**Proyecto 1: Análisis Sintáctico**

**(Anexos)**

*Susana Cob García 2017136954*

*Alejandro Tapia Álvarez 2017116952*

*Compiladores e Intérpretes*

*Profesor Ignacio Trejos Zelaya*

Primer Semestre 2020

Lunes 08 de junio del 2020

**Tabla de Contenidos**

[Section 1. Anexos 3](#_Toc54172128)

[1.1 Gramática del Lenguaje Triángulo 3](#_Toc54172129)

[1.2 Léxico de Triángulo 4](#_Toc54172130)

[1.3 Etiquetas XML 5](#_Toc54172131)

Anexos

* 1. Gramática del lenguaje Triángulo

Program ::= Command

Command ::= single-Command

| Command **;** single-Command

single-Command ::=

| V-name **:=** Expression

| Identifier **(** Actual-Parameter-Sequence **)**

| **begin** Command **end**

| **let** Declaration **in** single-Command

| **if** Expression **then** single-Command **else** single-Command

| **while** Expression **do** single-Command

Expression ::= secondary-Expression

| **let** Declaration **in** Expression

| **if** Expression **then** Expression **else** Expression

secondary-Expression

::= primary-Expression

| secondary-Expression Operator primary-Expression

primary-Expression

::= Integer-Literal

| Character-Literal

| V-name

| Identifier **(** Actual-Parameter-Sequence **)**

| Operator primary-Expression

| **(** Expression **)**

| **{** Record-Aggregate **}**

| **[** Array-Aggregate **]**

Record-Aggregate ::= Identifier **~** Expression

| Identifier **~** Expression **,** Record-Aggregate

Array-Aggregate ::= Expression

| Expression **,** Array-Aggregate

V-name ::= Identifier

| V-name **.** Identifier

| V-name **[** Expression **]**

Declaration ::= single-Declaration

| Declaration **;** single-Declaration

single-Declaration

::= **const** Identifier **~** Expression

| **var** Identifier **:** Type-denoter

| **proc** Identifier **(** Formal-Parameter-Sequence **) ~**

single-Command

| **func** Identifier **(** Formal-Parameter-Sequence **)**

**:** Type-denoter **~** Expression

| **type** Identifier **~** Type-denoter

Formal-Parameter-Sequence

::=

| proper-Formal-Parameter-Sequence

proper-Formal-Parameter-Sequence

::= Formal-Parameter

| Formal-Parameter **,** proper-Formal-Parameter-Sequence

Formal-Parameter ::= Identifier **:** Type-denoter

| **var** Identifier **:** Type-denoter

| **proc** Identifier **(** Formal-Parameter-Sequence **)**

| **func** Identifier **(** Formal-Parameter-Sequence **)**

**:** Type-denoter

Actual-Parameter-Sequence

::=

| proper-Actual-Parameter-Sequence

proper-Actual-Parameter-Sequence

::= Actual-Parameter

| Actual-Parameter **,** proper-Actual-Parameter-Sequence

Actual-Parameter ::= Expression

| **var** V-name

| **proc** Identifier

| **func** Identifier

Type-denoter ::= Identifier

| **array** Integer-Literal **of** Type-denoter

| **record** Record-Type-denoter **end**

Record-Type-Denoter

::= Identifier **:** Type-denoter

| Identifier **:** Type-denoter **,** Record-Type-denoter

* 1. Léxico de Triángulo

Program ::= (Token|Comment|Blank)\*

Token ::= Integer-Literal|Character-Literal|Identifier|Operator|

**array**|**begin**|**const**|**do**|**else**|**end**|

**func**|**if**|**in**|**let**|**of**|**proc**|**record**|

**then**|**type**|**var**|**while**|

**.**|**:**|**;**|**,**|**:=**|**~**|**(**|**)**|**[**|**]**|**{**|**}**

Integer-Literal ::= Digit Digit\*

Character-Literal ::= 'Graphic'

Identifier ::= Letter(Letter|Digit)\*

Operator ::= Op-character Op-Character\*

Comment ::= **!** Graphic\* end-of-line

Blank ::= space|tab|end-of-line

Graphic ::= Letter|Digit|Op-character|space|tab|**.**|**:**|**;**|**,**|

**~**|**(**|**)**|**[**|**]**|**{**|**}**|**\_**|**|**|**!**|**'**|**`**|**"**|**#**|**$**

Letter ::= **a**|**b**|**c**|**d**|**e**|**f**|**g**|**h**|**i**|**j**|**k**|**l**|**m**|

**n**|**o**|**p**|**q**|**r**|**s**|**t**|**u**|**v**|**w**|**x**|**y**|**z**|

**A**|**B**|**C**|**D**|**E**|**F**|**G**|**H**|**I**|**J**|**K**|**L**|**M**|

**N**|**O**|**P**|**Q**|**R**|**S**|**T**|**U**|**V**|**W**|**X**|**Y**|**Z**

Digit ::= **0**|**1**|**2**|**3**|**4**|**5**|**6**|**7**|**8**|**9**

Op-character ::= **+**|**-**|**\***|**/**|**=**|**<**|**>**|**\**|**&**|**@**|**%**|**^**|**?**

* 1. Etiquetas XML

Las etiquetas de △ext están resaltadas. Las etiquetas de cierre </> se omiten en este documento, pero son parte del lenguaje XML. Tenga en cuenta las etiquetas que se abren y se cierran inmediatamente, por ejemplo, <EmptyCommand/>.

Programa

<Program>

Comandos

<AssignCommand>

<CallCommand>

<EmptyCommand/>

<IfCommand>

<LetCommand>

<SequentialCommand>

<**WhileDoCommand**>

<**DoWhileCommand**>

<**UntilDoCommand**>

<**DoUntilCommand**>

<**RepeatVarCommand**>

<**RepeatVarDeclCommand**>

<**RepeatLoopCommand**>

<**ReturnCommand/**>

<**NextCommand**> | <**NextCommand/**>

<**ExitCommand**> | <**ExitCommand/**>

Expresiones

<ArrayExpression>

<BinaryExpression>

<CallExpression>

<CharacterExpression>

<EmptyExpression/>

<IfExpression>

<IntegerExpression>

<LetExpression>

<RecordExpression>

<UnaryExpression>

<VnameExpression>

Declaraciones

<BinaryOperatorDeclaration>

<ConstDeclaration>

<FuncDeclaration>

<ProcDeclaration>

<SequentialDeclaration>

<TypeDeclaration>

<UnaryOperatorDeclaration>

<VarDeclaration>

<**ProcFuncsDeclaration**>

<**RecDeclaration**>

<**PrivateDeclaration**>

Array Aggregates (Agregados de una colección)

<MultipleArrayAggregate>

<SingleArrayAggregate>

<MultipleRecordAggregate>

<SingleRecordAggregate>

Parámetros Formales

<ConstFormalParameter>

<FuncFormalParameter>

<ProcFormalParameter>

<VarFormalParameter>

<EmptyFormalParameterSequence/>

<MultipleFormalParameterSequence>

<SingleFormalParameterSequence>

Parámetros actuales

<ConstActualParameter>

<FuncActualParameter>

<ProcActualParameter>

<VarActualParameter>

<EmptyActualParameterSequence/>

<MultipleActualParameterSequence>

<SingleActualParameterSequence>

Especificadores de Tipos

<AnyTypeDenoter/>

<ArrayTypeDenoter>

<BoolTypeDenoter/>

<CharTypeDenoter/>

<ErrorTypeDenoter/>

<SimpleTypeDenoter>

<IntTypeDenoter/>

<RecordTypeDenoter>

<MultipleFieldTypeDenoter>

<SingleFieldTypeDenoter>

Identificadores, literales y operadores4

<Identifier value=""/>

<IntegerLiteral value=""/>

<Operator value=""/>

Nombres de variables o valores

<DotVname>

<SimpleVname>

<SubscriptVname>

[[1]](#footnote-1)

1. Entre las comillas debe estar el identificador, literal u operador utilizado. [↑](#footnote-ref-1)