<Pascal> -> <Header\_program> <Var\_main>| <Header\_program> <Main\_body>

<Var\_main> -> <Var\_declaration> <Main\_body>

{PROGRAM NAME}

<Header\_program> -> <Program> <Name\_program>

<Name\_program> -> <Identifier> <Semi\_colon>

<Program> -> “program”

<Semi\_colon> -> ";"

{VARIABLE DECLARATION}

<Var\_declaration> -> <Var> <Var\_name>

<Var> -> "var"

<Var\_name> -> <Fill\_one\_var> Var\_name>

<Fill\_one\_var> -> <Adv\_colon> <Type\_var>| <Iden\_colon> <Type\_var>

<Iden\_colon> -> <Identifier><Colon>

<Identifier> -> “identifier”

<Adv\_colon> -> <Adv\_var> <Colon>

<Adv\_var> -> <Identifier> <Add\_var>

<Type\_var> -> <Type> <Semi\_colon>

<Colon> -> ":"

<Add\_var> -> <Comma> <Adv\_var>

<Identifier> -> <Alphabet> <Add\_character>| <alphabet> | <Under\_score> <Add\_character>|"\_"

<Alphabet> -> “alphabet”

<Under\_score> -> “\_”

<Add\_character> -> <Character> <Add\_character>

<Comma> -> ","

{TYPE DEFINITION}

<Type> -> “integer” | “real” | “char” |<Number\_range> <Number> | <Array\_open> <Close\_array> | <Array\_range> <Of\_range>| <Array\_comma> <Close\_array>

<Array\_range> -> <Array\_open> <Range\_close>

<Of\_range> -> <Of> <Index\_close>

<Index\_close> -> <Index\_range> <Close\_type>

<Array\_comma> -> <Array\_index> <Comma>

<Array\_index> -> <Array\_open> <Index\_range>

<Close\_type> -> <Cl\_square\_brackets> <Of\_type>

<Close\_array> -> <Range\_close> <Of\_type>

<Of\_type> -> <Of> <Simple\_type>

<Array\_open> -> <Array> <Op\_square\_brackets>

<Range\_close> -> <index\_range> <Cl\_square\_brackets>

<Simple\_type> -> “integer” | “real” | “char”| <Number\_range> <Number>

<Index\_range> -> <Number\_range> <Number>

<Number\_range> -> <Number> <Range\_separator>

<Number> -> “number”

<Range\_separator> -> “..”

<Array> -> “array”

<Op\_square\_brackets> -> “[”

<Cl\_square\_brackets> -> “]”

<Character> -> character

<Full\_stop> -> “.”

**terminal**

<number> -> < sign> <integer> <decimal\_part> | <sign> <integer> | <integer> <decimal\_part> |<sign> <origin\_digit> <integer> | <sign> <origin\_digit>

<integer> -> <sign> <origin\_digit> <integer> | <sign> <origin\_digit>

<decimal part> -> <Full\_stop> <origin\_digit> <integer>

<character> -> <origin\_digit> | <alphabet> | <symbol>

{MAIN BODY}

<Main\_body> -> <Beginstat\_end> <Full\_stop> | <Begin\_endonly> <Full\_stop>

<Begin> -> “begin”

<End> -> “end”

<Stat\_list> -> <Stat> <Stat\_list> |<Var\_defined> <Exp\_scolon>| <If\_cond> <End\_if> |<If\_cond> <Then\_end> | <While\_cond> <Do\_begin\_end>| For\_is\_way> <Op\_do\_beginend> | <For\_is\_way> <Do\_begin\_end>| <Repeat\_until> <Semi\_colon>| <Read> <Opencloseio> | <Readln> <Opencloseio>| <Write> <Opencloseio> | <Writeln> <Opencloseio>

<Stat> -> <Var\_defined> <Exp\_scolon>| <If\_cond> <End\_if> |<If\_cond> <Then\_end> | <While\_cond> <Do\_begin\_end>| For\_is\_way> <Op\_do\_beginend> | <For\_is\_way> <Do\_begin\_end>| <Repeat\_until> <Semi\_colon>| <Read> <Opencloseio> | <Readln> <Opencloseio>| <Write> <Opencloseio> | <Writeln> <Opencloseio>

<Begin\_end> -> <Beginstat\_end> <Semi\_colon>| <Begin\_endonly> <Semi\_colon>

<Begin\_endonly> -> <Begin> <End>

<Begin\_stat> -> <Begin> <Stat\_list>

<Beginstat\_end> -> <Begin\_stat> <End>

{LOGIC OPERATOR}

<logic\_op> -> “and” | “or”

{MATH OPERATOR}

<math\_op> -> “+” | “-” | “\*” | “div” | “mod”

{CONDITION OPERATOR}

<condition\_op> -> “<” | “>” | “<=” | “>=” | “=” | “<>”

{CONDITIONAL EXPRESSION}

<Condition\_exp> -> <Open\_Cond> <Exp\_ClParent>

<Open\_Cond> -> <Op\_ParentExp> <Condition\_op>

<Open\_Exp> -> <Op\_Parenthesis> <Expression>

<Exp\_Close> -> <Expression> <Cl\_Parenthesis>

{ASSIGNMENT STATEMENT}

<Assign\_stat> -> <Var\_defined> <Exp\_scolon>

<Var\_defined> -> <Variable> <Is\_defined>

<Exp\_scolon> -> <Expression> <Semi\_colon>

<Variable> -> <Identifier> | <Ident\_opsquare> <Intexp\_clsquare> | <Ident\_opintclose> <Op\_intclose> | <Ident\_Opint> <Comma\_intclose>

<Ident\_Opint> -> <Ident\_opsquare> <Int\_expression>

<Ident\_opintclose> -> <Ident\_opsquare> <Intexp\_clsquare>

<Comma\_intclose> -> <Comma> <Intexp\_clsquare>

<Op\_intclose> -> <Op\_square\_brackets> <Intexp\_clsquare>

<Ident\_opsquare> -> <Identifier> <Op\_square\_brackets>

<Intexp\_clsquare> -> <Int\_expression> <Cl\_Square\_brackets

**/\*<number> for assign “real” without operation\*/**

<Expression> -> <Factor> <Add\_factor> |<Sign> <Variable> | “integer” | <Sign\_open> <Intexp\_close> | <Quot\_char> <Single\_quotation>| “number”

<Quot\_char> -> <Single\_quotation> <Character>

<Int\_expression> -> <Factor> <Add\_factor> |<Sign> <Variable> | “integer” | <Sign\_open> <Intexp\_close>

<Add\_factor> -> <Math\_op> <Adv\_factor> | <Math\_op> <Factor>

<Adv\_factor> -> <Factor> <Add\_factor>

<Factor> -> <Sign> <Variable> | <Integer> | <Sign\_open> | <Intexp\_close>

<Sign\_open> -> <Sign> <Op\_Parenthesis>

<Intexp\_close> -> <Int\_expression> <Cl\_Parenthesis>

<Sign> -> “+” | “-”

<Op\_Parenthesis> -> “(”

<Cl\_Parenthesis> -> “)”

<Single\_quotation> -> “’”

{IF ELSE STATEMENT}

<If\_else\_stat> -> <If\_cond> <End\_if> |<If\_cond> <Then\_end>

<If\_cond> -> <If> <Condition\_exp>

<Then\_end> -> <Then> <Begin\_end>

<End\_if> -> <Then\_end> <add\_else>

<Add\_else> -> <Else> <Begin\_end> | <Else> <If\_else\_stat>

<If> -> “if”

<Else> -> “else”

<Then> -> “then”

{INPUT OUTPUT}

<Input> -> <Read> <Opencloseio> | <Readln> <Opencloseio>

<Var\_close> -> <Adv\_var> <Cl\_Parenthesis>

<Close\_Scolon> -> <Var\_close> <Semi\_colon>

<Opencloseio> -> <Op\_Parenthesis> <Close\_Scolon>

<Output> -> <Write> <Opencloseio> | <Writeln> <Opencloseio>

<Read> -> “read”

<Readln> -> “readln”

<Write> -> “write”

<Writeln> -> “writeln”

{LOOPING STATEMENT}

{WHILE STATEMENT}

<while\_stat> -> <While\_cond> <Do\_begin\_end>

<While\_cond> -> <While> <Condition\_exp>

<While> -> “while”

{REPEAT STATEMENT}

<Repeat\_stat>-> <Repeat\_until> <Semi\_colon>

<Repeat\_until> -> <Repeat\_stat> <Until\_cond>

<Repeat\_stat> -> <Repeat> <Stat\_list>

<Until\_cond> -> <Until> <Condition\_exp>

<Repeat> -> “repeat”

<Until> -> “until”

{FOR\_STATEMENT}

<Way> -> “to” | “downto”

<Op\_step> -> <Step> <Integer>

<For\_stat> -> <For\_is\_way> <Op\_do\_beginend> | <For\_is\_way> <Do\_begin\_end>

<For\_is\_way> -> <For\_is> <Way\_int>

<For\_is> -> <For\_ident> <Isdefine\_int>

<For\_ident> -> <For> <Identifier>

<Isdefine\_int> -> <Is\_defined> <Integer>

<Way\_int> -> <Way> <Integer>

<Do\_begin\_end> -> <Do> <Begin\_end>

<Op\_do\_beginend> -> <Op\_step> <Do\_begin\_end> |

<Is\_defined> -> “:=”

<Step> -> “step”

<For> -> “for”

<Do> -> “do”

<Integer> -> “integer”