PROYECTO NO.2 SEGUNDO SEMESTRE 2022

201700747 – Dennis Alexander Gamboa Stokes

Resumen

La Programación Orientada (POO) a objetos permite que el código sea reutilizable, organizado y fácil de mantener. Sigue el principio de desarrollo de software utilizado por muchos programadores DRY (Don’t Repeat Yourself), para evitar duplicar el código y crear de esta manera programas eficientes mientras Graphviz es una herramienta utilizada por muchas personas que permite la creación de graficas usando un lenguaje especifico.

Compilador:

Es un programa informático que traduce todo el código fuente de un proyecto de software a código máquina antes de ejecutarlo. Solo entonces el procesador ejecuta el software, obteniendo todas las instrucciones en código máquina antes de comenzar

Se investigo el y se elaboro un compilador con la ayuda de JFLEX y JCUP para crear un traductor de Pseudocódigo a Python y Golang

The software development principle used by many programmers DRY (Don't Repeat Yourself), to avoid duplicating the code and thus create efficient programs while Graphviz is a tool used by many people that allows the creation of graphs using a language specific.

Compiler:

It is a computer program that translates all the source code of a software project into machine code before running it. Only then does the processor execute the software, fetching all instructions in machine code before starting

The was investigated and a compiler was developed with the help of JFLEX and JCUP to create a translator of

Pseudocode to Python and Golang

Introducción

En el Presente trabajo o programa se mostrará la una ventana donde se introducirá Pseudocódigo para poder traducido y ver si tiene errores léxico o sintáctico con el compilador que se creara con la librerías Jflex y Jcup, Los errores se recuperan para poder ser mostrados.

JFLEX:

JFlex es un generador de analizadores léxicos para Java escrito en Java. También es una reescritura de la herramienta JLex (Berk 1996) que fue desarrollada por Elliot Berk en la Universidad de Princeton. Como dice Vern Paxson para su herramienta flexible C/C++ (Paxson 1995): sin embargo, no comparten ningún código.

Un generador de analizador léxico toma como entrada una especificación con un conjunto de expresiones regulares y acciones correspondientes. Genera un programa (un lexer) que lee la entrada, compara la entrada con las expresiones regulares en el archivo de especificaciones y ejecuta la acción correspondiente si coincide una expresión regular. Los Lexers suelen ser el primer paso inicial en los compiladores, ya que combinan palabras clave, comentarios, operadores, etc., y generan un flujo de token de entrada para los analizadores. También se pueden utilizar para muchos otros fines.

JCUP

Este manual describe el funcionamiento básico y el uso del Constructor de analizadores útiles (CUP, por sus siglas en inglés) basado en Java(tm). CUP es un sistema para generar analizadores LALR a partir de especificaciones simples. Cumple la misma función que el ampliamente utilizado programa YACC [1] y, de hecho, ofrece la mayoría de las características de YACC. Sin embargo, CUP está escrito en Java, usa especificaciones que incluyen código Java incrustado y produce analizadores que se implementan en Java.

Aunque este manual cubre todos los aspectos del sistema CUP, es relativamente breve y se supone que tiene al menos un poco de conocimiento del análisis de LR. Un conocimiento práctico de YACC también es muy útil para comprender cómo funcionan las especificaciones de CUP. Varios libros de texto de construcción de compiladores (como [2,3]) cubren este material y analizan el sistema YACC (que es bastante similar a este) como un ejemplo específico.

.

Estructura de Programación:

Estructura Condicional:

Permite alterar la secuencia normal de pasos en un paso específico del Algoritmo, para crear 2 alternativas de bloques de ejecución, de manera excluyente entre ambos. En otras palabras: Solo uno de los 2 bloques se ejecutará, nunca ambos bloques y permite decidir por cuál alternativa seguirá el flujo del programa dependiendo del resultado de la evaluación de una condición. Para establecer condiciones complejas se utilizan los operadores relacionales y lógicos

Estructura cíclica:

Se llaman problemas repetitivos o cíclicos a aquellos en cuya solución es necesario utilizar un mismo conjunto de acciones que se puedan ejecutar una cantidad específica de veces. Esta cantidad puede ser fija (previamente determinada por el programador) o puede ser variable.

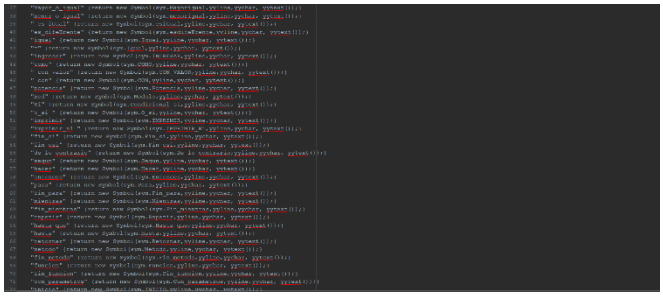
Estructura Secuencial:

La programación secuencial es más simple y fácil de usar. Como las instrucciones están relacionadas, será más sencillo entender lo que hace cada función en una instrucción. Las tareas se

llevan a cabo de tal manera que la salida de una es la entrada de la siguiente y así sucesivamente hasta finalizar un proceso; por esta razón se le conoce como secuencial.

Métodos Utilizados:

Creación de Archivo Léxico



LEXICO.JFLEX

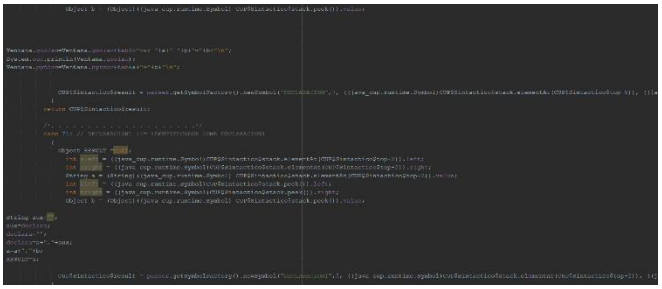
Resultado Del Archivo léxico que será el léxico.java aquí se muestra todos los token que son aceptados además que se captan los errores

LEXICO.JAVA

Se procede a crear la gramática para el análisis sintáctico con Cup además de que se pone variables externas para poder obtener la traducción mientras se hace el análisis sintactico

Sintactico.cup

Se produce el sintactico.java con las instrucción

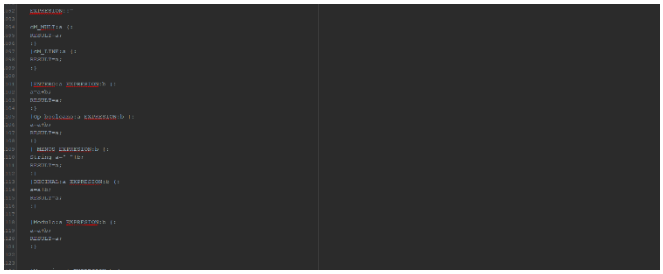


Estado Inicio aquí se introduce el incio y fin y entre ellos iran las Setencias

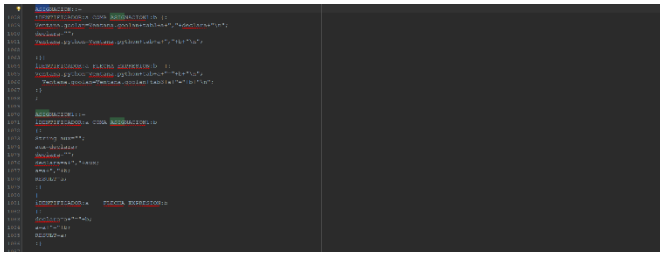
ESTADO SETENCIA AQUÍ VIENEN TODAS LAS INTRODUCCION

ESTADO EXPRESION DONDE VIENEN LA MAYORIA DE EXPRESIONES Y VARIABLES

ESTADO DECLARACION QUE SE USA PARA LA ESTRUCTURA DE DECLARACION



ESTADO ASIGNACION SE USA PARA LA ESTRUCTURA DE ASIGANCION DEL PROGRAMA



LOS ESTADOS CONDCIONALES_IF,FOR,WHILE, DOWHILE SE UTILIZAN PARA LAS ESTRUCTURAS DE FOR,IF,WHILE,DOWHILE

IF

FOR

WHILE

DO WHILE

EL CONDICIONAL_ MULTIPLE SE USA PARA LA ESTRUCTURA DE SWTICH