Universidad De San Carlos De Guatemala

Escuela de Ciencias y Sistemas

Lenguajes Formales de Programación

MANUAL TECNICO

ELI SAMUEL MAZARIEGOS RAMIREZ 201709426

Contenido

[Métodos utilizados en el programa: 3](#_Toc8679295)

[Clase Analisis\_Lexico: 3](#_Toc8679296)

[Clase Analisis\_Sintactico: 3](#_Toc8679297)

[Clase Reports: 4](#_Toc8679298)

[Método del árbol: 5](#_Toc8679299)

[Expresión regular utilizada para la resolución del proyecto: 5](#_Toc8679300)

[Árbol obtenido a partir de la expresión regular 5](#_Toc8679301)

[Tabla de siguientes 5](#_Toc8679302)

[Tabla de transiciones 6](#_Toc8679303)

[AFD 7](#_Toc8679304)

[GRAMATICA LIBRE DE CONTEXTO 8](#_Toc8679305)

## Métodos utilizados en el programa:

### Clase Analisis\_Lexico:

public void analizar(String texto) {

//este metodo es el encargado de realizar todo el analisis lexico, comprueba que los símbolos, caracteres etc, introducidos rean reconocidos por el lenguaje.

}

public int estado\_transicion(int numero){

//este metodo sirve para verificar cual es el character siguiente, retorna como tal un estado correspondiente al carácter encontrado.

}

public int palabra\_reservada(String lexema) {

//este metodo sirve para verificar si la cadena de aceptacion es una palabra reservada o simplemente una cadena, retorna como tal un numero de token.

}

public String[] separador(String texto, char separar) {

//este metodo tiene la function de split, es decir recibe dos parametros, una cadena y un carácter, el la cadena se ingresa el texto a separar y en el carácter se ingresa el char con el cual se va a separa la cadena de texto.

}

### Clase Analisis\_Sintactico:

public void etiquetas\_principales() {

//este es el metodo principal de esta clase, ya que aqui es donde se valida si viene una etiqueta principal y manda a llamar al método etiquetas\_secundarias.

}

public void etiquetas\_secundarias() {

//aquí es donde se valida si vienen etiquetas secundarias y asi mismo llama a cada método según su etiqueta.

}

public void capturar\_error(String esperado) {

//se captura el error sintáctico y recibe el carácter esperado.

}

public void etiqueta\_contenido() {

//tiene la función de hacer todas las validaciones sintácticas para la etiqueta contenido.

}

public void etiqueta\_img() {

//sirve para validar el patrón sintáctico para la inserción de imágenes.

}

public void etiqueta\_autor() {

//esta método tiene la función de reconocer la sintaxis de la etiquta autor.

}

public void etiqueta\_titulo() {

//esta método tiene la función de reconocer la sintaxis de la etiquta titulo.

}

public void consumir() {

//esta método tiene la función de incrementar la variable index en uno, este método es llamado cada vez que se encuentra alguna sintaxis reconocida.

}

### Clase Reports:

ublic void crear\_reporte(){

//este método crea el reporte del análisis realizado, adjuntando en el lo que son los errores léxicos, errores sintácticos y la tabla de tokens reconocidos.

}

## Método del árbol:

### Expresión regular utilizada para la resolución del proyecto:

**ER [(A-Z) | (a-z)] . [(A-Z) |(a-z) |(0-9)]\* | [(0-9) . (0-9)\*] |[<] | [>] | [‘] | [/] | [.] | |[,] | [!]**

### C:\Users\Samuel\Downloads\Untitled Diagram (10).pngÁrbol obtenido a partir de la expresión regular

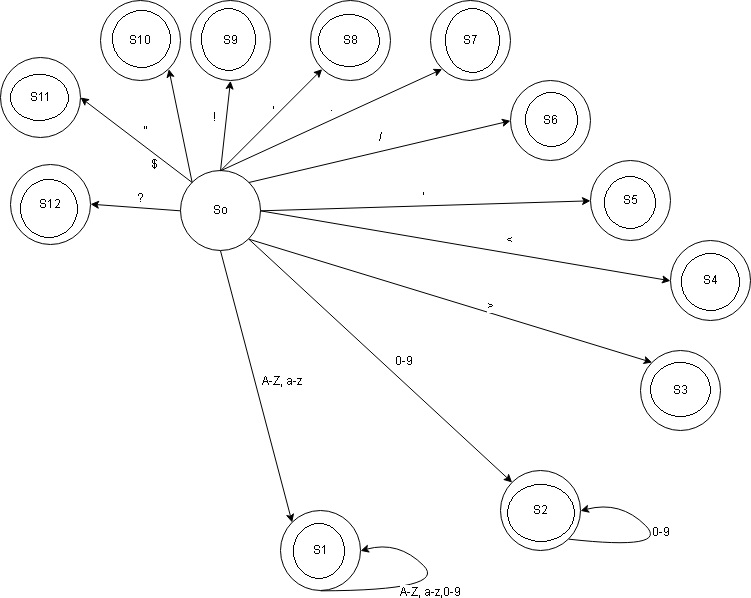
### Tabla de siguientes

|  |  |
| --- | --- |
| HOJA | Follow post |
| 1 | 15 |
| 2 | 15 |
| 3 | 15 |
| 4 | 15 |
| 5 | 15 |
| 6 | 15 |
| 7 | 15 |
| 8 | 8,9,15 |
| 9 | 8,9,15 |
| 10 | 10,11,12,13,14,15 |
| 11 | 10,11,12,13,14,15 |
| 12 | 10,11,12,13,14,15 |
| 13 | 10,11,12,13,14,15 |
| 14 | 10,11,12,13,14,15 |
| 15 | - |

### Tabla de transiciones

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ESTADO | ! | , | . | / | ' | > | < | 0-9 | A-Z | a-z |
| So = {1} | A = 15 |  |  |  |  |  |  |  |  |  |
| So = {2} |  | B = 15 |  |  |  |  |  |  |  |  |
| So = {3} |  |  | C = 15 |  |  |  |  |  |  |  |
| So = {4} |  |  |  | D = 15 |  |  |  |  |  |  |
| So = {5} |  |  |  |  | E = 15 |  |  |  |  |  |
| So = {6} |  |  |  |  |  | F = 15 |  |  |  |  |
| So = {7} |  |  |  |  |  |  | G = 15 |  |  |  |
| So = {8} |  |  |  |  |  |  |  | H = 8,9,15 |  |  |
| H = {8} |  |  |  |  |  |  |  | H |  |  |
| So = {9} |  |  |  |  |  |  |  | H |  |  |
| H = {9} |  |  |  |  |  |  |  | H |  |  |
| So = {10} |  |  |  |  |  |  |  |  | I =  10,11,12,13,14,15 |  |
| I ={10,..14} |  |  |  |  |  |  |  | I | I | I |
| So = {11} |  |  |  |  |  |  |  |  |  | I |

### AFD



## GRAMATICA LIBRE DE CONTEXTO

|  |  |
| --- | --- |
| **So** **→** | **T T** |
| **T T →** | T T  **T** |
| **T** **→** | <ID>**INNER**</ID> |
|  | **|** <ID>**TT**</ID> |
| **INNER** **→** | INNEN **IN** |
|  | **|** IN |
| **IN**  **→** | ID | ! | ' | NUM |