Lenguaje de programación WLOG

Joel Javier Antonio Vásquez

1 Variables

El nombre de las variables sólo podrán ser minúscula o mayúsculas en su totalidad. En caso de que el usuario requiera poner números, se le permitirá ponerlo en cualquier parte del nombre de la variable, menos al inicio. Se aceptará el símbolo '_' como parte del nombre de las variables, pero no podrá ponerse al final del nombre.

Ejemplo: variable, VARIABLE, variable1, some_variable.

2 Números

2.1 Enteros

Serán aceptados números enteros, tanto positivos como negativos.

Ejemplo: 20, -20.

2.2 Decimales

Serán aceptados números decimales. Para la parte decimal, se debe separar con el símbolo '.', se permitirá decimales tanto positivos como negativos.

Ejemplo: 10.00, -20.34.

3 Cadenas

Las cadenas deben empezar y terminar con el símbolo: ". Como parte de las cadenas, se aceptarán números, letras o símbolos especiales como: ./()\=?¿!¡"'´}{*-><][#. En caso se encuentre la cadena \n, se procederá a realizar un salto de línea.

Ejemplo: "(this) is a string?\n".

4 Operadores

4.1 Unitarios

Se permitirán los siguientes operadores unitarios:

- ++ (Aumentar en 1 a un número, sólo permitido en la parte izquierda).
- −− (Disminuir en 1 a un número, sólo permitido en la parte izquierda).
- [n] (Sirve para poder reservar memoria de tamaño n).

Ejemplo: ++variable, −−variable1, **int** some[4].

4.2 Binarios

Se permitirán los siguientes operadores binarios:

- + (Suma dos números).
- - (Resta dos números).
- * (Multiplica dos números).
- / (Divide dos números).
- % (Módulo de números).
- == (Comprobar si dos variables tienen el mismo contenido).
- != (Comprobar si dos variables tienen diferente contenidos).
- = (Almacena el contenido dado a una variable).
- > (Comprobar si un número es mayor a otro).
- >= (Comprobar si un número es mayor o igual a otro).
- < (Comprobar si un número es menor a otro).
- <= (Comprobar si un número es menor o igual a otro).

Ejemplo: variable = 5 + 3, variable1 != variable2.

5 Condicionales

Se tendrá el condicional **if** para saber si se necesita validar una acción, esta acción será declara dentro de paréntesis seguido del condicional **if**. Si se cumple la acción, será ejecutado lo que irá dentro de llaves.

Ejemplo: if(a > 4){++a}.

Si se require de una acción la cuál es ejecutada cuando no se cumple la condición, se tendrán que declarar el condicional **else**.

Ejemplo: if(a > 4){++a}else{−−a}.

Si se requiere de más acciones específicas se deberá colocar el condicional **else if**, el cual podrá ser declarado sólo si se tienen los condicionales declarados **if** y **else**.

Ejemplo: if(a > 4){++a}else if(a == 3){−−a} else {−−a}.

6 Bucles

Se tendrá el bucle **while**, el cuál cuenta con una condición que va entre paréntesis seeguido de una llaves la cuak repitirá las acciones que en encuentran en ahí hasta que la condición se cumpla.

Ejemplo: while(a > 9){−−a;}.

De igual forma, se tendrá el bucle **for**, el cual cuenta con tres pautas: La inicialización, una condición de parada y la iteración, las cuales estarán separadas por el símbolo: ;

Ejemplo: for(a=0; i<10; ++a){−−a;}.

7 Funciones

Las funciones van a contar con parámetros, los cuales serán pasados dentro de los paréntesis seguido del nombre de la función y separados por el símbolo: , Para la declaración de las funciones, vamos a tomar las mismas condiciones que para declarar el nombre de una variable. Una función puede o no retornar un valor, en caso lo retorne, tiene que ser de forma estricta sólo uno. Cada fin de línea termina con un ;

Ejemplo: int function1(int parameter1, int parameter2){a = parameter1 + parameter2;return a;}.

8 Tipos de datos

WLOG es un lenguaje tipado, por lo que se necesita declarar el tipo de variable con el cuál se va a trabajar, esto le ayudará al programador que pueda reservar un tamaño exacto de memoria. Se cuenta con los siguientes tipos:

- **int**, reserva 1 byte, 2147483648 a 2147483647.
- **float**, reserva 4 bytes, +/- 3.4e +/- 38 (7 digits).
- **char**, reserva 1 byte, -128 a 127 o 0 a 255.

En el caso de una función si no se retorna ningún tipo de valor, se deberá escribir **void** seguido del nombre de la función.

Ejemplo: void function(parameter1){++parameter1;}.

9 Función principal (main)

Se tendrá una función principal llamada **main**, donde va ser el cuerpo principal del programa y podrá server para ejecutar a otras funciones. Main, no permite paso de parámetros con lo hace las otras funciones e igual que cualquier otra función podrá retornar cualquier tipo de dato especificado en WLOG.

Ejemplo: int main(){return 0;}.

10 Terminales del Lenguaje

- **VAR**
- **NUM**
- **FLOAT**
- **CHAR**
- **STR**
- **BREAK:** break
- **TYPE:** int — float — string — char — bool — void
- **ASIG:** =
- **IF:** if

- **ELSEIF:** else if
- **ELSE:** else
- **WHILE:** while
- **FOR:** for
- **TO:** to
- **NEXT:** next
- **OP_PR:** + — -
- **OP_SE:** * — /
- **OP_LO:** AND — OR — NOT
- **CONDICIONALES:** ¡ — == — ¡= — ¿= — ¿ — !=
- **INCLUDE:** include
- **RETURN:** return

11 Gramática

Símbolo inicial

START := INCLUDE STR (R1)START — DE (R2)START — ϵ

1. if(doesExists(INCLUDE.path)) (R1)
   START.val = INCLUDE + START.val
   else ERROR

2. START := DE START (R2)
   R2 :=
   START.val = DE.val + START.val

1. IM := INCLUDE STR (R3)
   R3 :=
   if(path_exists(STR.str))
   include(STR.str);
   else ERROR

Declaración y Definición de Variable y funciones (DE)

DE := TYPE VAR LF

1. DE := TYPE VAR (R4) (R5) LF)
   R4 :=
   type = TYPE.str
   name = getName(VAR)
   setType(name, type)
   LF.type = type

LF.name = name

R5 :=
name = getName(VAR)
if(LF.array){
setArray(name, LF.array)
setSize(name, LF.size)
}
else{
setValue(name, LF.value)
}

LF := L(R6) — (R7)VF(R8) — ϵ

1. LF := L (R6) LF.array = L.array (R6)
   L.size = L.size

2. VF.type = LF.type (R7)

3. LF.val = VF.val (R8)

L := [NUM]

1. L := [NUM] (R7)
   R7 :=
   L.array = True
   L.size = NUM.str

VF := (R9)AV(R10) — AF — ϵ

1. AV.type = VF.type (R9)

2. VF.val = AV.val (R10)

AF := (PARAMETRO) { CODE }

PARAMETRO := TYPE VAR LV — ϵ

LV := , PARAMETRO — ϵ

AV := ASIG(R11)MV

if(MV.type == AV.type) (R11)
MV.val = AV.val
else ERROR

MV := [EX PL] — OP

1. MV := [EX PL] (R12)

2. MV := OP R13)

PL := , EX PL— ϵ

OP := BO(R14) — DP(R15) — TV(R16)

1. OP.val := BO.val (R14)

2. OP.val := DP.val (R15)

3. OP.val := TV.val (R16)

BO := OP_PR (R17)(R18)BOO

1. if(OP_PR.type == '+') (R17)
   BOO.op = +
   else if(OP_PR.type == '-')
   BOO.op = -
   else ERROR
2. BO.val = BOO.val (R18)

BOO := DP(R19) — TV(R20)

1. BOO.type = DP.type, BOO.name = DP.name, BOO.val = DP.val (R19)
2. BOO.type = TV.type, BOO.val = TV.val (R20)

DP := (DPP(R29))

1. DP.type = (DPP.type), DP.name_param = (DPP.name) (R29)

DPP := BO(R30) — TV(R31)

1. if(DPP.type == BO.type) (R30)
   DPP.val = BO.val
2. if(DPP.type == TV.type) (R31)
   DPP.val = TV.val

TV := PS (R26) TVV

1. TVV.type = PS.type, TVV.value = PS.value (R26)

TVV := BO(R27) — OP_SE (R28)TV — FF(R29) — ϵ

1. if(BO.type == TVV.type) (R27)
   BO.val = TVV.val
   else ERROR

2. if(TVV.type == TV.type) (R28)
   if(OP_SE == '*')
   if(TVV.type == int)
   if(TVV.type == float)
   TV.val * TVV.val
   else ERROR
   if(OP_SE == '/')
   if(TVV.type == int)
   if(TVV.type == float)
   TV.val / TVV.val
   else ERROR else ERROR else ERROR else ERROR

3. No hay reglas (R29)

EX := NUM(R32) — FLOAT(R33) — CHAR(R34) — STR(R35)

1. EX.type = NUM.type, EX.val = NUM.val (R32)
2. EX.type = FLOAT.type, EX.val = FLOAT.val (R33)
3. EX.type = CHAR.type, EX.val = CHAR.val (R34)
4. EX.type = STR.type, EX.val = STR.val (R35)

BODY

BODY := VAR (R36)BODY — IFS (R37)BODY — WH (R38)BODY — FR (R39)BODY — B (R40)BODY — R(R41) — ϵ

1. VAR.type = BODY.type, VAR.val = C.val (R36)
2. IFS.code = BODY.code (R37)
3. WH.code = BODY.code (R38)
4. FR.code = BODY.code (R39)
5. B.val = BODY.val (R40)
6. R = BODY.return (R41)

R:= RETURN (R42)PS

1. PS.val = R.val (R42)

B := BREAK(R43)

1. if(isWhile()) (R43)
   BREAK = TRUE
   else FALSE

VAR := TYPE VAR AS — VAR LAV

LAV := L AV — AV — FF

FF := (PS MPS)

MPS := , PS MPS — ϵ

AS := AV — ϵ

IFS := IF EI EL

IF := IF (CON) { CODE }

EI := ELSEIF (CON) { CODE } EI — ϵ

EL := ELSE { CODE } — ϵ

CON := (PS ZZ) MO

MO := OP_LO CON— ϵ

PS := VAR(R21) — NUM(R22) — FLOAT(R23) — CHAR(R24) — STR(R25)

1. PS.type = getType(VAR), if(doesExists(VAR)){ PS.val = VAR.str} (R21)
2. PS.type = NUM.type, PS.val = NUM.str (R22)
3. PS.type = getType(FLOAT.str), PS.val = FLOAT.str (R23)
4. PS.type = getType(CHAR.str), PS.val = CHAR.str (R24)
5. PS.type = getType(STR.str), PS.val = STR.str (R25)

WH := WHILE (CON) { CODE }

FR := FOR (FV) TO (VL) NEXT (VL) { CODE }

FV := VAR AV — FA

FA := TYPE VAR ASIG VL

VL := NUM — FLOAT

ZZ := CONDICIONALES PS — ϵ