</> MFM SCRIPT

PROYECTO 2 - COMPILADORES 1

MANUAL TECNICO

Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Rodrigo Alejandro Hernández de León
201900042

I. Introducción

El programa de PS-Traductor se encarga de ser un IDE del lenguaje OLC que es un lenguaje de programación donde se usa la arquitectura de cliente servidor para poder interpretar el código ingresado por el usuario.

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, sintáctico y semantico para la solución de problemas y desarrollar nuevos lenguajes.

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, sintáctico y semantico 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.
- React
- Node JS
- Express JS
- Lenguaje de Programación JavaScript y TypeScript.
- Visual Studio Code o cualquier otro editor de texto.
- Navegador web.
- JISON

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
RINT	Palabra reservada	int
RDOUBLE	Palabra reservada	double
RBOOLEAN	Palabra reservada	boolean
RCHAR	Palabra reservada	char
RSTRING	Palabra reservada	string
RTRUE	Palabra reservada	true
RFALSE	Palabra reservada	false
RIF	Palabra reservada	if
RELSE	Palabra reservada	else
RELIF	Palabra reservada	elif
RSWITCH	Palabra reservada	switch
RCASE	Palabra reservada	case
RDEFAULT	Palabra reservada	default
RBREAK	Palabra reservada	break
RWHILE	Palabra reservada	while
RFOR	Palabra reservada	for
RDO	Palabra reservada	do
RUNTIL	Palabra reservada	until
RCONTINUE	Palabra reservada	continue
RRETURN	Palabra reservada	return
RVOID	Palabra reservada	void
RPRINT	Palabra reservada	print
RPRINTLN	Palabra reservada	println
RTOLOWER	Palabra reservada	toLower
RTOUPPER	Palabra reservada	toUpper
RROUND	Palabra reservada	round
RLENGTH	Palabra reservada	length
RTYPEOF	Palabra reservada	typeOf
RTOSTRING	Palabra reservada	toString
RTOCHARARRAY	Palabra reservada	toCharArray

RPUSH	Palabra reservada	push
RPOP	Palabra reservada	pop
RRUN	Palabra reservada	run
RNEW	Palabra reservada	new
INCREMENTO	Carácter	++
DECREMENTO	Carácter	--
MODULO	Carácter	%
MAYOROIGUAL	Carácter	>=
MENOROIGUAL	Carácter	<=
IGUALIGUAL	Carácter	==
DIFERENTE	Carácter	¡=
MAYOR	Carácter	>
MENOR	Carácter	<
MAS	Carácter	+
MENOS	Carácter	-
MULTIPLICACION	Carácter	*
DIVISION	Carácter	/
POTENCIA	Carácter	^
OR	Carácter	| |
AND	Carácter	&&
NOT	Carácter	¡
INTERROGACION	Carácter	?
DOSPUNTOS	Carácter	:
PARABRE	Carácter	[
PARCIERRA	Carácter	]
PTCOMA	Carácter	;
IGUAL	Carácter	=
LLAVEA	Carácter	{
LLAVEC	Carácter	}
COMA	Carácter	;
PUNTO	Carácter	.
CORABRE	Carácter	(
CORCIERRA	Carácter	)

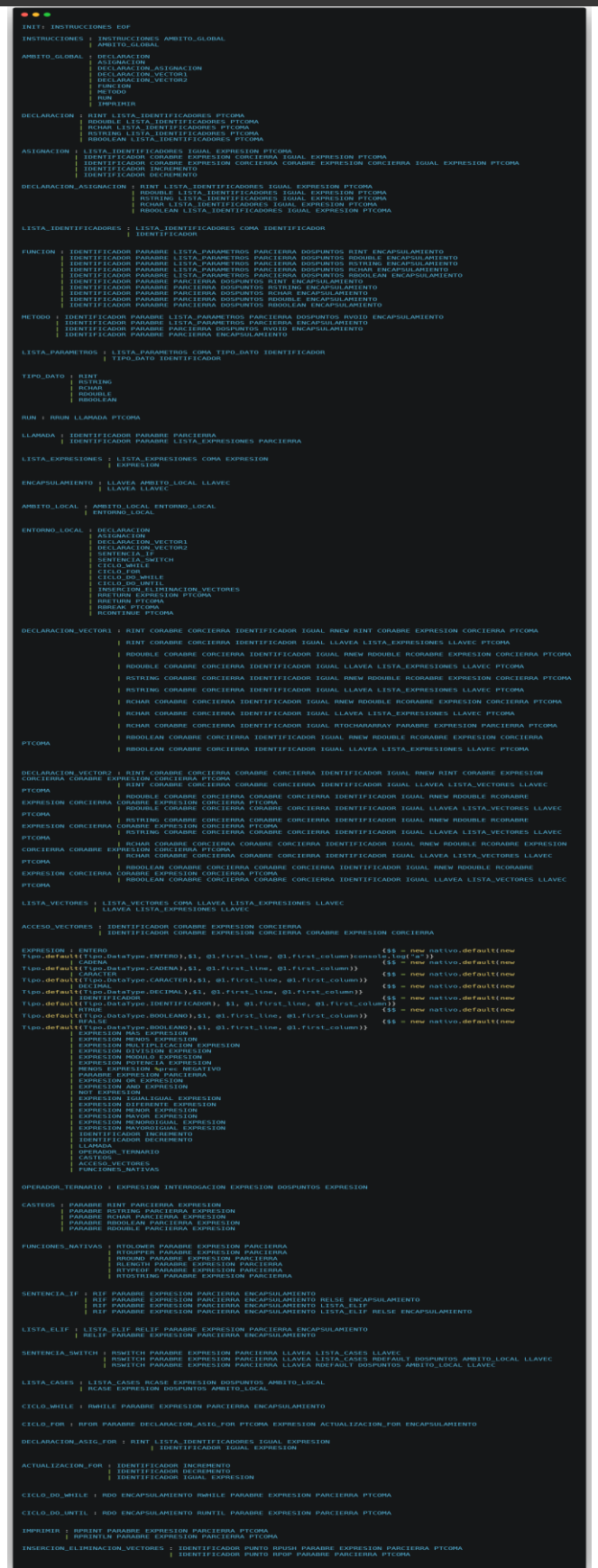
La siguiente tabla mostrará las expresiones regulares utilizadas para obtener más tokens aceptados en el lenguaje.

Token	Expresión Regular	Ejemplo
CARACTER	\'(([^\"\'\\\\]{0,1}|\\\'|\\\"|\\n|\\r|\\t|\\\\))\'	'A'
ENTERO	[0-9]+\b	2
DECIMAL	[0-9]+"."[0-9]+\b	2.5
CADENA	[\"](((\\\') | (\\\") | (\\n) | (\\t) | (\\)) | [^\\\"\n])*[\"]	"hola"

COMENTARIOL	("//".*\r\n*) | ("//".*\n*) | ("//".*\r\n*) | (\/\/(.*)*)	//hola
COMENTARIOML	[/][*][^*]*[*]+([^/*][^*]*[*]+)*[/]	/*hola*/
IDENTIFICADOR	([a-zA-Z_$])[a-zA-Z0-9_]*	Hola_a
CARACTER	(" ' "([A-Za-zñÑ])" ' ")	'a'

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.



```
INIT: INSTRUCCIONES EOF

INSTRUCCIONES : INSTRUCCIONES AMBITO_GLOBAL
              | AMBITO_GLOBAL

AMBITO_GLOBAL : DECLARACION
              | ASIGNACION
              | DECLARACION_ASIGNACION
              | DECLARACION_VECTOR1
              | DECLARACION_VECTOR2
              | FUNCION
              | METODO
              | RUN
              | IMPRIMIR

DECLARACION : RINT LISTA_IDENTIFICADORES PTCOMA
            | RDOUBLE LISTA_IDENTIFICADORES PTCOMA
            | RCHAR LISTA_IDENTIFICADORES PTCOMA
            | RSTRING LISTA_IDENTIFICADORES PTCOMA
            | RBOOLEAN LISTA_IDENTIFICADORES PTCOMA

ASIGNACION : LISTA_IDENTIFICADORES IGUAL EXPRESION PTCOMA
           | IDENTIFICADOR CORABRE EXPRESION CORCIERRA IGUAL EXPRESION PTCOMA
           | IDENTIFICADOR CORABRE EXPRESION CORCIERRA CORABRE EXPRESION CORCIERRA IGUAL EXPRESION PTCOMA
           | IDENTIFICADOR INCREMENTO
           | IDENTIFICADOR DECREMENTO

DECLARACION_ASIGNACION : RINT LISTA_IDENTIFICADORES IGUAL EXPRESION PTCOMA
                       | RDOUBLE LISTA_IDENTIFICADORES IGUAL EXPRESION PTCOMA
                       | RSTRING LISTA_IDENTIFICADORES IGUAL EXPRESION PTCOMA
                       | RCHAR LISTA_IDENTIFICADORES IGUAL EXPRESION PTCOMA
                       | RBOOLEAN LISTA_IDENTIFICADORES IGUAL EXPRESION PTCOMA


LISTA_IDENTIFICADORES : LISTA_IDENTIFICADORES COMA IDENTIFICADOR
                      | IDENTIFICADOR


FUNCION : IDENTIFICADOR PARABRE LISTA_PARAMETROS PARCIERRA DOSPUNTOS RINT ENCAPSULAMIENTO
        | IDENTIFICADOR PARABRE LISTA_PARAMETROS PARCIERRA DOSPUNTOS RDOUBLE ENCAPSULAMIENTO
        | IDENTIFICADOR PARABRE LISTA_PARAMETROS PARCIERRA DOSPUNTOS RSTRING ENCAPSULAMIENTO
        | IDENTIFICADOR PARABRE LISTA_PARAMETROS PARCIERRA DOSPUNTOS RCHAR ENCAPSULAMIENTO
        | IDENTIFICADOR PARABRE LISTA_PARAMETROS PARCIERRA DOSPUNTOS RBOOLEAN ENCAPSULAMIENTO
        | IDENTIFICADOR PARABRE PARCIERRA DOSPUNTOS RINT ENCAPSULAMIENTO
        | IDENTIFICADOR PARABRE PARCIERRA DOSPUNTOS RSTRING ENCAPSULAMIENTO
        | IDENTIFICADOR PARABRE PARCIERRA DOSPUNTOS RCHAR ENCAPSULAMIENTO
        | IDENTIFICADOR PARABRE PARCIERRA DOSPUNTOS RDOUBLE ENCAPSULAMIENTO
        | IDENTIFICADOR PARABRE PARCIERRA DOSPUNTOS RBOOLEAN ENCAPSULAMIENTO

METODO : IDENTIFICADOR PARABRE LISTA_PARAMETROS PARCIERRA DOSPUNTOS RVOID ENCAPSULAMIENTO
       | IDENTIFICADOR PARABRE LISTA_PARAMETROS PARCIERRA ENCAPSULAMIENTO
       | IDENTIFICADOR PARABRE PARCIERRA DOSPUNTOS RVOID ENCAPSULAMIENTO
       | IDENTIFICADOR PARABRE PARCIERRA ENCAPSULAMIENTO


LISTA_PARAMETROS : LISTA_PARAMETROS COMA TIPO_DATO IDENTIFICADOR
                 | TIPO_DATO IDENTIFICADOR


TIPO_DATO : RINT
          | RSTRING
          | RCHAR
          | RDOUBLE
          | RBOOLEAN


RUN : RRUN LLAMADA PTCOMA


LLAMADA : IDENTIFICADOR PARABRE PARCIERRA
        | IDENTIFICADOR PARABRE LISTA_EXPRESIONES PARCIERRA


LISTA_EXPRESIONES : LISTA_EXPRESIONES COMA EXPRESION
                  | EXPRESION


ENCAPSULAMIENTO : LLAVEA AMBITO_LOCAL LLAVEC
                | LLAVEA LLAVEC


AMBITO_LOCAL : AMBITO_LOCAL ENTORNO_LOCAL
             | ENTORNO_LOCAL


ENTORNO_LOCAL : DECLARACION
              | ASIGNACION
              | DECLARACION_VECTOR1
              | DECLARACION_VECTOR2
              | SENTENCIA_IF
              | SENTENCIA_SWITCH
              | CICLO_WHILE
              | CICLO_FOR
              | CICLO_DO_WHILE
              | CICLO_DO_UNTIL
              | INSERCION_ELIMINACION_VECTORES
              | RRETURN EXPRESION PTCOMA
              | RRETURN PTCOMA
              | RBREAK PTCOMA
              | RCONTINUE PTCOMA


DECLARACION_VECTOR1 : RINT CORABRE CORCIERRA IDENTIFICADOR IGUAL RNEW RINT CORABRE EXPRESION CORCIERRA PTCOMA

                    | RINT CORABRE CORCIERRA IDENTIFICADOR IGUAL LLAVEA LISTA_EXPRESIONES LLAVEC PTCOMA

                    | RDOUBLE CORABRE CORCIERRA IDENTIFICADOR IGUAL RNEW RDOUBLE RCORABRE EXPRESION CORCIERRA PTCOMA

                    | RDOUBLE CORABRE CORCIERRA IDENTIFICADOR IGUAL LLAVEA LISTA_EXPRESIONES LLAVEC PTCOMA

                    | RSTRING CORABRE CORCIERRA IDENTIFICADOR IGUAL RNEW RDOUBLE RCORABRE EXPRESION CORCIERRA PTCOMA

                    | RSTRING CORABRE CORCIERRA IDENTIFICADOR IGUAL LLAVEA LISTA_EXPRESIONES LLAVEC PTCOMA

                    | RCHAR CORABRE CORCIERRA IDENTIFICADOR IGUAL RNEW RDOUBLE RCORABRE EXPRESION CORCIERRA PTCOMA

                    | RCHAR CORABRE CORCIERRA IDENTIFICADOR IGUAL LLAVEA LISTA_EXPRESIONES LLAVEC PTCOMA

                    | RCHAR CORABRE CORCIERRA IDENTIFICADOR IGUAL RTOCHARARRAY PARABRE EXPRESION PARCIERRA PTCOMA

                    | RBOOLEAN CORABRE CORCIERRA IDENTIFICADOR IGUAL RNEW RDOUBLE RCORABRE EXPRESION CORCIERRA
PTCOMA

                    | RBOOLEAN CORABRE CORCIERRA IDENTIFICADOR IGUAL LLAVEA LISTA_EXPRESIONES LLAVEC PTCOMA


DECLARACION_VECTOR2 : RINT CORABRE CORCIERRA CORABRE CORCIERRA IDENTIFICADOR IGUAL RNEW RINT CORABRE EXPRESION
CORCIERRA CORABRE EXPRESION CORCIERRA PTCOMA
                    | RINT CORABRE CORCIERRA CORABRE CORCIERRA IDENTIFICADOR IGUAL LLAVEA LISTA_VECTORES LLAVEC
PTCOMA

                    | RDOUBLE CORABRE CORCIERRA CORABRE CORCIERRA IDENTIFICADOR IGUAL RNEW RDOUBLE RCORABRE
EXPRESION CORCIERRA CORABRE EXPRESION CORCIERRA PTCOMA
                    | RDOUBLE CORABRE CORCIERRA CORABRE CORCIERRA IDENTIFICADOR IGUAL LLAVEA LISTA_VECTORES LLAVEC
PTCOMA

                    | RSTRING CORABRE CORCIERRA CORABRE CORCIERRA IDENTIFICADOR IGUAL RNEW RDOUBLE RCORABRE
EXPRESION CORCIERRA CORABRE EXPRESION CORCIERRA PTCOMA
                    | RSTRING CORABRE CORCIERRA CORABRE CORCIERRA IDENTIFICADOR IGUAL LLAVEA LISTA_VECTORES LLAVEC
PTCOMA

                    | RCHAR CORABRE CORCIERRA CORABRE CORCIERRA IDENTIFICADOR IGUAL RNEW RDOUBLE RCORABRE EXPRESION
CORCIERRA CORABRE EXPRESION CORCIERRA PTCOMA
                    | RCHAR CORABRE CORCIERRA CORABRE CORCIERRA IDENTIFICADOR IGUAL LLAVEA LISTA_VECTORES LLAVEC
PTCOMA

                    | RBOOLEAN CORABRE CORCIERRA CORABRE CORCIERRA IDENTIFICADOR IGUAL RNEW RDOUBLE RCORABRE
EXPRESION CORCIERRA CORABRE EXPRESION CORCIERRA PTCOMA
                    | RBOOLEAN CORABRE CORCIERRA CORABRE CORCIERRA IDENTIFICADOR IGUAL LLAVEA LISTA_VECTORES LLAVEC
PTCOMA


LISTA_VECTORES : LISTA_VECTORES COMA LLAVEA LISTA_EXPRESIONES LLAVEC
               | LLAVEA LISTA_EXPRESIONES LLAVEC


ACCESO_VECTORES : IDENTIFICADOR CORABRE EXPRESION CORCIERRA
                | IDENTIFICADOR CORABRE EXPRESION CORCIERRA CORABRE EXPRESION CORCIERRA


EXPRESION : ENTERO                                                              {$$ = new nativo.default(new
Tipo.default(Tipo.DataType.ENTERO),$1, @1.first_line, @1.first_column);console.log("a")}
          | CADENA                                                              {$$ = new nativo.default(new
Tipo.default(Tipo.DataType.CADENA),$1, @1.first_line, @1.first_column)}
          | CARACTER                                                            {$$ = new nativo.default(new
Tipo.default(Tipo.DataType.CARACTER),$1, @1.first_line, @1.first_column)}
          | DECIMAL                                                             {$$ = new nativo.default(new
Tipo.default(Tipo.DataType.DECIMAL),$1, @1.first_line, @1.first_column)}
          | IDENTIFICADOR                                                       {$$ = new nativo.default(new
Tipo.default(Tipo.DataType.IDENTIFICADOR), $1, @1.first_line, @1.first_column)}
          | RTRUE                                                               {$$ = new nativo.default(new
Tipo.default(Tipo.DataType.BOOLEANO),$1, @1.first_line, @1.first_column)}
          | RFALSE                                                              {$$ = new nativo.default(new
Tipo.default(Tipo.DataType.BOOLEANO),$1, @1.first_line, @1.first_column)}
          | EXPRESION MAS EXPRESION
          | EXPRESION MENOS EXPRESION
          | EXPRESION MULTIPLICACION EXPRESION
          | EXPRESION DIVISION EXPRESION
          | EXPRESION MODULO EXPRESION
          | EXPRESION POTENCIA EXPRESION
          | MENOS EXPRESION %prec NEGATIVO
          | PARABRE EXPRESION PARCIERRA
          | EXPRESION OR EXPRESION
          | EXPRESION AND EXPRESION
          | NOT EXPRESION
          | EXPRESION IGUALIGUAL EXPRESION
          | EXPRESION DIFERENTE EXPRESION
          | EXPRESION MENOR EXPRESION
          | EXPRESION MAYOR EXPRESION
          | EXPRESION MENORIGUAL EXPRESION
          | EXPRESION MAYORIGUAL EXPRESION
          | IDENTIFICADOR INCREMENTO
          | IDENTIFICADOR DECREMENTO
          | LLAMADA
          | OPERADOR_TERNARIO
          | CASTEOS
          | ACCESO_VECTORES
          | FUNCIONES_NATIVAS


OPERADOR_TERNARIO : EXPRESION INTERROGACION EXPRESION DOSPUNTOS EXPRESION


CASTEOS : PARABRE RINT PARCIERRA EXPRESION
        | PARABRE RSTRING PARCIERRA EXPRESION
        | PARABRE RCHAR PARCIERRA EXPRESION
        | PARABRE RBOOLEAN PARCIERRA EXPRESION
        | PARABRE RDOUBLE PARCIERRA EXPRESION


FUNCIONES_NATIVAS : RTOLOWER PARABRE EXPRESION PARCIERRA
                  | RTOUPPER PARABRE EXPRESION PARCIERRA
                  | RROUND PARABRE EXPRESION PARCIERRA
                  | RLENGTH PARABRE EXPRESION PARCIERRA
                  | RTYPEOF PARABRE EXPRESION PARCIERRA
                  | RTOSTRING PARABRE EXPRESION PARCIERRA


SENTENCIA_IF : RIF PARABRE EXPRESION PARCIERRA ENCAPSULAMIENTO
             | RIF PARABRE EXPRESION PARCIERRA ENCAPSULAMIENTO RELSE ENCAPSULAMIENTO
             | RIF PARABRE EXPRESION PARCIERRA ENCAPSULAMIENTO LISTA_ELIF
             | RIF PARABRE EXPRESION PARCIERRA ENCAPSULAMIENTO LISTA_ELIF RELSE ENCAPSULAMIENTO


LISTA_ELIF : LISTA_ELIF RELIF PARABRE EXPRESION PARCIERRA ENCAPSULAMIENTO
           | RELIF PARABRE EXPRESION PARCIERRA ENCAPSULAMIENTO


SENTENCIA_SWITCH : RSWITCH PARABRE EXPRESION PARCIERRA LLAVEA LISTA_CASES LLAVEC
                 | RSWITCH PARABRE EXPRESION PARCIERRA LLAVEA LISTA_CASES RDEFAULT DOSPUNTOS AMBITO_LOCAL LLAVEC
                 | RSWITCH PARABRE EXPRESION PARCIERRA LLAVEA RDEFAULT DOSPUNTOS AMBITO_LOCAL LLAVEC


LISTA_CASES : LISTA_CASES RCASE EXPRESION DOSPUNTOS AMBITO_LOCAL
            | RCASE EXPRESION DOSPUNTOS AMBITO_LOCAL


CICLO_WHILE : RWHILE PARABRE EXPRESION PARCIERRA ENCAPSULAMIENTO


CICLO_FOR : RFOR PARABRE DECLARACION_ASIG_FOR PTCOMA EXPRESION ACTUALIZACION_FOR ENCAPSULAMIENTO


DECLARACION_ASIG_FOR : RINT LISTA_IDENTIFICADORES IGUAL EXPRESION
                     | IDENTIFICADOR IGUAL EXPRESION


ACTUALIZACION_FOR : IDENTIFICADOR INCREMENTO
                  | IDENTIFICADOR DECREMENTO
                  | IDENTIFICADOR IGUAL EXPRESION


CICLO_DO_WHILE : RDO ENCAPSULAMIENTO RWHILE PARABRE EXPRESION PARCIERRA PTCOMA


CICLO_DO_UNTIL : RDO ENCAPSULAMIENTO RUNTIL PARABRE EXPRESION PARCIERRA PTCOMA


IMPRIMIR : RPRINT PARABRE EXPRESION PARCIERRA PTCOMA
         | RPRINTLN PARABRE EXPRESION PARCIERRA PTCOMA


INSERCION_ELIMINACION_VECTORES : IDENTIFICADOR PUNTO RPUSH PARABRE EXPRESION PARCIERRA PTCOMA
                               | IDENTIFICADOR PUNTO RPOP PARABRE PARCIERRA PTCOMA
```

Instrucción

Para lograr la interpretación de codigo se necesito de esta clase:

```
export abstract class Instruccion {
    public linea: number;
    public columna: number;
    exp: any;

    constructor(linea: number, columna: number) {
        this.linea = linea;
        this.columna = columna;
    }
    abstract interpretar(arbol: Arbol, tabla: TablaSimbolo): any;
}
```

Además se usaron las clases de:

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

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

AST

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

Clase Nodo

```
export default class Nodo{
    valor: String;
    tipo: String;
    hijos:Nodo[];
    id: number;
    constructor(valor:any, tipo:any){
        this.id = 0;
        this.valor = valor;
        this.tipo = tipo;
        this.hijos = [];
    }
    getValor(){
        return this.valor;
    }
    getTipo(){
        return this.tipo;
    }
    agregarHijo(nodo:Nodo){
        this.hijos.push(nodo);
    }
}
```

Clase GenerarAST

```
import Nodo from "./Nodo";
var id_n = 1;
export default class GenerarAST{
    cadena:String
    constructor(){
        this.cadena = "graph G {\n";
    }
    recorrer_arbol(nodo: Nodo){
        if(nodo.id == 0){
            nodo.id = id_n;
            id_n++;
        }
        this.cadena+=nodo.id + " [shape=\"circle\" label=\"" + nodo.valor + "\" ];\n";
        nodo.hijos.forEach(element => {
            this.cadena+=nodo.id + " -- " + id_n + ";\n";
            this.recorrer_arbol(element);
        });
    }
}
```

Errores

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

Error

```
export default class Error {
    private tipoError: tipoErr;
    private desc: String;
    private fila: number;
    private columna: number;

    public getDesc(): String{
        return this.desc;
    }
    public getTipoError(): tipoErr {
        return this.tipoError;
    }
    public getColumna(): number {
        return this.columna;
    }
    public getFila(): number {
        return this.fila;
    }
    constructor(tipo: tipoErr, desc: String, fila: number, columna: number){
        this.tipoError = tipo;
        this.desc = desc;
        this.fila = fila;
        this.columna = columna;
    }
    public returnError(): String {
        if(this.getTipoError() === tipoErr.LEXICO){
            return (
                'Se obtuvo: ERROR LEXICO' +
                ' desc: {' + this.desc +
                '} en la fila: ' + this.fila +
                ' en la columna: ' + this.columna + '\n'
            );
        }else if(this.getTipoError() === tipoErr.SINTACTICO){
            return (
                'Se obtuvo: ERROR SINTACTICO' +
                ' desc: {' + this.desc +
                '} en la fila: ' + this.fila +
                ' en la columna: ' + this.columna + '\n'
            );
        }else if(this.getTipoError() === tipoErr.SEMANTICO){
            return (
                'Se obtuvo: ERROR SEMANTICO' +
                ' desc: {' + this.desc +
                '} en la fila: ' + this.fila +
                ' en la columna: ' + this.columna + '\n'
            );
        }else{
            return '';
        }
    }
}
```

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 octubre de 2022 al 04 de noviembre de 2022.

Repositorio ubicado en GitHub del proyecto:

https://github.com/rodrialeh01/OLC1-Proyecto2_201900042