Universidad Rafael Landivar
Facultad de Ingeniería
Ingeniería Informática y en Sistemas
Lenguajes Formales y automatas
Ing. Moises Alonso

PROYECTO
EXPRESIONES REGUALRES
FASE 1

César Silva - 1184519
Mariandre Gómez Espino – 1000119
Julio Anthony Engels Ruiz Coto – 1284719
Eddie Alejandro Girón Carranza – 1307419

Guatemala 17 de febrero del 2024

INTRODUCCION

El proyecto aborda el análisis de los lenguajes formales, centrándose en la fase compilación. El objetivo principal es el uso de la función de analizadores léxicos. El programa resultante deberá ser capaz de reconocer un lenguaje y evaluar si las palabras utilizadas cumplen con la gramática establecida.

El proyecto se divide en tres fases, donde la finalización exitosa de cada fase es crucial para avanzar a la siguiente. Comenzamos con la lectura de un archivo llamado "GRAMATICA.txt", que contiene la definición de la gramática, dividiéndose en secciones como "SETS", "TOKENS", "ACTIONS" y "ERROR".

La sección "SETS" define conjuntos de símbolos terminals. En "TOKENS", se definen los símbolos terminales y no terminales a través de expresiones regulares. La sección "ACTIONS" presenta funciones

Además, se incluye una sección "ERROR" para gestionar posibles errores.

EXPRESIONES REGULARES

EXPRESION	SIGNIFICADO
^	Inicio de linea
*	Cero o mas
=	Representa igualdad
[\s\t]	Espacios en blanco
[0-9]	Cualquier digito entre ese rango
?	Digito opcional
$	Finalizar cadena
|	Una opcion o otra
+	Una o mas veces
d+	Uno o mas digitos
\t	Caracter de tabulacion
\n	Nueva linea
\r	Caracter de retorno

- `var patternError = @"^[\s\t]*ERROR[\s\t]*=[\s\t]*[0-9][0-9]?$";`

Expresión regular que se utiliza para buscar líneas que contengan la palabra “ERROR”

- `@"^TOKENS|TOK2N|TOKE|TOK|TO|T(\s|\t)*$"`

Expresión regular que valida que la línea comience con cualquiera de esas opciones seguido por espacios en blanco opcionales

- `@"^(SET|TOKE|ACTION|SE|T|TO|TOK|A|AC|ACT|ACTI|ACTIO)(\s|\t)*$"`

Expresión regular que valida los títulos y si tienen algún espacio

- `var pattern = @"^SETS(\s\t)*$";`

Expresión regular que valida el titulo de los SETS

- `var pattern2 = @"^(\s\t)*([a-zA-Z_]+)(\s|\t)*=(\s|\t)*((('[a-zA-Z]')((\s|\t)*\.\.(\s|\t)*('[a-zA-Z]'))?)|('[_]'))?((\s|\t)*\+(\s|\t)*((('[a-zA-Z]')((\s|\t)*\.\.(\s|\t)*('[a-zA-Z]'))?)|('[_]')))*(\s|\t)*$";`

expresión que valida la parte de “LETRAS” del SETS

- `var pattern3 = @"^(\s\t)*([a-zA-Z_]+)(\s|\t)*=(\s|\t)*(('[0-9]')((\s|\t)*\.\.(\s|\t)*('[0-9]')))$";`

Expresión regular que valida la parte de DIGITOS DE SETS

- `var pattern4 = @"^([a-zA-Z_]+)\s*=\s*\s*CHR\(\d+\)\s*\.\.\s*CHR\(\d+\)\s*$";`

expresión que valida la parte de CHARSETS de SETS

- `var pattern5 = @"^(\s\t)*$";`

expresión regular que valida una línea sin caracteres

- `var patternTitulo = @"^TOKENS(\s\t)*$";`

expresión regular que valida el titulo TOKENS

- `var pattern4 = @"^TOKEN\s+\d+\s*=\s*('[^']*'|[+\-*?()|]+)\s*$";`
- `var pattern5 = @"^TOKEN\s+\d+\s*=\s*'([A-Za-z0-9]+|[+\-*?()|]+)'\s*$";`
- `var pattern6 = @"^TOKEN\s+\d+\s*=\s*('.*?'|[+\-*?()|]+)\s*$";`
- `var pattern7 = @"^TOKEN\s+\d+\s*=\s*DIGITO\s+DIGITO\s+\*\s*$"; // Ajustado para incluir el caso DIGITO DIGITO *`
- `var pattern8 = @"^TOKEN\s+\d+\s*=\s*LETRA(\s*\(\s*LETRA\s*\|\s*DIGITO\s*\)\s*\*)\s*\{\s*RESERVADAS\(\)\s*\}\s*$";//para el token 3`
- `var pattern9 = @"^(\s|\t|\r|\n)*$";`

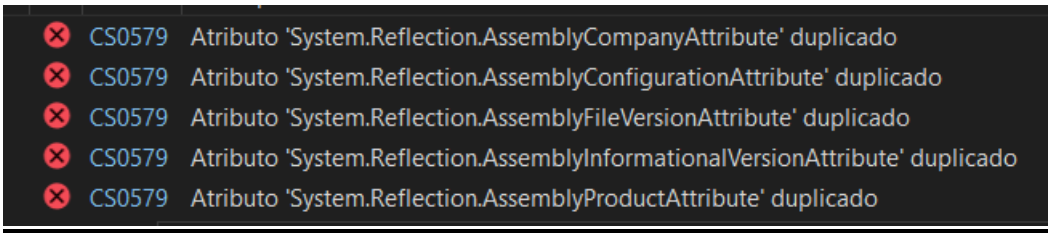
Expresiones regulares que valida las líneas de tokens, verifica (+-/*^), que comience con numero de dos dígitos o más y otros símbolos.

- `var pattern = @"^ACTIONS(\s\t)*$";`
- `var pattern2 = @"^RESERVADAS\(\)(\s\t)*$";`
- `var pattern3 = @"^(\s\t)*{$";`
- `var pattern4 = @"^(\s\t)*[0-9]+(\s|\t)*=(\s|\t)*('[a-zA-Z]+')(\s|\t)*$";`
- `var pattern5 = @"^}";`
- `var pattern6 = @"^(\s\t)*$";`

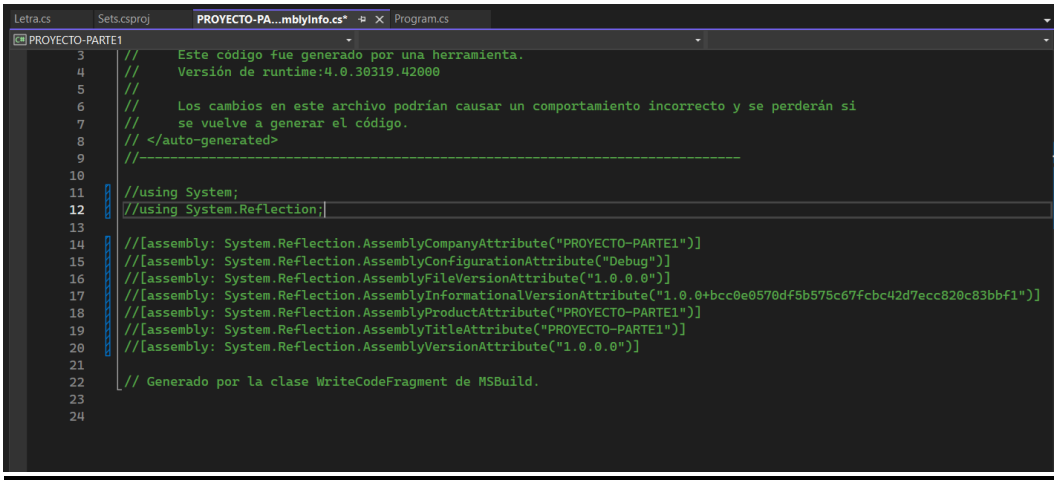
Expresiones regulares que valida el estado ACTIONS, que comiencen con la palabra reservadas, los espacios en blanco y la cantidad de dígitos.

¿CÓMO EJECUTAR EL PROGRAMA?

- Al ejecutar por primera vez el programa , aparecerá este error por lo que hay que seleccionarlo para que lleve al lugar indicado

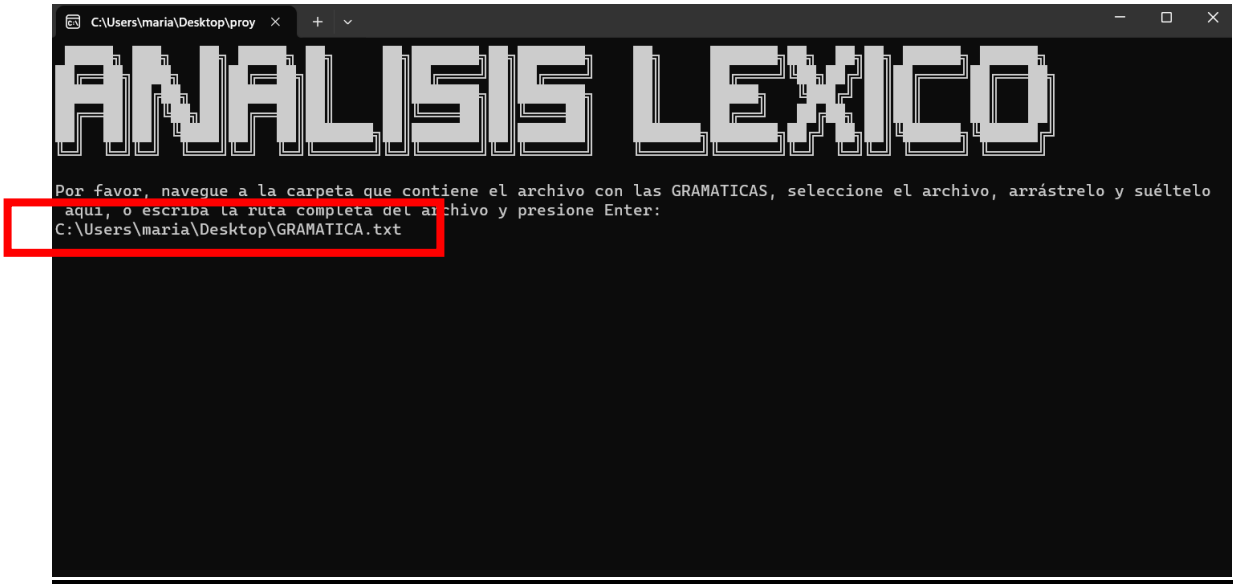


- Al seleccionar el error, tira a esta ventana



por lo que hay que comentar todas las líneas y ya se ejecutara completamente el programa

- Igualmente, al ingresar el txt al programa, este deberá de estar en el escritorio para que a la hora de arrastrarlo al se pueda leer correctamente



GIT PROYECTO(RAMA FINAL): https://github.com/engelsruiz09/LENGUAJE-FORMALES-Y-AUTOMATAS/tree/FINAL

LINK VIDEO: https://youtu.be/TBVW74mcleI

LINK VIDEO ¿COMO EJECUTAR EL PROGRAMA?: https://youtu.be/E2rAcx_BoiI