"Implementation of Computational Methods (Grupo 600)"

TC2037.600

"Actividad Integradora 3.4

Resaltador de sintaxis (evidencia de competencia)"

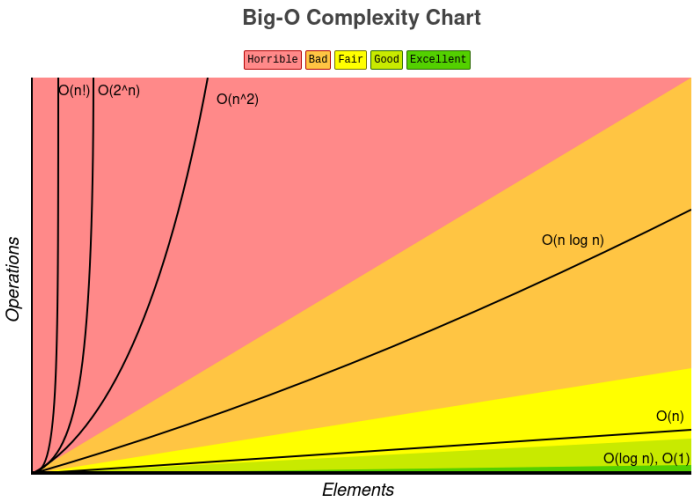
27 de marzo de 2021

Carla Morales López A01639225

Sobre el programa

Lex, analizador léxico, obtiene su input y lo transforma en una secuencia de tokens, crea un programa en lex que produce un programa en c *lex.yy.c*. Este a la vez genera un programa objeto *a.out* que transforma su input en la secuencia de tokens necesarios.

La solución al problema ocupa Lex, donde su declaración de definiciones se realiza en la parte superior en el caso de reservados, funciones, o tipos de datos, mientras que la mayoría de uso de expresiones regulares usadas para nuestro texto default, strings y comentarios se encuentran en la sección de nuestras subrutinas. Al declarar, nos regresan un valor que se ocupa en nuestro header como main, último dónde con el uso de un while y switch case nos determinan de qué tokens se hablan y se usa el estilo establecido en nuestro css. Usando los colores de Visual Studio, en el caso de ser palabras reservadas es color aqua, operadores son royal blue, paréntesis o corchetes de color fuchsia, booleanos son darkorange, los numeros ocupan lateblue, strings de amarillo, comentarios de verde, operadores lógicos en skyblue, binarios en steelblue, funciones predefinidas en indigo, tipos de datos en hotpink y preprocesadores en crimson.



Con el uso de chronos, calculamos que el tiempo ocupado para ejecutar, usando nuestro propio archivo *main.cpp*, este ejecuta en 1752 segundos. Mientras nuestro switch tiene una complejidad de *O(n)* en su peor caso y *O(1)* en el mejor, Lex es la parte importante a calcular debido a su proceso. Ya que resuelve lo que es un DFA, este tiene una complejidad de *O(n)* también ya que realiza una cantidad constante de operaciones por símbolo en su input. Con ayuda del Big-O Complexity Chart, vemos que se trata de una una buena complejidad de tiempo para el programa tanto en el uso de Lex como nuestro *main.cpp.*

El uso de Lex/Flex se expande más allá de resaltadores de sintaxis, nos puede ayudar en la captura de datos en la revisión de documentos o archivos, y debido precisamente a su velocidad del programa, puede ser bastante beneficioso para tareas que, sin el uso de este, serían bastante tediosas. Este igual puede ser usado con fines maliciosos, aunque así es con la mayoría de métodos computacionales que existen, dependen mucho de su aplicación.

Referencias:

Javatpoint (2021) LEX. Recuperado el 28 de marzo de 2022 de https://www.javatpoint.com/lex

Rowell, E. (2012) Big-O Cheat Sheet. Recuperado el 28 de marzo de 2022 de https://www.bigocheatsheet.com/

Schwarz, K. (2012) Lexical Analysis. Stanford University. Recuperado el 28 de marzo de 2022 de https://web.stanford.edu/class/archive/cs/cs143/cs143.1128/lectures/01/Slides01.pdf