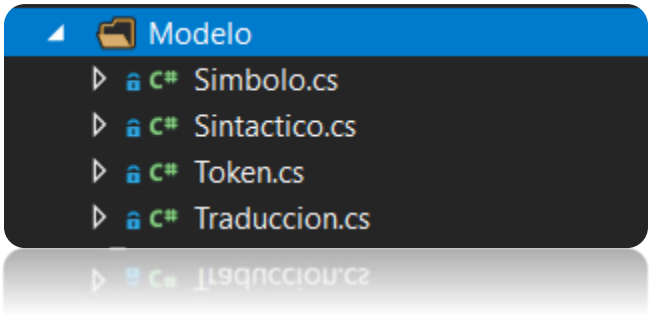
MANUAL TÉCNICO

José Rafael Morente González - 201801237

LENGUAJES FORMALES Y DE PROGRAMACIÓN

CONTENIDO

Modelos



Simbolo.cs

Este modelo es utilizado por la tabla de simbolos para almacenar los valores de las variables.

Sintactico.cs

Este modelo es utilizado para almacenar los errores sintacticos que puedan venir en el programa.

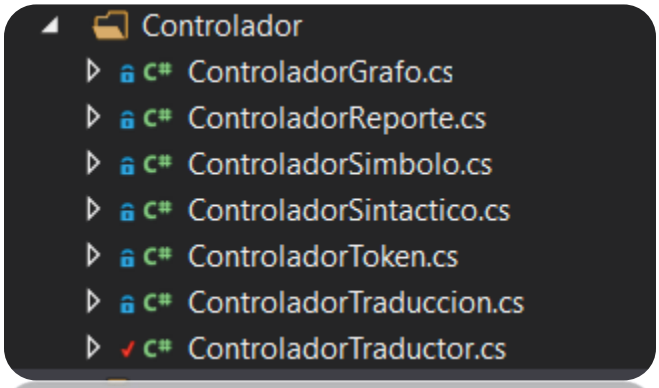
Token.cs

Este modelo es utilizado para administrar las propiedades del token en él analizador.

Traduccion.cs

Este modelo es utilizado para administrar los tokens utilizados para traducir en él lenguaje.

Controladores



ControladorGrafo.cs

Este controlador lo que realiza es generar los grafos de los arreglos.

ControladorReporte.cs

Este controlador nos trae los cuatro reportes en HTML, el de errores sintacticos, tokens, tokens errores y tabla de símbolo.

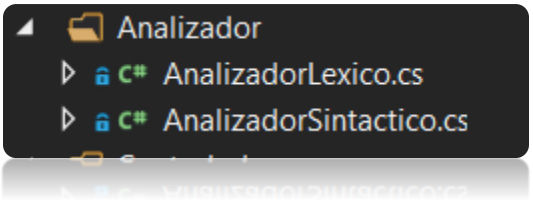
ControladorSimbolo.cs, ControladorSintactico.cs, ControladorToken.cs, ControladorTraduccion.cs

Este controlador es utilizado para administrar y agregar los tokens correspondientes a cada clase.

ControladorTraductor.cs

Este controlador es el que se encarga de la interpretación y conversión del lenguaje de C# a Python.

Analizador



AnalizadorLexico.cs

El analizador léxico es la clase en la cual nosotros aceptamos cadenas de caracteres para nuestro lenguaje.

AnalizadorSintactico.cs

El analizador sintactico es donde nosotros verificamos el orden correcto de las palabras de nuestro lenguaje que queremos traducir para ello se utilizó un método llamado Match() el cual empareja los tokens correspondientes y su nivel de recuperación es a modo pánico hasta encontrar una palabra de la gramática el cual es punto y coma.

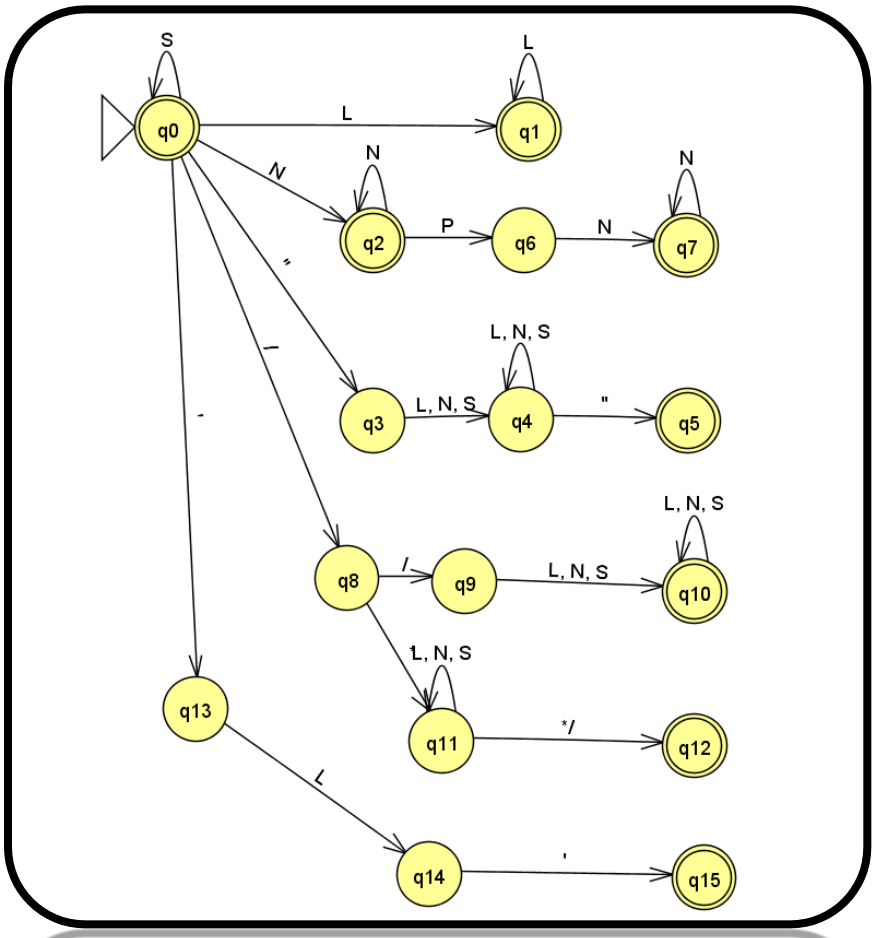
Conjuntos

$L = \{a, A, b, B, \dots, \dots, z, Z\}$

$S = \{Todos\ los\ Signos\}$

$N = \{0,1 \dots ,9\}$

AFD


S
L
q0
L
q1
N
N
N
q2
P
q6
N
q7
"
L, N, S
'
L, N, S
q3
L, N, S
q4
"
q5
-
L, N, S
q8
/
q9
L, N, S
q10
L, N, S
q13
*/
q11
q12
L
'
q14
q15

Gramática Libre del Contexto

<Inicio> ::= <Clase>

<Clase> ::= class identificador llave_izquierda <MetodoPrincipal> llave_derecha

< MetodoPrincipal > ::= static void main parentisis_izquierdo <ParametroMain> paréntesis_derecho llaveIzquierda <ListaDeclaracion> llaveDerecha

<ParametroMain> ::= string corchete_izquierdo corchete_derecho identificador

| ε

<ListaDeclaracion> ::= <InicioVariable>

| <If>

| <Switch>

| <While>

| <For>

| <ComentarioLinea>

| <ComentarioMultilinea>

| <InicioConsole>

| <AsignacionSinTipo>

| ε

<If> ::= if paréntesis_izquierdo <IdentificadorIf> <SimboloIf> <IdentificadorIf> parentesis_derecho llave_izquierda <ListaDeclaracion> llave_derecha <ElseIf>

<IdentificadorIf> ::= identificador

| digito

| ε

<SimboloIf> ::= igual igual

| mayor

| mayor igual

| menor

| menor igual

| admiración igual

<ElseIf> ::= else llave_izquierda <ListaDeclaracion> llave_derecha

| ε

<Switch> ::= switch paréntesis_izquierdo identificador parentesis_derecho llave_izquierda <CuerpoSwitch> llave_derecha

<CuerpoSwitch> ::= case Identificador dos_puntos <CuerpoCase> break punto_coma

| default dos_puntos <CuerpoCase> break punto_coma

<CuerpoCase> ::= <ListaDeclaracion>

<For> :: = for paréntesis_izquierdo <DeclaracionFor> punto_coma <ExpresionFor> punto_coma <IncrementoDecremento> parentesis_derecho llave_izquierda <ListaDeclaracion> llave_derecha

<DeclaracionFor> ::= int identificador igual digito

| identificador igual digito

<ExpresionFor> ::= <IdentificadorFor> <SimboloFor> <IdentificadorFor>

<IdentificadorFor> ::= identificador

| digito

| ε

<SimboloFor> ::= igual igual

| mayor

| mayor igual

| menor

| menor igual

| admiración igual

<While> :: = while paréntesis_izquierdo<IdentificadorWhile> <SimboloWhile> <IdentificadorWhile> parentesis_derecho llave_izquierda <ListaDeclaracion> llave_derecha

<IdentificadorWhile> ::= identificador

| digito

| ε

<SimboloWhile> ::= igual igual

| mayor

| mayor igual

| menor

| menor igual

| admiración igual

<ComentarioLinea> ::= ComentarioLinea <ListaDeclaracion>

<ComentarioMultilinea> ::= ComentarioMultilinea <ListaDeclaracion>

<InicioConsole> ::= console punto writeline paréntesis_izquierdo <Expresión> paréntesis_derecho punto_coma

<InicioVariable> ::= <DeclaracionVariable>

<DeclaracionVariable> ::= <Tipo> <ListaId> <OpcAsignacion> punto_y_coma

<Tipo> ::= int

| string

| char

| float

| bool

<ListaId> ::= Identificador <ListaId2>

<ListaId2> ::= coma Identificador <ListaId2>

| ε

<OpcAsignacion> ::= Signo_igual <valor>

| ε

<valor> ::= int

| string

| char

| float

| bool

<Expresion> ::= <Termino> <ExpresionPrima>

<ExpresionPrima> ::= Simbolo_mas

| Simbolo_menos

| ε

<Termino> ::= <Factor> <TerminoPrima>

<TerminoPrima> ::= Simbolo_por

| Simbolo_div

| ε

<Factor> ::= Parentesis_derecho <Expresion> Parentesis_derecho

| Digito

| cadena

| Identificado

<DeclaracionSinTipo> ::= Identificador Simbolo_igual <Expresion> Simbolo_puntoYcoma <ListaDeclaracion>