Compilador

Primera entrega

Integrantes

-Barrera Rangel Guillermo

-Fernández Mora José Enrique

-Guerrero López Enrique

-López Soto Miguel Ángel

-Luna Fierros Leonardo

Introducción

Un compilador es un programa que transforma el código fuente escrito por el desarrollador en un lenguaje de programación de alto nivel en su expresión equivalente en lenguaje máquina, que puede ser interpretado por el procesador. El proceso de convertir el código fuente a su representación en código de objeto es conoció como compilación.

Un compilador ejecuta cuatro funciones principales:

Scaning: El escáner lee un elemento a la vez del código fuente y mantiene presente que elementos existen por línea de código.

Análisis Léxico: El compilador convierte la secuencia de elementos o caracteres que aparecen en el código fuente a una serie de cadenas de caracteres conocidos como “tokens” que son asociados a un a regla especifica por el programa llamada analizador léxico. Una tabla de símbolos o estructura de datos análoga es usada por el analizador léxico para guardar las cadenas del código fuente que corresponden con el token generado.

Análisis Sintáctico: En este paso, el análisis de la sintaxis es realizado, esto conlleva reprocesar la información capturada para verificar si los tokens creados durante el análisis léxico están propiamente ordenados para su uso. El orden correcto de un conjunto de palabras clave o “keywords” para obtener el resultado deseado se conoce como sintaxis. El compilador tiene que examinar que el código fuente cumpla con la sintaxis esperada.

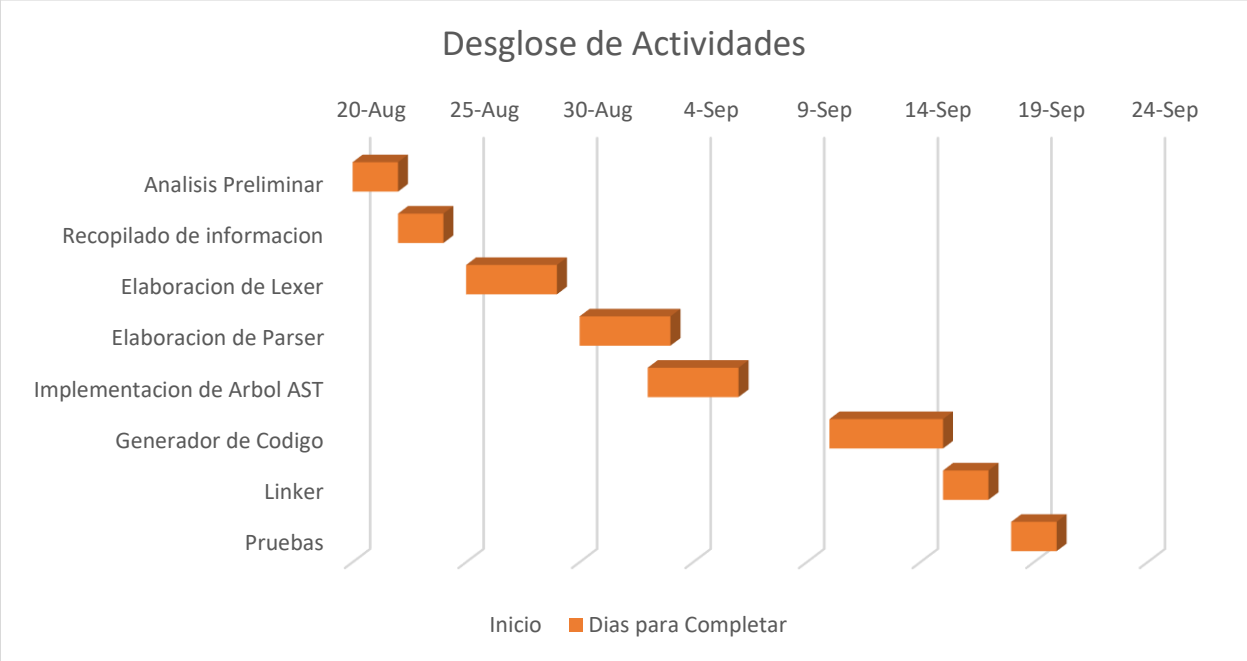
Análisis Semántico: Este paso esta compuesto de varios procesos intermedios. Primero la estructura de tokens es verificada, así como el orden de estos de acuerdo con la gramática especifica de cada lenguaje. El significado de la estructura de tokens es interpretado por el parser y el analizador para finalmente generar código intermedio, llamado también como código de objeto. El código de objeto incluye instrucciones que representan comandos interpretables por el procesador.

Plan de proyecto

La elaboración del proyecto fue coordinada de manera exitosa a lo largo de 25 días, plazo necesario para cumplir con todas las tareas necesarias para cumplir con los requerimientos planteados. Se sostuvieron sesiones semanales en las que se discutió el avance de la elaboración de los entregables, además cada integrante de equipo compartió el esquema de actividades que seguiría en los días posteriores.

La fase de análisis preliminar y recopilado de información corresponden a las actividades necesarias para familiarizar a los integrantes con conceptos fundamentales de funcionamiento de compiladores, programación funcional, y sintaxis básica del lenguaje de programación Elixir.

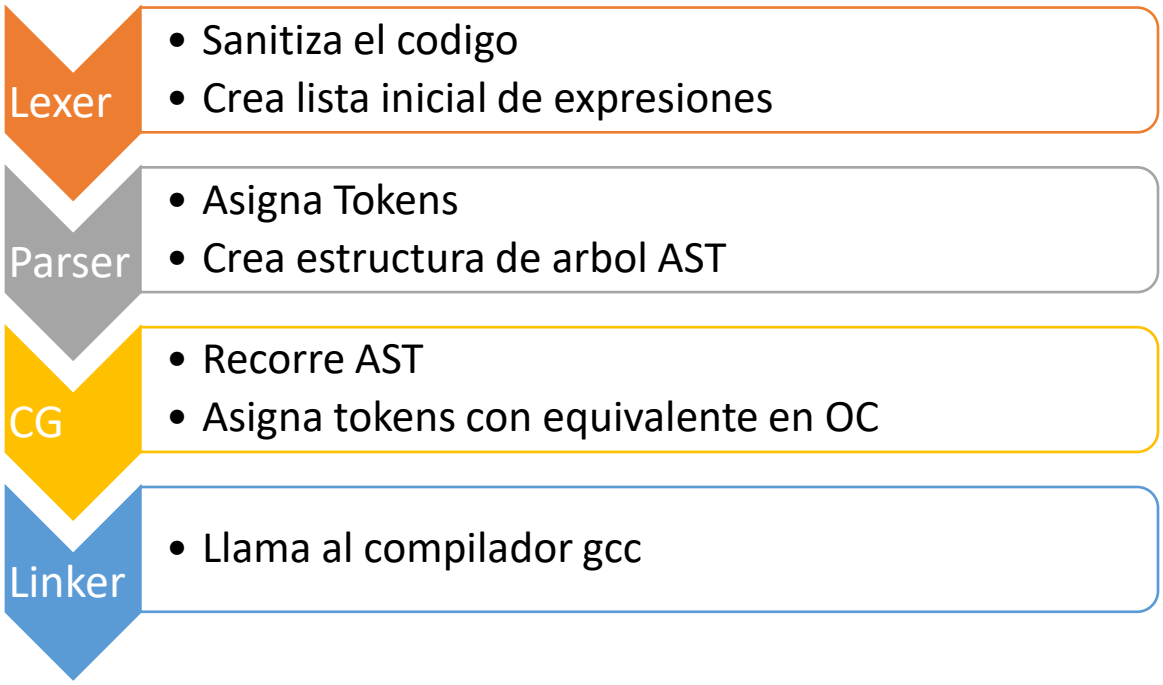
Las secciones posteriores en el diagrama de Gantt obedecen a los pasos y metodologías expresadas en el tutorial de la programadora Nora Sandler.



Arquitectura

La estructura general del compilador esta fundamentada en las recomendaciones del tutorial de Nora Sandler. Por lo anterior el compilador cuenta con cuatro módulos fundamentales, estos son: Lexer, Parser, Code Generator, Linker. En esta fase el compilador no realiza tareas adicionales como optimización de código.

Los módulos anteriormente mencionados procesan el código de manera secuencial como se ve en el siguiente diagrama.



La estructura del proyecto tiene tres secciones principales, con la estructura que se presenta a continuación:

- _build
  - .mix
  - Consolidated
  - ebin
- lib
  - CodeGenerator
  - Nodo
  - Token
  - Compiler
  - Lexer
  - Linker
  - parser
- test
  - compiler_test.exs
  - test_helper.exs

Plan de pruebas y test suite

Para la realización de las pruebas se utilizó el código provisto por Nora Sandler para el testeo del proyecto, dichas pruebas se realizaron de maneara automática y se verifico que el compilador se comportara de la forma esperada para casos en donde la sintaxis y gramática del programa fueran correctas y casos en donde esta condición no se satisficiere.

Conclusiones

La elaboración del compilador probo ser un reto mas grande de lo que todos los miembros del equipo anticipábamos, no solo fue necesario familiarizarnos por primera vez con paradigmas como la programación funcional sino también adaptarnos al uso de un nuevo lenguaje de programación, así como retomar conceptos relacionados con ciencia de la computación que en ciertos casos no aplicábamos desde hace varios meses.

Otro reto importante que encontramos a lo largo del desarrollo del proyecto fue el de alcanzar la forma más eficiente de coordinar y organizar nuestros esfuerzos, en más de una ocasión la falta de comunicación oportuna provoco retrasos importantes para alcanzar los hitos planeados originalmente. La falta de comunicación y una adecuada planeación generaron también en algunas ocasiones un ambiente de tensión dentro de la dinámica del grupo.

Es por lo anterior que para promover un ambiente mas eficiente y de armonía hemos planteado las siguientes medidas que serán puestas en marcha para la siguiente iteración del proyecto:

- Plantear plazos realistas para elaboración de actividades
- Llevar una mejor coordinación y comunicación de avances
- Promover el trabajo en parejas
- Establecer como fecha limite de entrega una semana anterior a la oficial

Apéndice

https://github.com/hiphoox/c201-whiletrue