Tecnológico de Monterrey

Tecnológico de Monterrey Campus Guadalajara

Implementacion de Metodos Computacionales (Gpo 601)

Actividad Integradora 3.3

Resaltador de sintaxis

Reporte

Joel Isaias Solano Ocampo

A01639289

Índice:

Lenguajes de programación:

- C/C++:

  C es un lenguaje de programación (considerado como uno de lo más importantes en la actualidad) con el cual se desarrollan tanto aplicaciones como sistemas operativos a la vez que forma la base de otros lenguajes más actuales como Java, C++ o C#.
  El lenguaje de C se eligió en esta solución ya que la librería de Flex y Bison funcionan bajo este lenguaje de programación.

- Python:

  Python es un lenguaje de programación de alto nivel, orientado a objetos, con una semántica dinámica integrada, principalmente para el desarrollo web y de aplicaciones informáticas.
  El lenguaje de Python se seleccionó ya que tiene una sintaxis un tanto diferente a lo que es C y C++, tener un resaltador de expresiones con diferentes lenguajes de una forma compleja sería una implementación interesante.

- PHP:

  PHP (acrónimo recursivo de PHP: Hypertext Preprocessor) es un lenguaje de código abierto muy popular especialmente adecuado para el desarrollo web y que puede ser incrustado en HTML.
  Se eligió a PHP también ya que la forma en la que se trabaja con este lenguaje de programación es totalmente diferente a Python y a C/C++ ya que este lenguaje está basado para funcionar en la web.

Expresiones regulares:

Para la solución de la actividad integradora, se eligieron algunas expresiones regulares que se vieron que se compartían en los 3 lenguajes de programación mencionados anteriormente o que solo se enfocan en un lenguaje en específico. Se tomó más en prioridad al lenguaje de programación de C ya que su código es compatible con C++ pero no viseversa, por lo que de haber elegido a C++ de preferencia, algunas expresiones no hubieran sido válidas. Aquellas expresiones regulares son las siguientes:

- Comentarios.
- Definir o incluir.
- Librerías.
- Bloques, funciones y elementos de vector.
- Delimitadores, comillas y nulas.
- Palabras reservadas.
- Constantes de compilación
- Tipos de variables.
- Estados booleanos.
- Bucles.
- Etiquetas.
- Saltos.
- Selección.
- Salida o entrada de consola.
- Operadores y asignaciones.
- Enteros, flotantes y variables.

Reflexión de la solución:

- ## Flex:

  Resultó ser muy complejo de comprender, instalar y sobre todo entender la forma de debuggear con el código. Es una gran herramienta que no descarta en seguir usando el día de hoy pero si pierde peso contra otras librerías más modernas, con mayor uso entre la comunidad y que se desempeñan mejor.

- ## Documentación:

  Encontrar documentación al respecto fue un trabajo arduo y casi imposible, había muy poca en línea y la poca información que había se le podían rescatar muy pocos puntos útiles o que aplicaran a nuestro sistema de desarrollo ya que algunas soluciones estaban basadas en Linux y no Windows.

- ## HTML y CSS:

  Fue agradable volver a trabajar con HTML y CSS, ya tenía rato que no hacía un documento de este tipo y volver a las etiquetas y clases. Agradezco ya haber tenido conocimiento previo porque de no ser así, se me hubiera sido aún más difícil, hubiera tenido que investigar y posiblemente la presentación visual no hubiera sido muy atractiva al final.

- ## Bootstrap 4:

  Me gusta mucho Bootstrap para proyectos escolares, te brinda un montón de elementos útiles y te ofrece una gran cantidad de documentación organizada, fácil

de entender y no tienes que preocuparte por hacer un buen CSS ya que todo lo encuentras en el mismo Bootstrap.

- Resultado final:

Debo decir que no me gusto mucho la paleta de colores de mi resaltador de sintaxis pero supongo que es lo de menos, me gusta como quedo el resultado final, se ve que funciona muy bien mi programa y ahora tengo una idea de cómo funcionan los otros resaltadores de sintaxis de muchos IDEs.

Complejidad del algoritmo:

- myscanner.c:

En este archivo, podemos ver que el algoritmo solo tiene un bucle que recorre todo el archivo de entrada y evalúa de este archivo palabra por palabra donde lo compara con cada expresión regular del archivo myscanner.l donde tenemos más de 100 expresiones regulares. Además, al final, tenemos que imprimir el resultado de las comparaciones en el archivo de salida.

Esto nos queda que por cada ciclo en el bucle, evaluamos n número de tokens en el archivo de entrada donde por cada n se compara con m número de expresiones regulares del archivo myscanner.l donde imprimimos cada resultado en el archivo de salida expresado con j número de veces.

Esto nos queda con una complejidad de Log(On^3) donde Log(O) es el ciclo while, n el número de veces que se evalúa un token, see eleva un exponente cuando se

compara con m número de expresiones y un exponente más cuando se imprime el resultado de salida j veces.

- html_gen.cpp:

  En este archivo, podemos ver como tenemos dos ciclos while, donde uno recorre todo el archivo de plantilla.html y el segundo ciclo recorre todo el archivo de salida.

  Esto nos queda con una complejidad de Log(2On) donde el primer Log(O) lo tenemos para el primer ciclo y el segundo Log(O) para el segundo ciclo. como ambos recorren todo el documento, línea por línea, podemos simplemente sumar ambas complejidades.

Implicaciones éticas:

Encuentro varias implicaciones éticas que se podrían usar actualmente con este tipo de identificador de expresiones regulares. Algo similar a lo que hace Youtube actualmente pero con la voz como entrada, donde se evalúa cada palabra dicha en el video y dependiendo las palabras que se digan se tienen sanciones para el canal. Esto puede limitar la libertad de expresión en plataformas sociales como lo son las redes.

Otra cuestión ética que puedo en lo que esta herramienta se podría involucrar dependería según su uso. Toda tecnología puede usarse en favor o en contra y algo como un analizador léxico es algo complejo, estamos hablando de como un autómata puede interpretar nuestro lenguaje y actuar en base a ello. Podemos limitar la libertad de expresión pero igualmente podemos evitar ataques terroristas, suicidios, predecir eventos o escenarios e incluso regular las plataformas con un analizador léxico, pero igualmente esto sería atentar contra la privacidad de la gente.