TC3048 Compilers Design
Dr. Héctor Ceballos / Ing. Elda Quiroga

Project name: CONCHA

February-June 2021
Tecnológico de Monterrey, Campus Monterrey

Gabriela Corona Garza
A01282529

Carolina Obregón Barrenechea
A01251983

INDEX

PROGRAMA	"programa"
IF	"si"
ELSE	"sino"
VAR	"var"
PRINT	"escribir"
WHILE	"mientras"
INT	"entero"
FLOT	"flotante"
LPARENS	"("
RPARENS	")"
CORCH_LEFT	"["
CORCH_RIGHT	"]"
FOR	"por"
FUNCION	"funcion"
LKEY	"{"
RKEY	"}"
SUM	"+"
SUB	"-"

MUL	"*"
DIV	"/"
EQ	"="
NEQ	"!="
COLN	":"
COMM	","
PTO	"."
PTOCOM	";"
MOTHN	">"
LETHN	"<"
NEQ	"!="
CONSTANTE_ENT	0 | (1...9)(0...9)*
CONSTANTE_FLOT	CONSTANTE_ENT '.' [0.9]*
STRING	"(^(" | nn))*"
ID	([a...z] | [A...Z])([a...z] | [A...Z] | [0...9])*

2.2. Syntax Diagrams for the Structures of the Language
2.2.1. Grammar

```
programa :: = PROGRAMA ID COLN prog_aux
            | PROGRAMA ID COLN prog_aux prog_aux_func
prog_aux :: = vars bloque
            | bloque
prog_aux_func :: = func prog_aux_func
                 | func
vars :: = VAR varaux COLN tipo PTOCOM
varaux :: = ID COMM varaux
         | ID
tipo :: = INT
       | FLOT
       | STR
bloque :: = LKEY bloqaux RKEY
bloqaux :: = estatuto bloqaux
          | estatuto
func :: = FUNCION ID LPARENS parms RPARENS bloque
parms :: = tipo ID COMM parms
```

```
                | tipo ID
    estatuto :: = call_func
              | declaracion
              | asignacion
              | condicion
              | escritura
              | ciclo
    call_func :: = ID LPARENS call_func_aux RPARENS PTOCOM
    call_func_aux :: = accepted_params COMM call_func_aux
                  | accepted_params
    accepted_params :: = constante
                    | STRING
    ciclo :: = wh_loop
          | for_loop
    for_loop :: = FOR LPARENS INT ID EQ exp PTOCOM expresion_comp
PTOCOM asign_op RPARENS bloque
    wh_loop :: = WHILE cond_body bloque
    declaracion :: = tipo ID PTOCOM
              | tipo ID arr_idx PTOCOM
    asignacion :: = asign_op PTOCOM
              | ID arr_idx EQ expresion PTOCOM
              | ID EQ STRING PTOCOM
    asign_op :: = ID EQ expresion
    arr_idx :: = CORCH_LEFT CONSTANTE_ENT CORCH_RIGHT
    escritura :: = PRINT LPARENS escaux RPARENS PTOCOM
    escaux :: = expresion COMM escaux
            | STRING COMM escaux
            | expresion
            | STRING
    expresion :: = expresion_comp
            | exp
    expresion_comp :: = exp MOTHN exp
            | exp LETHN exp
            | exp NEQ exp
    condicion :: = IF cond_body bloque cond_aux
    cond_body :: = LPARENS expresion RPARENS
    cond_aux :: = ELSE bloque PTOCOM
            | PTOCOM
    exp :: = termino
        | termino SUM exp
        | termino SUB exp
    termino :: = factor
```
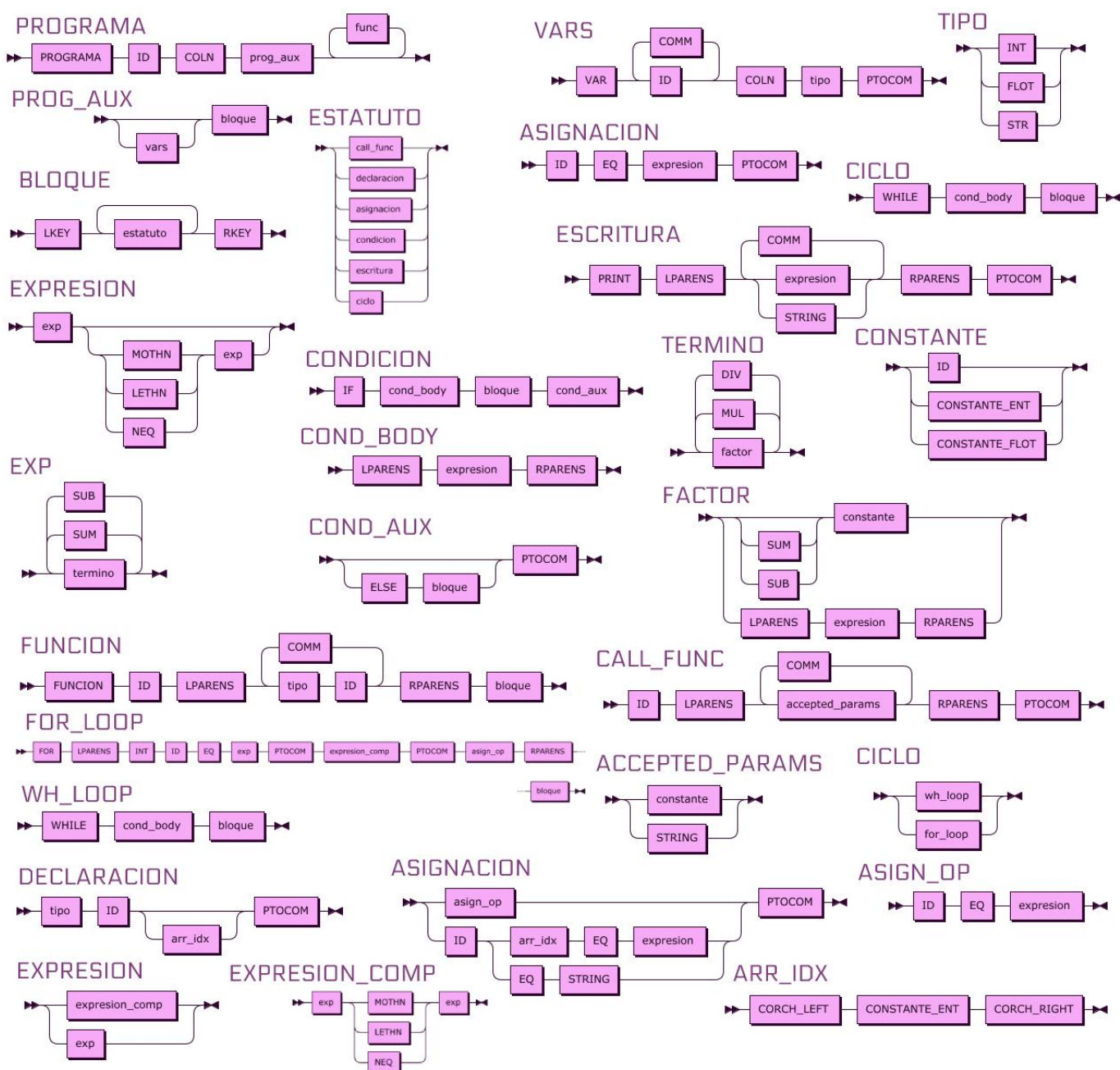
```
            | factor MUL termino
            | factor DIV termino
factor :: = SUM constante
         | SUB constante
         | constante
         | LPARENS expresion RPARENS
constante :: = ID
            | CONSTANTE_ENT
            | CONSTANTE_FLOT
```

2.2.2.   Syntax Diagrams

PROGRAMA

PROG_AUX

BLOQUE

EXPRESION

EXP

FUNCION

FOR_LOOP

WH_LOOP

DECLARACION

EXPRESION

ESTATUTO

CONDICION

COND_BODY

COND_AUX

CALL_FUNC

ACCEPTED_PARAMS

ASIGNACION

EXPRESION_COMP

VARS

ASIGNACION

ESCRITURA

TERMINO

FACTOR

CICLO

ASIGN_OP

ARR_IDX

TIPO

CICLO

CONSTANTE

2.3.   Main Semantic Characteristics

2.4.   Brief description of the special functions as well as rarely used instructions of the language

2.5.   Data Types

'Concha' will support integers, floating point numbers, strings of characters and individual characters

3.   Language and OS that will be used for development

'Concha' will be developed in Python using Lark, a modern parsing library.

4.   Bibliography