



Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ciencias y Sistemas

Ps-Traductor

Organización de Lenguajes y Compiladores 1

Proyecto 1 - Segundo Semestre 2022


Manual Técnico

Presentado por

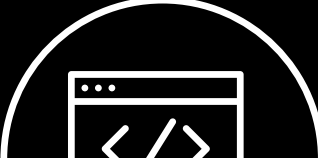
Rodrigo Hernández

Carnet

201900042



ENCUENTRO



• I. INTRODUCCIÓN	1
• II.OBJETIVOS	1
• III. DIRIGIDO	1
• IV. ESPECIFICACION TECNICA	2
1.Requerimientos de Hardware	
2.Requerimientos de Software	
• V. LÓGICA DEL PROGRAMA	3
1.Lenguaje OLC	3
2.Análisis Léxico	3
3.Análisis Sintáctico	5
4.Traducción	7
5.AST	8
6.Errores	10
• VI. CRÉDITOS	11

I. Introducción

El programa de PS-Traductor se encarga de ser un IDE del lenguaje OLC que es un lenguaje de tipo Pseudo código, lo lee, analiza y procede a traducirlo en los lenguajes de Golang y Python con las instrucciones escritas por el usuario. Todo este IDE y sus analizadores fue desarrollados en el lenguaje de programación Java con el uso de las herramientas de JFlex y JCup.

II. Objetivos

El objetivo primordial de este manual es ayudar a los distintos programadores y aspirantes al conocimiento de las ciencias de la computación así mismo del funcionamiento de los compiladores en su análisis léxico y sintáctico para la solución de problemas y desarrollar nuevos lenguajes.

El objetivo del sistema desarrollado es para que todos los aspirantes al mundo de la programación comprendan el uso del pseudocódigo y su ayuda a la hora de poder programar en los lenguajes de programación actuales como Python y Golang.

III. Dirigido

Este manual esta orientado a todos los distintos programadores interesados en el campo de las ciencias de la computación y el funcionamiento de los compiladores así mismo de conocer cómo funciona el análisis léxico y sintáctico en la lectura de nuevos lenguajes.

IV. Especificación técnica

1. Requerimientos de Hardware

- Computadora portátil o de escritorio.
- Mínimo 2GB de Memoria RAM.
- 250GB de Disco Duro o Superior.
- Procesador Intel Core i3 o superior.
- Resolución gráfica máximo 1900 x 1070 píxeles.

2. Requerimientos de Software

- Sistema Operativo Windows 7 o superior.
- Java Runtime Enviroment (JRE) versión 8.2 o superior.
- Java Development Kit (JDK) version 8.2 o superior.
- Lenguaje de Programación Java.
- NetBeans IDE 8.2 o superior.
- Acrobat Reader DC o algún lector de PDF.
- Navegador web.
- Librería Java Cup
- Librería Java Flex
- Librería rsyntaxtextarea

V. Lógica del Programa

Lenguaje OLC

Para conocer como esta construido el lenguaje OLC, puede visualizar en la página 3 del manual de usuario para entender la sintaxis del lenguaje.

Análisis Léxico

La siguiente tabla mostrará los siguientes tokens generados en el analizador Léxico.

Token	Descripción	Lexema
RMAYOR	Palabra reservada (operador)	mayor
RMENOR	Palabra reservada (operador)	menor
RMAYOROIGUAL	Palabra reservada (operador)	mayor_o_igual
RMENOROIGUAL	Palabra reservada (operador)	menor_o_igual
RESIGUAL	Palabra reservada (operador)	es_igual
RESDIFERENTE	Palabra reservada (operador)	es_diferente
ROR	Palabra reservada (operador)	or
RAND	Palabra reservada (operador)	and
RNOT	Palabra reservada (operador)	not
RVERDADERO	Palabra reservada	verdadero
RFALSO	Palabra reservada	falso
RINICIO	Palabra reservada	inicio
RFIN	Palabra reservada	fin
RINGRESAR	Palabra reservada	ingresar
RCOMO	Palabra reservada	como
RCONVALOR	Palabra reservada	con_valor
RNUMERO	Palabra reservada (Tipo de dato)	Numero
RCADENA	Palabra reservada (Tipo de dato)	Cadena
RBOOLEAN	Palabra reservada (tipo de dato)	Boolean

RCARACTER	Palabra reservada (tipo de dato)	Caracter
RSI	Palabra reservada	si
RENTONCES	Palabra reservada	entonces
RFINSI	Palabra reservada	fin_si
RDELOCONTRARIO	Palabra reservada	de_lo_contrario
ROSI	Palabra reservada	o_si
RSEGUN	Palabra reservada	segun
RHACER	Palabra reservada	hacer
RFINSEGUN	Palabra reservada	fin_segun
RPARA	Palabra reservada	para
RHASTA	Palabra reservada	hasta
RFINPARA	Palabra reservada	fin_para
RCONINCREMENTAL	Palabra reservada	con incremental
RMIENTRAS	Palabra reservada	mientras
RFINMIENTRAS	Palabra reservada	fin_mientras
RREPETIR	Palabra reservada	repetir
RHASTAQUE	Palabra reservada	hasta_que
RRETORNAR	Palabra reservada	retornar
RMETODO	Palabra reservada	metodo
RFINMETODO	Palabra reservada	fin_metodo
RCONPARAMETROS	Palabra reservada	con_parametros
RFUNCION	Palabra reservada	funcion
RFINFUNCION	Palabra reservada	fin_funcion
REJECUTAR	Palabra reservada	ejecutar
RIMPRIMIR	Palabra reservada	imprimir
RIMPRIMIRNL	Palabra reservada	imprimir_nl
PTCOMA	Carácter	;
COMA	Carácter	,
PARIZQ	Carácter	(
PARDER	Carácter)
CORIZQ	Carácter	[
CORDER	Carácter]
FLECHA	Carácter	->
INTCIERRA	Carácter	?
SUMA	Carácter	+
RESTA	Carácter	-
MULTIPLICACION	Carácter	*
DIVIDIR	Carácter	/
POTENCIA	Palabra reservada (Operación)	potencia

MODU	Palabra reservada (Operación)	modulo
------	----------------------------------	--------

La siguiente tabla mostrará las expresiones regulares utilizadas para obtener más tokens aceptados en el lenguaje.

Token	Expresión Regular	Ejemplo
INTABRE	<code>[\t]</code>	<code>\t</code>
ENTERO	<code>[0-9]+</code>	<code>2</code>
DECIMAL	<code>[0-9]+(\.[0-9]+)?</code>	<code>2.5</code>
CADENA	<code>[\"](((\\\" \\\\\\\" \\\\n \\\\\\) [^\\"\\\\n])*)[\"]</code>	<code>"hola"</code>
COMENTARIOL	<code>(\"/\".*\\r\\n*) (\"/\".*\\n*) (\"/\".*\\r\\n*) (\\//\\/(.*)*)</code>	<code>//hola</code>
COMENTARIOML	<code>[/][*][^]*[*]+([\\/*][^]*[*]+)*[/]</code>	<code>/*hola*/</code>
IDENTIFICADOR	<code>("_">([A-Za-z0-9_])+"_")</code>	<code>_a_</code>
CHARACTER	<code>("'">([A-Za-zñÑ])"'")</code>	<code>'a'</code>
CARASCCI	<code>("\"\$\"(((6[5-9]) ([7-8][0-9]) (90)) ((9[7-9]) (1[0-1][0-9]) (12[0-2])) (164) (165) (32)))\"")</code>	<code>'\${165}'</code>

Análisis Sintáctico

En este apartado se encuentra lo que es la gramática independiente del contexto con recursividad por la izquierda en la siguiente pagina o en el archivo de Gramatica.txt.

```

GRAMATICA

G = (N,T,P,S)
Donde:
  N = { NO TERMINALES }
  T = { TERMINALES }
  P = { PRODUCCIONES }
  S = { INICIO }

----- NO TERMINALES -----
NO TERMINALES = {
  INICIO, INSTRUCCIONES, DECLARACION, LISTA_IDENTIFICADORES, ASIGNACION, EXPRESION, TIPODATO, FUNCION, METODO, GLOBAL,
  LISTA_PARAMETROS, ENTORNO_LOCAL,
  CONDICIONAL_SI, CONDICION_LOCAL, LISTA_OSI, SELECCION_MULTIPLE, LISTA_VALORENTONCES, CICLO_PARA, CICLO_MIENTRAS,
  CICLO_REPETIR, EJECUTAR, IMPRESION,
  IMPRESION_CONSALTO, ALGORITMO, MAIN, COMENTARIOS, COMENTARIO, INSERCCION_PARAMETROS, EJECUTAR_2, RELACIONAL, FVM,
  AGRUPACION
}

----- TERMINALES -----
TERMINALES = {
  RENICIO, RFIN, RMETODO, RFIMETODO, RFUNCION, RFIFUNCION, RCOMPANPARAMETROS, RRETORNAR, RINGRESAR, RCOMP, RCONVALOR, OSI, RENTONCES, ROS,
  T, ROELCONTRARIO, RFINSI, RSEGUN,
  RHACER, RFSEGUN, RCOMENTARIO, RCOMENTARIOC, RPARA, RHASTA, RCONINCIDENTAL, RFINPARA, RMIENTRAS, RFIMIENTRAS, REPETIR, RHASTADE,
  REJECUTAR, RIMPRIMIR, RIMPRIMIRL,
  IDENTIFICADOR, ENTERO, CARASCCI, CADENA, CARACTER, DECIMAL, SUMA, RESTA, MULTIPLICACION, DIVIDIR, MODU, POTENCIA, RMAYOR, RMENOR, RMAYO,
  ROTIGUAL, RMENOROTIGUAL, RESIGUAL,
  RESIDIFERENTE, ROR, RAND, RNOT, RNUMERO, RCADENA, RBOOLEAN, RCARACTER, RVERDADERO, RFALSO, PARIZO, PARDER, INTABRE, INTCIERRA, FLECHA, PT,
  COMA, CORIZO, CORDER
}

----- PRODUCCIONES -----
INICIO::=INSTRUCCIONES

INSTRUCCIONES::={MAIN
               |COMENTARIOS

MAIN::=RENICIO ALGORITMO RFIN
      |RENICIO RFIN
      |error RFIN

COMENTARIOS::={COMENTARIOS COMENTARIO
               |COMENTARIO

COMENTARIO::={COMENTARIOC
              |COMENTARIOC

ALGORITMO::={ALGORITMO GLOBAL
            |GLOBAL

GLOBAL::={ ENTORNO_LOCAL
         |FVM

FVM::={FUNCION
      |METODO

DECLARACION::={RINGRESAR LISTA_IDENTIFICADORES RCOMP TIPODATO RCONVALOR CONDICION PTCOMA

LISTA_IDENTIFICADORES::={LISTA_IDENTIFICADORES COMA IDENTIFICADOR
                        |IDENTIFICADOR

EXPRESION::={ENTERO
            |DECIMAL
            |CADENA
            |CARACTER
            |CARASCCI
            |VERDADERO
            |FALSO
            |IDENTIFICADOR
            |HASTA EXPRESION
            |EXPRESION SUMA EXPRESION
            |EXPRESION RESTA EXPRESION
            |EXPRESION MULTIPLICACION EXPRESION
            |EXPRESION DIVIDIR EXPRESION
            |EXPRESION POTENCIA CORIZO EXPRESION CORDER
            |EXPRESION MODU EXPRESION
            |AGRUPACION
            |EJECUTAR_2

RELACIONAL::={EXPRESION
              |EXPRESION RMAYOR EXPRESION
              |EXPRESION RMENOR EXPRESION
              |EXPRESION RMAYOROTIGUAL EXPRESION
              |EXPRESION RMENOROTIGUAL EXPRESION
              |EXPRESION RESIGUAL EXPRESION
              |EXPRESION RESIDIFERENTE EXPRESION

CONDICION::={RELACIONAL ROR RELACIONAL
            |RELACIONAL RAND RELACIONAL
            |RNOT RELACIONAL
            |RELACIONAL

AGRUPACION::={PARIZO CONDICION PARDER

ASIGNACION::={LISTA_IDENTIFICADORES FLECHA EXPRESION PTCOMA

FUNCION::={RFUNCION IDENTIFICADOR TIPODATO LOCAL RRETORNAR CONDICION PTCOMA RFIFUNCION
          |RFUNCION IDENTIFICADOR TIPODATO RRETORNAR CONDICION PTCOMA RFSEGUN
          |RFUNCION IDENTIFICADOR TIPODATO RCOMPANPARAMETROS PARIZO LISTA_PARAMETROS PARDER LOCAL RRETORNAR CONDICION PTCOMA
          |RFIFUNCION
          |RFUNCION IDENTIFICADOR TIPODATO RCOMPANPARAMETROS PARIZO LISTA_PARAMETROS PARDER RRETORNAR CONDICION PTCOMA
          |RFIFUNCION

METODO::={RMETODO IDENTIFICADOR LOCAL RFIMETODO
         |RMETODO IDENTIFICADOR RFIMETODO
         |RMETODO IDENTIFICADOR RCOMPANPARAMETROS PARIZO LISTA_PARAMETROS PARDER LOCAL RFIMETODO
         |RMETODO IDENTIFICADOR RCOMPANPARAMETROS PARIZO LISTA_PARAMETROS PARDER RFIMETODO

LISTA_PARAMETROS::={LISTA_PARAMETROS COMA IDENTIFICADOR TIPODATO
                  |IDENTIFICADOR TIPODATO

TIPODATO::={RNUMERO
          |RCADENA
          |RBOOLEAN
          |RCARACTER

LOCAL::={LOCAL ENTORNO_LOCAL
        |ENTORNO_LOCAL

ENTORNO_LOCAL::={ASIGNACION
                |DECLARACION
                |CONDICIONAL_SI
                |SELECCION_MULTIPLE
                |CICLO_PARA
                |CICLO_MIENTRAS
                |CICLO_REPETIR
                |EJECUTAR
                |IMPRESION
                |IMPRESION_CONSALTO
                |COMENTARIOC
                |COMENTARIOC

CONDICIONAL_SI::={RSI CONDICION RENTONCES LOCAL RFINSI
                 |RSI CONDICION RENTONCES LOCAL LISTA_OSI RFINSI
                 |RSI CONDICION RENTONCES LOCAL LISTA_OSI ROELCONTRARIO LOCAL RFINSI
                 |RSI CONDICION RENTONCES LOCAL ROELCONTRARIO LOCAL RFINSI

LISTA_OSI::={LISTA_OSI OSI CONDICION RENTONCES LOCAL
            |ROSI CONDICION RENTONCES LOCAL

SELECCION_MULTIPLE::={RSEGUN EXPRESION RHACER LISTA_VALORENTONCES RFSEGUN
                    |RSEGUN EXPRESION RHACER LISTA_VALORENTONCES ROELCONTRARIO RENTONCES LOCAL RFSEGUN

LISTA_VALORENTONCES::={LISTA_VALORENTONCES INTABRE EXPRESION INTCIERRA RENTONCES LOCAL
                     |INTABRE EXPRESION INTCIERRA RENTONCES LOCAL

CICLO_PARA::={HPARA IDENTIFICADOR FLECHA EXPRESION RHASTA EXPRESION RHACER LOCAL RFINPARA
            |HPARA IDENTIFICADOR FLECHA EXPRESION RHASTA EXPRESION RCONINCIDENTAL EXPRESION RHACER LOCAL RFINPARA
            |HPARA IDENTIFICADOR FLECHA EXPRESION RHASTA EXPRESION RCONINCIDENTAL EXPRESION RHACER RFINPARA

CICLO_MIENTRAS::={RMIENTRAS CONDICION RHACER LOCAL RFIMIENTRAS
                 |RMIENTRAS CONDICION RHACER RFIMIENTRAS

CICLO_REPETIR::={REPETIR LOCAL RHASTADE CONDICION
                |REPETIR RHASTADE CONDICION

EJECUTAR::={REJECUTAR IDENTIFICADOR PARIZO PARDER PTCOMA
           |REJECUTAR IDENTIFICADOR PARIZO INSERCCION_PARAMETROS PARDER PTCOMA

EJECUTAR_2::={REJECUTAR IDENTIFICADOR PARIZO PARDER
             |REJECUTAR IDENTIFICADOR PARIZO INSERCCION_PARAMETROS PARDER

INSERCCION_PARAMETROS::={INSERCCION_PARAMETROS COMA EXPRESION
                       |EXPRESION


IMPRESION::={RIMPRIMIR EXPRESION PTCOMA

IMPRESION_CONSALTO::={RIMPRIMIRL EXPRESION PTCOMA

```


Traducción

Para lograr la traducción de pseudocódigo fue necesario el uso de una interfaz llamada Instrucción con las siguientes funciones:



```
public interface Instruccion {  
    public String traductorGolang(int identacion);  
    public String traductorPython(int identacion);  
}
```

Además se usaron las clases de:

- Asignacion
- Case
- Comentarios
- Declaracion
- Ejecutar
- Funciones
- Impresión
- Main
- Metodo
- Mientras
- Operación
- Osi
- Para
- Parametro
- Repetir
- Retornar
- SeleccionMultiple
- Si

Quienes tuvieron dentro de sus clases la interfaz de instrucción para poder traducir.

AST

Para poder producir el Abstract Syntax Tree fue necesario la creación de las siguientes clases:

Clase Nodo

```
public class Nodo {
    private String valor;
    private String tipo;
    private LinkedList<Nodo> hijos;
    private int id;

    public Nodo(String valor, String tipo) {
        this.id = 0;
        this.valor = valor;
        this.tipo = tipo;
        this.hijos = new LinkedList<Nodo>();
    }

    public String getValor() {
        return valor;
    }

    public int getId() {
        return id;
    }

    public void setValor(String valor) {
        this.valor = valor;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getTipo() {
        return tipo;
    }

    public void setTipo(String tipo) {
        this.tipo = tipo;
    }

    public LinkedList<Nodo> getHijos() {
        return hijos;
    }

    public void setHijos(LinkedList<Nodo> hijos) {
        this.hijos = hijos;
    }

    public void agregarHijo(Nodo hijo){
        this.hijos.add(hijo);
    }
}
```

Clase Arbol

```

public class Arbol {

    private Nodo raiz;

    public Arbol() {
    }

    public String codigo = "graph G {\nsplines=false;\n";

    public void graficarAST(Nodo nodo) {
        graficarNodos(nodo);
        codigo += "\n";
        System.out.println(codigo);
        try (FileOutputStream archivo = new FileOutputStream("./Reportes/AST.dot")) {
            archivo.write(codigo.getBytes());
            archivo.close();
            Runtime.getRuntime().exec("dot -Tpdf ./Reportes/AST.dot -o ./Reportes/AST.pdf");
            abrirArbol();
        } catch (Exception ex) {
            ex.printStackTrace();
        }
    }

    public void graficarNodos(Nodo nodo) {
        //ASIGNAR ID
        if (nodo.getId() == 0) {
            nodo.setId(OLC_Proyecto1_201900042.id_sig);
            OLC_Proyecto1_201900042.id_sig++;
        }
        if (nodo.getTipo() == "CADENA") {
            codigo += nodo.getId() + "[label=\"" + nodo.getValor().replace("\\", "\\\"") + "\" shape=\"circle\";]\n";
        } else {
            codigo += nodo.getId() + "[label=\"" + nodo.getValor() + "\" shape=\"circle\";]\n";
        }
        for (Nodo hijo : nodo.getHijos()) {
            codigo += nodo.getId() + " -- " + OLC_Proyecto1_201900042.id_sig + "[headport=n;]\n";
            this.graficarNodos(hijo);
        }
    }

    public void abrirArbol(){
        try
        {
            File file = new File("AST.pdf");
            if(!Desktop.isDesktopSupported())
            {
                System.out.println("not supported");
                return;
            }
            Desktop desktop = Desktop.getDesktop();
            if(file.exists())
                desktop.open(file);
            System.out.println("XD");
        }
        catch(Exception e)
        {
            e.printStackTrace();
        }
    }

}

```

Errores

Para la obtención de errores sintácticos y léxicos fue necesario el uso de esta clase:

ErrorLenguaje

```
public class ErrorLenguaje {
    String tipo,caracter, descripcion;
    int linea,columna;

    public ErrorLenguaje(String tipo,String caracter, String descripcion, int linea, int columna) {
        this.tipo = tipo;
        this.caracter = caracter;
        this.descripcion = descripcion;
        this.linea = linea;
        this.columna = columna;
    }

    public String getTipo() {
        return tipo;
    }

    public String getDescripcion() {
        return descripcion;
    }

    public int getLinea() {
        return linea;
    }

    public int getColumna() {
        return columna;
    }

    public String getCaracter() {
        return caracter;
    }
}
```

VI. Créditos

Elaborado por el estudiante Rodrigo Alejandro Hernández de León para el curso de Organización de Lenguajes y Compiladores 1, en el país de Guatemala con fecha desde el 17 de agosto de 2022 al 19 de agosto de 2022.

Repositorio ubicado en GitHub del proyecto:

<https://github.com/rodrialeh01/OLC1-201900042.git>