Gramática en notación EBNF modificada para ser LL(1)

1. <Program> ::= **program id ;** <BlockBody> **.**
2. <BlockBody> ::= [<ConstantDefinitionPart>] [<TypeDefinitionPart>] [<VariableDefinitionPart>] {<ProcedureDefinition>} <CompoundStatement>
3. <ConstantDefinitionPart> ::= **const** <ConstantDefinition> {<ConstantDefinition>}
4. <ConstantDefinition> ::= **id =** <Constant> **;**
5. <TypeDefinitionPart> ::= **type** <TypeDefinition> {<TypeDefinition>}
6. <TypeDefinition> ::= **id =** <NewType> **;**
7. <NewType> ::= <NewArrayType> | <NewRecordType>
8. <NewArrayType> ::= **array [** <IndexRange> **] of id**
9. <IndexRange> ::= <Constant> .. <Constant>
10. <NewRecordType> ::= **record** <FieldList> **end**
11. <FieldList> ::= <RecordSection> {**;** <RecordSection>}
12. <RecordSection> ::= **id** {**,** **id**} **: id**
13. <VariableDefinitionPart> ::= **var** <VariableDefinition> {<VariableDefinition>}
14. <VariableDefinition> ::= <VariableGroup> **;**
15. <VariableGroup> ::= **id** {**,** **id**} : **id**
16. <ProcedureDefinition> ::= **procedure id** <ProcedureBlock> **;**
17. <ProcedureBlock> ::= [**(** <FormalParameterList> **)**] **;** <BlockBody>
18. <FormalParameterList> ::= <ParameterDefinition> {**;** <ParameterDefinition>}
19. <ParameterDefinition> ::= [**var**] < VariableGroup>
20. <Statement> ::= **id** <StatementGroup> | <IfStatement> | <WhileStatement> | <CompoundStatement> | ε
21. <StatementGroup> ::= {<Selector>} **:=** <Expression> | <ProcedureStatement>

    Se elimina la Regla 21

22. <ProcedureStatement> ::= **(** <ActualParameterList> **)**
23. <ActualParameterList> ::= <Expression> {, <Expression>}

    Se elimina la Regla 24

24. <IfStatement> ::= **if** <Expression> **then** <Statement> [**else** <Statement>]
25. <WhileStatement> ::= **while** <Expression> **do** <Statement>
26. <CompoundStatement> ::= **begin** <Statement> {**;** <Statement>} **end**
27. <Expression> ::= <SimpleExpression> [<RelationalOperator> <SimpleExpression>]
28. <RelationalOperator> ::= **<** | **=** | **>** | **<=** | **<>** | **>=**
29. <SimpleExpression> ::= [<SignOperator>] <Term> {<AdditiveOperator> <Term>}
30. <SignOperator> ::= **+** | **-**

31. <AdditiveOperator> ::= **+** | **-** | **or**
32. <Term> ::= <Factor> {<MultiplyingOperator> <Factor>}
33. <MultiplyingOperator> ::= ***** | **div** | **mod** | **and**
34. <Factor> ::= **numeral** | **id** {<Selector>} | **(** <Expression> **)** | **not** <Factor>
35. <Selector> ::= <IndexSelector> | <FieldSelector>
36. <IndexSelector> ::= **[** <Expression> **]**
37. <FieldSelector> ::= **. id**
38. <Constant> ::= **numeral** | **id**

Se elimina la Regla 36

Gramática en notación BNF

1. <Program> ::= **program id ;** <BlockBody> **.**
2. <BlockBody> ::= <ConstantDefinitionPart> <TypeDefinitionPart> <VariableDefinitionPart> <ProcedureDefinition> <CompoundStatement>
3. <ConstantDefinitionPart> ::= **const** <ConstantDefinition> <ConstantDefinition2> | ε
4. <ConstantDefinition> ::= **id =** <Constant> **;**
5. <ConstantDefinition2> ::= <ConstantDefinition> <ConstantDefinition2> | ε
6. <TypeDefinitionPart> ::= **type** <TypeDefinition> <TypeDefinition2> | ε
7. <TypeDefinition> ::= **id =** <NewType> **;**
8. <TypeDefinition2> ::= <TypeDefinition> <TypeDefinition2> | ε
9. <NewType> ::= <NewArrayType> | <NewRecordType>
10. <NewArrayType ::= **array [** <IndexRange> **] of id**
11. <IndexRange> ::= <Constant> **..** <Constant>
12. <NewRecordType> ::= **record** <FieldList> **end**
13. <FieldList> ::= <RecordSection> <FieldList2>
14. <FieldList2> ::= **;** <RecordSection> <FieldList2> | ε
15. <RecordSection> ::= **id** <RecordSection2> **: id**
16. <RecordSection2> ::= **, id** <RecordSection2> | ε
17. <VariableDefinitionPart> ::= **var** <VariableDefinition> <VariableDefinition2> | ε
18. <VariableDefinition> ::= <VariableGroup> **;**
19. <VariableDefinition2> ::= <VariableDefinition> <VariableDefinition2> | ε
20. <VariableGroup> ::= **id** <VariableGroup2> **: id**
21. <VariableGroup2> ::= **, id** <VariableGroup2> | ε
22. <ProcedureDefinition> ::= **procedure id** <ProcedureBlock> **;** <ProcedureDefinition> | ε
23. <ProcedureBlock> ::= <ProcedureBlock2> **;** <BlockBody>
24. <ProcedureBlock2> ::= **(<**FormalParameterList> **)** | ε
25. <FormalParameterList> ::= <ParameterDefinition> <ParameterDefinition2> | ε
26. <ParameterDefinition> ::= **var** <VariableGroup> | <VariableGroup>
27. <ParameterDefinition2> ::= **;** <ParameterDefinition> <ParameterDefinition2> | ε
28. <Statement> ::= **id** <StatementGroup> | <IfStatement> | <WhileStatement> | <CompoundStatement> | ε
29. <StatementGroup> ::= <Factor2> **:=** <Expression> | <ProcedureStatement>
30. <ProcedureStatement> ::= **(** <ActualParameterList> **)** | ε
31. <ActualParameterList> ::= <Expression> <Expression2>
32. <Expression2> ::= **,** <Expression> <Expression2> | ε

33. <IfStatement> ::= **if** <Expression> **then** <Statement> <IfStatement2>
34. <IfStatement2> ::= **else** <Statement> | ε
35. <WhileStatement> ::= **while** <Expression> **do** <Statement>
36. <CompoundStatement> ::= **begin** <Statement> <Statement2> **end**
37. <Statement2> ::= **;** <Statement> <Statement2> | ε
38. <Expression> ::= <SimpleExpression> <ExpressionGroup>
39. <ExpressionGroup> ::= <RelationalOperator> <SimpleExpression> | ε
40. <RelationalOperator> ::= **<** | **=** | **>** | **<=** | **<>** | **>=**
41. <SimpleExpression> ::= <Sign> <Term> <SimpleExpressionGroup>
42. <Sign> ::= <SignOperator> | ε
43. <SimpleExpressionGroup> ::= <AdditiveOperator> <Term> <SimpleExpressionGroup> | ε
44. <SignOperator> ::= **+** | **-**
45. <AdditiveOperator> ::= **+** | **-** | **or**
46. <Term> ::= <Factor> <Multiplying>
47. <Multiplying> ::= <MultiplyingOperator> <Factor> <Multiplying> | ε
48. <MultiplyingOperator> ::= ***** | **div** | **mod** | **and**
49. <Factor> ::= **numeral** | **id** <Factor2> | **(** <Expression> **)** | **not** <Factor>
50. <Factor2> ::= <Selector> <Factor2> | ε
51. <Selector> ::= <IndexSelector> | <FieldSelector>
52. <IndexSelector> ::= **[** <Expression> **]**
53. <FieldSelector> ::= **. id**
54. <Constant> ::= **numeral** | **id**

NO TERMINAL	FIRST	FOLLOW
Program	**program**	**$**
BlockBody	**const type var procedure begin**	**. ;**
ConstantDefinitionPart	**const ε**	**type var procedure begin**
ConstantDefinition	**id**	**id type var procedure begin**
ConstantDefinition2	**id ε**	**type var procedure begin**
TypeDefinitionPart	**type ε**	**var procedure begin**
TypeDefinition	**id**	**id var procedure begin**
TypeDefinition2	**id ε**	**var procedure begin**
NewType	**array record**	**;**
NewArrayType	**array**	**;**
IndexRange	**numeral id**	**]**
NewRecordType	**record**	**;**
FieldList	**id**	**end**
FieldList2	**; ε**	**end**
RecordSection	**id**	**; end**
RecordSection2	**, ε**	**:**
VariableDefinitionPart	**var ε**	**procedure begin**
VariableDefinition	**id**	**id procedure begin**
VariableDefinition2	**id ε**	**procedure begin**
VariableGroup	**id**	**;)**
VariableGroup2	**, ε**	**:**
ProcedureDefinition	**procedure ε**	**begin**
ProcedureBlock	**(;**	**;**
ProcedureBlock2	**(ε**	**;**
FormalParameterList	**var id ε**	**)**
ParameterDefinition	**var id**	**;)**
ParameterDefinition2	**; ε**	**)**
Statement	**id if while begin ε**	**else end ;**
StatementGroup	**[. := (ε**	**else end ;**
ProcedureStatement	**(**	**else end ;**
ActualParameterList	**+ - numeral id not (**	**)**
Expression2	**, ε**	**)**
IfStatement	**if**	**else end ;**

IfStatement2	**else ε**	**else end ;**
WhileStatement	**while**	**else end ;**
CompoundStatement	**begin**	**else end ; .**
Statement2	**; ε**	**end**
Expression	**+ - numeral id not (**	**else end ; ,) do] then**
ExpressionGroup	**< = > <= <> >= ε**	**else end ; ,) do] then**
RelationalOperator	**< = > <= <> >=**	**+ - numeral id not (**
SimpleExpression	**+ - numeral id not (**	**else end ; ,) do] then < = > <= <> >=**
Sign	**+ - ε**	**numeral id not (**
SimpleExpressionGroup	**+ - or ε**	**else end ; ,) do] then < = > <= <> >=**
SignOperator	**+ -**	**numeral id not (**
AdditiveOperator	**+ - or**	**numeral id not (**
Term	**numeral id not (**	**+ - or else end ; ,) do] then < = > <= <> >=**
Multiplying	*** div mod and ε**	**+ - or else end ; ,) do] then < = > <= <> >=**
MultiplyingOperator	*** div mod and**	**numeral id not (**
Factor	**numeral id not (**	*** div mod and + - or else end ; ,) do] then < = > <= <> >=**
Factor2	**[. ε**	**:= * div mod and + - or else end ; ,) do] then < = > <= <> >=**
Selector	**[.**	**[. := * div mod and + - or else end ; ,) do] then < = > <= <> >=**
IndexSelector	**[**	**[. := * div mod and + - or else end ; ,) do] then < = > <= <> >=**
FieldSelector	**.**	**[. := * div mod and + - or else end ; ,) do] then < = > <= <> >=**
Constant	**numeral id**	**; ..]**

PRODUCCIÓN	PREDICTION
Program -> **program id ;** BlockBody **.**	**program**
BlockBody -> ConstantDefinitionPart TypeDefinitionPart VariableDefinitionPart ProcedureDefinition CompoundStatement	**const type var procedure begin**
ConstantDefinitionPart -> **const** ConstantDefinition ConstantDefinition2	**const**
ConstantDefinitionPart -> ε	**type var procedure begin**
ConstantDefinition -> **id** = Constant **;**	**id**
ConstantDefinition2 -> ConstantDefinition ConstantDefinition2	**id**
ConstantDefinition2 -> ε	**type var procedure begin**
TypeDefinitionPart -> **type** TypeDefinition TypeDefinition 2	**type**
TypeDefinitionPart -> ε	**var procedure begin**
TypeDefinition -> **id** = NewType **;**	**id**
TypeDefinition2 -> TypeDefinition TypeDefinition2	**id**
TypeDefinition2 -> ε	**var procedure begin**
NewType -> NewArrayType	**array**
NewType -> NewRecordType	**record**
NewArrayType -> **array [** IndexRange **] of id**	**array**
IndexRange -> Constant **..** Constant	**numeral id**
NewRecordType -> **record** RecordSection FieldList **end**	**record**
FieldList -> <RecordSection> <FieldList2>	**id**
FieldList2 -> **;** RecordSection FieldList2	**;**
FieldList2 -> ε	**end**
RecordSection -> **id** RecordSection2 **: id**	**id**
RecordSection2 -> **,** RecordSection2	**,**
RecordSection2 -> ε	**:**
VariableDefinitionPart -> **var** VariableDefinition VariableDefinition2	**var**
VariableDefinitionPart -> ε	**procedure begin**
VariableDefinition -> RecordSection **;**	**id**
VariableDefinition2 -> VariableDefinition VariableDefinition2	**id**
VariableDefinition2 -> ε	**procedure begin**
VariableGroup -> **id** VariableGroup 2 **: id**	**id**
VariableGroup 2 -> **,** VariableGroup 2	**,**
VariableGroup 2 -> ε	**:**
ProcedureDefinition -> **procedure id** ProcedureBlock **;** ProcedureDefinition	**procedure**
ProcedureDefinition -> ε	**begin**

ProcedureBlock -> ProcedureBlock2 **;** BlockBody	**(;**
ProcedureBlock2 -> **(** FormalParameterList **)**	**(**
ProcedureBlock2 -> ε	**;**
FormalParameterList -> ParameterDefinition ParameterDefinition2	**var id**
FormalParameterList -> ε	**)**
ParameterDefinition -> **var** RecordSection	**var**
ParameterDefinition -> RecordSection	**id**
ParameterDefinition2 -> **;** ParameterDefinition ParameterDefinition2	**;**
ParameterDefinition2 -> ε	**)**
Statement -> **id** StatementGroup	**id**
Statement -> IfStatement	**if**
Statement -> WhileStatement	**while**
Statement -> CompoundStatement	**begin**
Statement -> ε	**else end ;**
StatementGroup -> Factor2 **:=** Expression	**[. :=**
StatementGroup -> ProcedureStatement	**(else end ;**
ProcedureStatement -> **(** ActualParameterList **)**	**(**
ProcedureStatement -> ε	**else end ;**
ActualParameterList -> Expression Expression2	**+ - numeral id not (**
Expression2 -> **,** Expression Expression2	**,**
Expression2 -> ε	**)**
IfStatement -> **if** Expression **then** Statement IfStatement2	**if**
IfStatement2 -> **else** Statement	**else**
IfStatement2 -> ε	**else end ;**
WhileStatement -> **while** Expression **do** Statement	**while**
CompoundStatement -> **begin** Statement Statement2 **end**	**begin**
Statement2 -> **;** Statement Statement2	**;**
Statement2 -> ε	**end**
Expression -> SimpleExpression ExpressionGroup	**+ - numeral id not (**
ExpressionGroup -> RelationalOperator SimpleExpression	**< = > <= <> >=**
ExpressionGroup -> ε	**else end ; ,) do] then**
RelationalOperator -> **<** | **=** | **>** | **<=** | **<>** | **>=**	**<** | **=** | **>** | **<=** | **<>** | **>=**
SimpleExpression -> Sign Term SimpleExpressionGroup	**+ - numeral id not (**

Sign -> SignOperator	**+ -**
Sign -> ε	**numeral id not (**
SimpleExpressionGroup -> AdditiveOperator Term SimpleExpressionGroup	**+ - or**
SimpleExpressionGroup -> ε	**else end ; ,) do] then < = > <= <> >=**
SignOperator -> **+** | **-**	**+** | **-**
AdditiveOperator -> **+** | **-** | **or**	**+** | **-** | **or**
Term -> Factor Multiplying	**numeral id not (**
Multiplying -> MultiplyingOperator Factor Multiplying	*** div mod and**
Multiplying -> ε	**+ - or else end ; ,) do] then < = > <= <> >=**
MultiplyingOperator -> ***** | **div** | **mod** | **and**	***** | **div** | **mod** | **and**
Factor -> **numeral**	**numeral**
Factor -> **id** Factor2	**id**
Factor -> **(** Expression **)**	**(**
Factor -> **not** Factor	**not**
Factor2 -> Selector Factor2	**[.**
Factor2 -> ε	**:= * div mod and + - or else end ; ,) do] then < = > <= <> >=**
Selector -> IndexSelector	**[**
Selector -> FieldSelector	**.**
IndexSelector -> **[** Expression **]**	**[**
FieldSelector -> **. id**	**.**
Constant -> **numeral**	**numeral**
Constant -> **id**	**id**