange type> | <type identifier>
```

```
<scalar type> ::= (<identifier> {,<identifier>})
<subrange type> ::= <constant> .. <constant>
<type identifier> ::= <identifier>
<structured type> ::= <array type> | <record type> | <set type> | <file type>
<array type> ::= array [<index type>{,<index type>}] of <component type>
<index type> ::= <simple type>
<component type> ::= <type>
<record type> ::= record <field list> end
<field list> ::= <fixed part> | <fixed part> ; <variant part> | <variant part> |
<fixed part> ::= <record section> {;<record section>}
<record section> ::= <field identifier> {, <field identifier>} : <type> | <empty>
<variant type> ::= case <tag field> <type identifier> of <variant> { : <variant>}
<tag field> ::= <field identifier> : | <empty>
<variant> ::= <case label list> : ( <field list> ) | <empty>
<case label list> ::= <case label> {, <case label>}
<case label> ::= <constant>
<set type> ::=set of <base type>
<base type> ::= <simple type>
<file type> ::= file of <type>
<pointer type> ::= <type identifier>
<variable declaration part> ::= <empty> | var <variable declaration> {; <variable declaration> };
<variable declaration> ::= <identifier> {,<identifier>} : <type>
<procedure and function declaration part> ::= {cprocedure or function declaration > ;}
<procedure or function declaration > ::= <procedure declaration > | <function declaration >
cedure declaration> ::= cedure heading> <block>
```

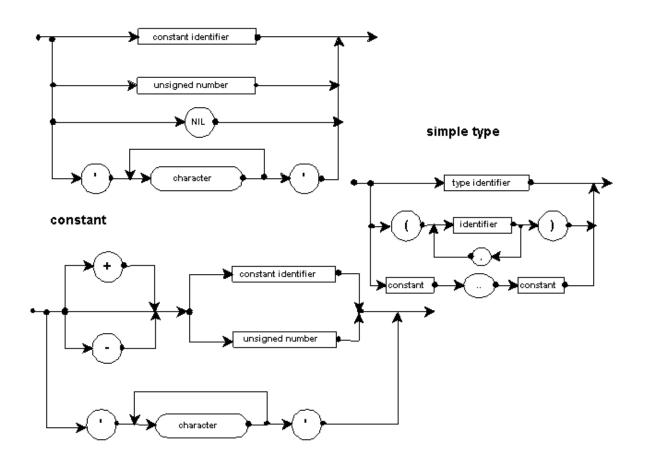
```
procedure <identifier> ( <formal parameter section> {:<formal parameter section>} );
<formal parameter section> ::= <parameter group> | var <parameter group> |
<parameter group> ::= <identifier> {, <identifier>} : <type identifier>
<function declaration> ::= <function heading> <block>
<function heading> ::= function <identifier> : <result type> ; |
function <identifier> ( <formal parameter section> {;<formal parameter section>} ) : <result type> ;
<result type> ::= <type identifier>
<statement part> ::= <compund statement>
<statement> ::= <unlabelled statement> | <label> : <unlabelled statement>
<unlabelled statement> ::= <simple statement> | <structured statement>
<simple statement> ::= <assignment statement> | <procedure statement> | <go to statement> | <empty
statement>
<assignment statement> ::= <variable> := <expression> | <function identifier> := <expression>
<variable> ::= <entire variable> | <component variable> | <referenced variable>
<entire variable> ::= <variable identifier>
<variable identifier> ::= <identifier>
<component variable> ::= <indexed variable> | <field designator> | <file buffer>
<indexed variable> ::= <array variable> [<expression> {, <expression>}]
<array variable> ::= <variable>
<field designator> ::= <record variable> . <field identifier>
<record variable> ::= <variable>
<field identifier> ::= <identifier>
<file buffer> ::= <file variable>
<file variable> ::= <variable>
<referenced variable> ::= <pointer variable>
```

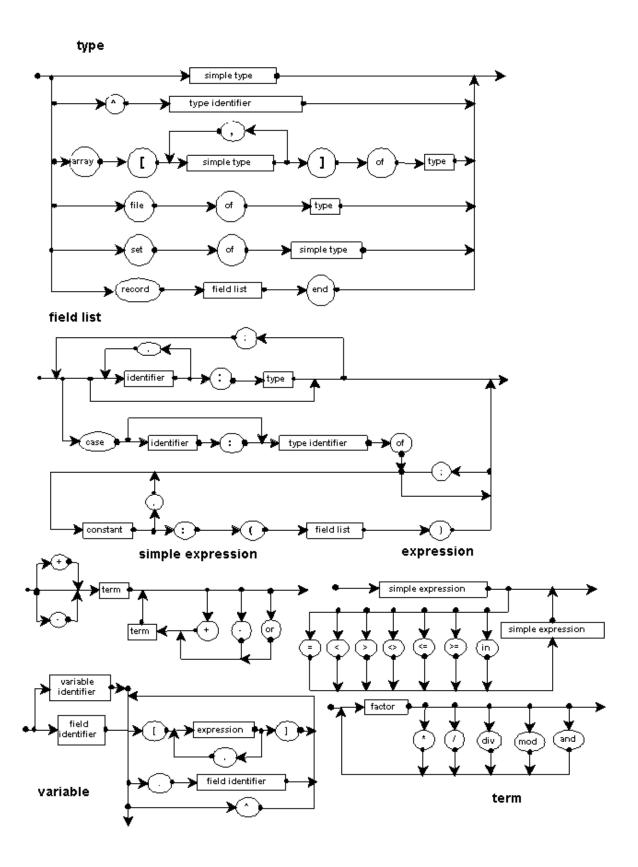
```
<pointer variable> ::= <variable>
<expression> ::= <simple expression> | <simple expression> <relational operator> <simple expression>
<relational operator> ::= = | <> | < | <= | >= | > | in
<simple expression> ::= <term> | <simple expression> <adding operator> <term>
<adding operator> ::= + | - | or
<term> ::= <factor> | <term> <multiplying operator> <factor>
<multiplying operator> ::= * | / | div | mod | and
<factor> ::= <variable> | <unsigned constant> | ( <expression> ) | <function designator> | <set> | not <factor>
<unsigned constant> ::= <unsigned number> | <string> | < constant identifier> < nil>
<function designator> ::= <function identifier> | <function identifier ( <actual parameter> {, <actual parameter>} )
<function identifier> ::= <identifier>
<set> ::= [ <element list> ]
<element list> ::= <element> {, <element> } | <empty>
<element> ::= <expression> | <expression> .. <expression>
<procedure statement> ::= <procedure identifier> | <procedure identifier> (<actual parameter> {, <actual</pre>
parameter> })
<actual parameter> ::= <expression> | <variable> | <procedure identifier> | <function identifier>
<go to statement> ::= goto <label>
<empty statement> ::= <empty>
<empty> ::=
<structured statement> ::= <compound statement> | <conditional statement> | <repetitive statement> | <with
statement>
<compound statement> ::= begin <statement> {; <statement> } end;
<conditional statement> ::= <if statement> | <case statement>
<if statement> ::= if <expression> then <statement> | if <expression> then <statement> else <statement>
<case statement> ::= case <expression> of <case list element> {; <case list element> } end
```

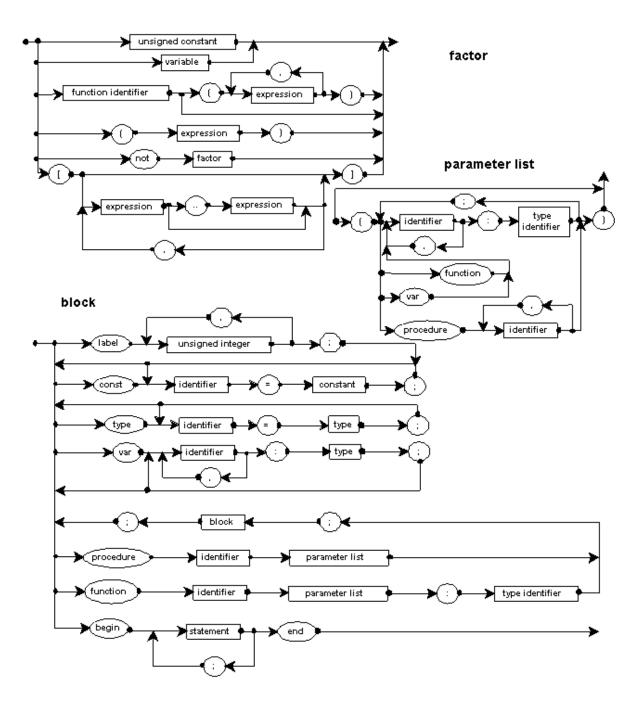
Syntax Diagrams for PASCAL



unsigned constant







program



