Minor C

Manual Técnico

Herbert Rafael Reyes Portillo

201612114

Indice

Contenido

Introducción

MinorC es un subconjunto del lenguaje C, creado con el fin de poner en práctica los conceptos del proceso de compilación, para cubrir las competencias del curso. Minor C es capaz de realzar las operaciones básicas de C para luego poder ser traducido al lenguaje Augus.

Descripción de la Aplicación

Lenguaje Utilizado

Python: es un lenguaje de programación interpretado cuya filosofía hace hincapié en la legibilidad de su código.2 Se trata de un lenguaje de programación multiparadigma, ya que soporta orientación a objetos, programación imperativa y, en menor medida, programación funcional. Es un lenguaje interpretado, dinámico y multiplataforma.

IDE Utilizado:

PyCharm: es un entorno de desarrollo integrado (IDE) utilizado en la programación de computadoras, específicamente para el lenguaje Python . Está desarrollado por la empresa checa JetBrains . Proporciona análisis de código, un depurador gráfico, un probador de unidad integrado, integración con sistemas de control de versiones (VCS), y soporta el desarrollo web con Django y Data Science con Anaconda.

Descripción de Arit Software:

Lenguaje: Augus

Analizadores:

Ply -> Ascendente

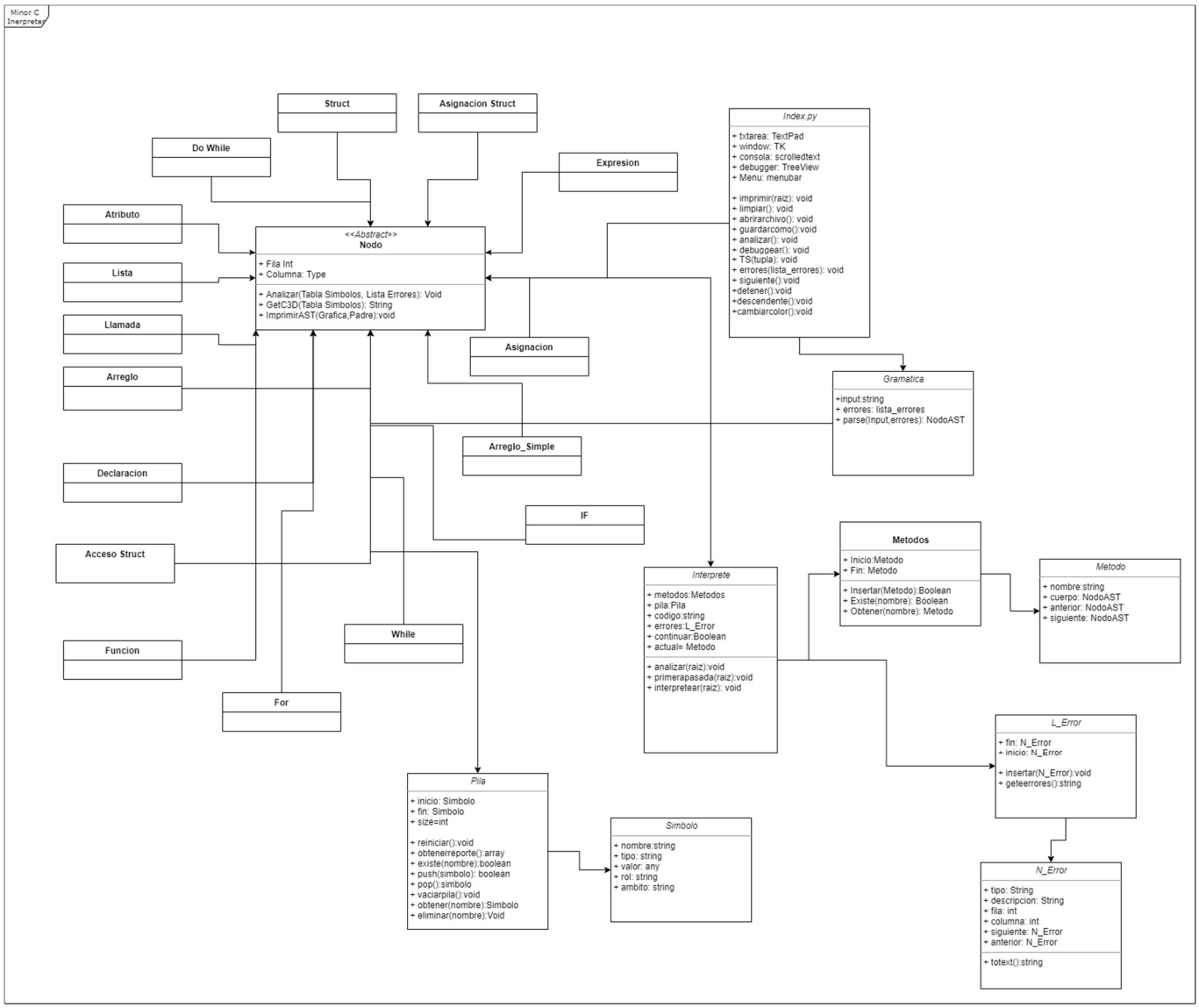
Fuentes de desarrollo:

1. Sistema Operativo Windows 10 (64bits)
2. PyCharm
3. Ply
4. Tkinter
5. Graphviz
6. Libreria MagicStick_Editor
7. Disco Duro de 512 GB
8. Procesador Intel Core i3, 4.00GH

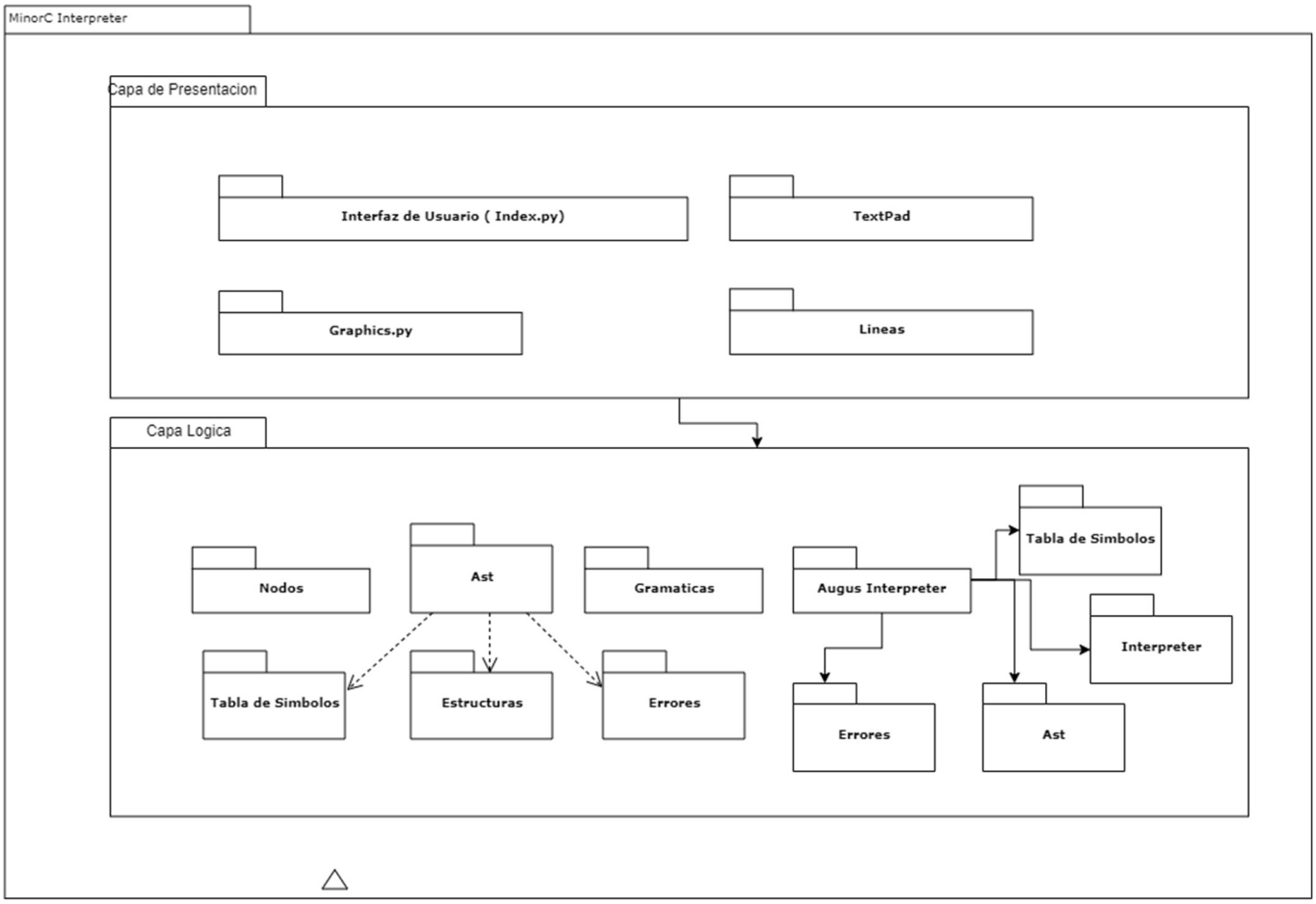
Requerimientos del usuario:

1. 55 MB libres de memoria en el disco duro
2. 1 GB de memoria RAM
3. Python 3.8

Diagrama de Clases



Diagramas de paquetes



Herramientas Utilizadas

PLY: es una implementación de herramientas de análisis lex y yacc para Python. Entre sus características esenciales tenemos:

- Está implementado completamente en Python.
- Utiliza el análisis LR que es razonablemente eficiente y adecuado para gramáticas más grandes.
- PLY proporciona la mayoría de las características estándar de lex / yacc, que incluyen soporte para producciones vacías, reglas de precedencia, recuperación de errores y soporte para gramáticas ambiguas.
- PLY es fácil de usar y proporciona *una* amplia verificación de errores.
- PLY no intenta hacer nada más o menos que proporcionar la funcionalidad básica de lex / yacc. En otras palabras, no es un marco de análisis grande o un componente de un sistema más grande.

MagicStick_Editor: Un editor de texto, del cual se utilizo el código para poder realizar el numero de líneas en la interfaz. Para mas información: https://github.com/surajsinghbisht054/MagicStick_Editor?fbclid=IwAR1mU6NeX1kJTLs96QjYYxqfD9zuCKDBDouRbSuk4NRp9cO9ZFVyG3T4MJc

Tkinter: es la interfaz estándar de Python para el kit de herramientas Tk GUI. Tanto Tk como tkinter están disponibles en la mayoría de las plataformas Unix, así como en los sistemas Windows. (Tk en sí no es parte de Python; se mantiene en ActiveState)

Explicación De Acciones Semanticas

Declaraciones: Para una declaración se verifica si la variable que se intenta declarar existe, de ser asi se muestra un error de que la variable ya existe, por otro lado, si la variable no existe se procede a procesar la expresión y si esta no contiene ningún error es asignada a la variable y la misma guardada en la tabla de símbolos, por otro lado, si la expresión contiene errores se reportara un error.

Retornar: La sentencia retornar que puede contener una expresión, se procesa la expresión y si esta contiene errores se reporta un error, si sucede lo contrario el valor de la expresión es devuelto para poder ser utilizado en una posible asignación, u otra operación que sea necesario el uso de una llamada a una función que retorne un valor.

Funciones: Cuando se encuentra una función, se verifica si el nombre de esta misma ya existe y de ser asi se retorna un error, por el lado contrario si no existe se procesan sus parámetros y sentencias, luego de ellos se añade la función a la tabla de símbolos para un posible posterior uso.

Sentencias de control: Al encontrar una sentencia de control se procede a procesar la expresión la cual se evaluara para ver si cumple una condición para poder ejecutar las sentencias asociadas al bloque, si la expresión contiene errores se reporta un error y por otro lado si no se posee errores se crea un nuevo entorno para el manejo de las variables (lo cual significa que las variables declaradas dentro de una sentencia de control no existirán fuera de estas mismas). Además de esto se utiliza una estructura de datos llamada display la cual permite reportar saber si se encuentra dentro de una sentencia de datos o dentro de una función.

Funciones propias de arreglos: para los arreglos existen funciones propias, para lo cual al mandarlas a llamar se verifica si la variable efectivamente es un arreglo, de ser así se continua y se verifica si el arreglo es homogéneo (ya que en la mayoría de las funciones propias de los arreglos es necesario que estos sean homogéneos), luego de ellos se realiza las operaciones correspondientes y se manda a llamar a la función asociada como parámetro según sea el caso. De existir algún error este es reportado.

Imprimir: Esta sentencia permite imprimir valores en consola, para lo cual se evalúa una expresión y se imprime su valor, de contener errores dicha expresión se despliega un error.