*Instituto Tecnológico de Costa Rica*

*Sede Regional San Carlos*

*Escuela Ingeniería en Computación*

*Compiladores e Intérpretes*

***Tarea Programada #1***

***“Mini - C#”***

*Responsable:*

*Hellen Rojas Rojas 2013083934*

*Henry Solís Chacón    2013085706*

*6 de abril del 2016*

*Santa Clara, San Carlos*

**Análisis del lenguaje**

Según la gramática solicitada y los extras (Foreach) se logra hacer todo.

**Soluciones e implementación**

Para la implementación del proyecto se trabajó la lógica del compilador sobre dos archivos Lexer o Parser, donde el primero contiene la gramática que va a evaluar el Parser, estos dos archivos comprueban la correcta escritura o sintaxis del código recibido.

Una vez que se comprueba y se logra compilar se genera un AST que es recorrido e impreso por un archivo llamado Pretty Print el cual utiliza el componente TreeView para mostrarlo.

En cuanto al editor de texto el mismo fue encontrado en internet, este permite usar pestañas para observar el código, crear y guardar nuevos archivos de texto o para código fuente en este caso.

También contiene la vista para el árbol AST.

**Resultados obtenidos**

|  |  |  |
| --- | --- | --- |
| **Tarea** | **Estado** | **Observaciones** |
| Generar Lexer | Completo |  |
| Generar Parser | Completo |  |
| Agregar Editor | Completo |  |
| Agregar Foreache | Completo |  |
| Generar AST | Completo |  |
| Errores en Español | Completo |  |

**Conclusiones**

El trabajo asignado es una buena manera de entender el funcionamiento de un compilador, dado que muchas veces en los cursos la teoría simplemente se da pero nunca se logra tener una idea práctica o clara del funcionamiento real, es interesante observar la preparación de sus reglas a evaluar hasta su generación del AST.

La herramienta ANTRL4, es algo compleja de entender por su extraña sintaxis y por los varios problemas que presentar al intentar aplicar mejoras o cambios a una sección, sin embargo su facilidad de ayuda para generar el compilador es muy buena.

Una de las mayores dificultades es lograr obtener un parser capaz de reconocer todos los errores del código, para evitar fallos en la futura sección de evaluación contextual.

# Bibliografía

ANTRL4. (05 de 04 de 2016). Obtenido de http://www.antlr.org/api/JavaTool/org/antlr/v4/runtime/DefaultErrorStrategy.html#endErrorCondition(org.antlr.v4.runtime.Parser)

Microsoft Development Network. (05 de 04 de 2016). *Microsoft Development Network*. Obtenido de https://msdn.microsoft.com/es-es/library/aa664754(v=vs.71).aspx

Universidad de Cordoba. (05 de 04 de 2016). *UCO*. Obtenido de http://www.uco.es/users/ma1fegan/Comunes/manuales/pl/ANTLR/Analisis-lexico-con-ANTLR.pdf