**PROYECTO PRÁCTICO FASE III**

**GENERADOR DE SCANNER**

**Objetivo**

En la Fase III del proyecto es pre-requisito haber entregado completamente la Fase I y la Fase II culminando con la tabla de transiciones, ya que en esta fase, el entregable el generador de scanner completo.

¿Qué es un generador de scanner?

Un scanner o analizador léxico es según Aho: “*La primera fase de un compilador. Su principal función consiste en leer los caracteres de entrada y elaborar como salida una secuencia de componentes léxicos que utiliza el analizador sintáctico para hacer el análisis*”[AHO].

De acuerdo a lo anterior nuestro programa es un generador de estos analizadores léxicos ya que a través de las expresiones regulares obtenidas en la sección “tokens” del archivo será capaz de generar programas que reconozcan un determinado lenguaje regular.

**Explicación de la salida**

En la tercera fase del proyecto es necesario la lectura de un archivo de texto llamado: GRAMATICA.txt el cual contiene la definición de la gramática.



Al determinar que los resultados de la fase I y fase II son correctos entonces se generará un programa en **java** el cual al ser compilado será capaz de reconocer un lenguaje regular (Definido por las expresiones regulares de la sección tokens).

**SET DE PRUEBAS**

Tenemos el siguiente archivo de prueba:

SETS

LETRA='A'..'Z'+'a'..'z'+'\_'

DIGITO='0'..'9'

TOKENS

TOKEN 1=DIGITO DIGITO \*

TOKEN 2='='

TOKEN 3=':''='

TOKEN 4=LETRA(LETRA|DIGITO)\* {RESERVADAS()}

ACTIONS

RESERVADAS()

{

5='PROGRAM'

6='INCLUDE'

7='CONST'

8='TYPE'

}

ERROR=9

Esto generará un **java** que al ser compilado y generar un programa ejecutable que sea capaz de reconocer la siguiente línea de código:

**Program x a:=b c=d const a**

Dando como resultado lo siguiente:

**Program = 5**

**x = 4**

**a = 4**

**:= = 3**

**b = 4**

**c = 4**

**= = 2**

**Const = 7**

**a = 4**

**Entregables Fase III**

El programa debe codificarse en el lenguaje **C#** Utilizando Windows Forms o consola, versión de IDE recomendado es Visual Studio 2019 aunque se puede utilizar cualquiera de sus versiones anteriores.

El programa que se genera es recomendable que sea en consola y el lenguaje a su elección.

Entregables:

* Programa fuente, debidamente documentado internamente
* Programa ejecutable
* Set de casos de prueba utilizados con sus respectivos archivos y resultados esperados.
* Documentación de análisis que contenga las expresiones regulares y árboles de expresiones creados para parte del archivo.
* Generación de scanners los cuales serán generados de acuerdo a las expresiones regulares de los tokens.

FECHA DE ENTREGA:

Calificación virtual e individual el día: **miércoles 3 de mayo de 2023** en horario de clase, el código fuente a evaluar será el que esté subido en la sección de entrega del portal.

**Aspectos a evaluar**

* Validación de errores, uso de **procedimientos, funciones, clases, herencia, estructuras de datos** en todas las operaciones.
* Adecuada aplicación de los conocimientos vistos en clase
* Calidad de la documentación: ortografía, orden, limpieza y que esté completa.
* Calidad de la solución propuesta: que solucione el problema (que haga lo que requiere el enunciado) en forma eficaz.
* Funcionalidad del programa: debe cumplir a cabalidad con todos los requerimientos.
* Evidencia de la creación del programa y dominio de los conceptos utilizados.
* Uso de herramientas para versiones de código, específicamente github utilizando la cuenta de correo universitaria, de acuerdo a la cantidad y calidad de los “commit” será el porcentaje de su nota en el proyecto
* Creatividad.

**Baremo de calificación**

|  |  |  |  |
| --- | --- | --- | --- |
| ITEM | Resultado | Ponderación | Observaciones |
| Solución entregada en el portal | Sí / No | -- | Si no está en el portal no se califican los demás ítems y se obtiene 0 puntos |
| Solución compila | Sí / No | -- | Si la solución no compila se obtiene 0 puntos |
| Lectura de archivos correcta | Sí / No | 10 puntos | Lee archivos de forma correcta con filtros y no se produce ninguna excepción durante la lectura de archivos. |
| Generación de código que reconozca lenguaje | Sí / No | 15 puntos por cada archivo de prueba, total: 45 puntos | Genera los resultados esperados de cada archivo de prueba (El catedrático pondrá los archivos de prueba) |
| El programa reconoce el lenguaje adecuadamente. | Sí / No | 10 puntos por cada prueba, total: 30 puntos. | No utiliza librerías de evaluación de expresiones regulares ni herramientas de análisis lexicográfico |
| Documentación técnica con expresiones regulares y árboles de expresión | Sí / No | 15 puntos | Presenta documento que muestra razonamiento utilizado. |