Instituto Tecnológico de Estudios Superiores de Monterrey,
Campus Querétaro

Implementación de métodos computacionales

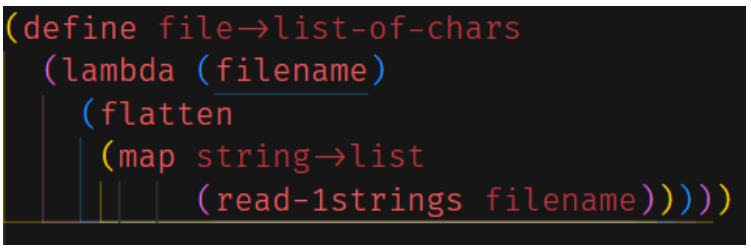
Actividad Integradora 3.4 Resaltador de sintaxis

Diego Ernesto Sandoval Vargas A01709113
Yuna Chung A01709043
Olimpia Helena García Huerta A01708462

Querétaro, Querétaro 14 de abril 2023

Algoritmos y complejidad

`File -> list-of-chars`



```
(define file->list-of-chars
  (lambda (filename)
    (flatten
     (map string->list
          (read-1strings filename)))))
```

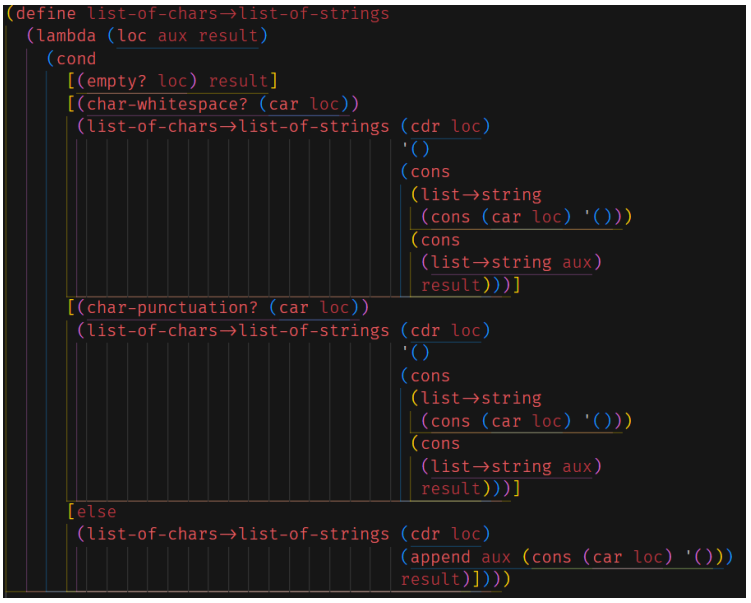
Descripción:

Función que recibe y lee un archivo, devuelve en formato de lista los caracteres que lo conforman.

Complejidad:

O(n) debido a que depende de la suma de la longitud de cada cadena en el archivo y el número de cadenas que se leen

`list-of-chars->list-of-strings`



```
(define list-of-chars->list-of-strings
  (lambda (loc aux result)
    (cond
      [(empty? loc) result]
      [(char-whitespace? (car loc))
       (list-of-chars->list-of-strings (cdr loc)
                                       '()
                                       (cons
                                        (list->string
                                         (cons (car loc) '()))
                                        (cons
                                         (list->string aux)
                                         result)))]
      [(char-punctuation? (car loc))
       (list-of-chars->list-of-strings (cdr loc)
                                       '()
                                       (cons
                                        (list->string
                                         (cons (car loc) '()))
                                        (cons
                                         (list->string aux)
                                         result)))]
      [else
       (list-of-chars->list-of-strings (cdr loc)
                                       (append aux (cons (car loc) '()))
                                       result)])))
```

Descripción:

Función que convierte una lista de caracteres a una lista de strings.

Se utiliza:

- una lista de caracteres: (`LOC -> list-of-chars`) y
- 2 listas vacías: `AUX` y `result`.

La función recorre toda la lista de caracteres para almacenarlos de manera temporal en la lista `AUX`. Al momento de encontrar un espacio en blanco o un salto de línea convierte la lista en un string.

Complejidad:

O(n^2) debido a que el algoritmo debe recorrer cada carácter de la lista de caracteres y en el peor de los casos se realizarán llamadas recursivas para cada carácter.

`file->list-of-strings`

```
(define file→list-of-strings
  (lambda (filename)
    (reverse
     (list-of-chars→list-of-strings
      (file→list-of-chars input-file) '() '()))))
```

Descripción:

Función que utiliza las funciones anteriormente declaradas para convertir un archivo en una lista de strings

Complejidad:

O(n^2) debido a que este algoritmo depende principalmente de `list-of-chars->list-of-strings`

`Type-of-C-reserved-Word`

```
(define (type-of-c-reserved-word word)
  (cond
    ;;access modifiers
   [(regexp-match? #px"\\b(public|private|protected)\\b" word) (string-append"<span class='accessModifier'>"
   word "</span>")]

    ;;data types
   [(regexp-match? #px"\\b(bool|byte|char|decimal|double|float|int|long|object|sbyte|short|string|uint|ulong|
   ushort)\\b" word) (string-append"<span class='dataType'>" word "</span>")]

   ;;Control Flow statement
   [(regexp-match? #px"\\b(if|else|switch|case|default|for|foreach|while|do|break|continue|goto|return|yield|
   checked|unchecked)\\b" word) (string-append"<span class='controlFlow'>" word "</span>")]

    ;;Class or struct keyword
   [(regexp-match? #px"\\b(class|struct|interface|enum)\\b" word) (string-append"<span class='class'>" word "</
   span>")]
```

Descripción:

Función que recibe una palabra y determina mediante regex si es un tipo de palabra reservada de C# devolviendo un string, el cual es una etiqueta span de HTML, con la palabra y la clase a la cual pertenece.

Complejidad:

O(n) ya que depende de la cantidad de palabras que se envíen

HTML-File

```
(define html-file
    (lambda(stringLst htmlStr)
        (cond
        [(empty? stringLst) (string-append htmlStr "</body> </html>")]

        [else
        (html-file (rest stringLst) (string-append htmlStr (type-of-c-reserved-word (first stringLst))))])))
```

Descripción:

Función la cual recibe una lista de string y devuelve un sólo string el cual contiene código html

Complejidad:

O(n) debido a que depende del número de palabras/strings que se envían a la función `type-of-c-reserved-word`

FinalHTML

```
(define finalHtml
    (lambda (input-file)
        (string-append html-header (html-file (file→list-of-strings input-file) "") html-footer html-styles)))
```

Descripción

Función que agrega strings header, footer y los estilos al string tipo html

Complejidad:

O(n^2) ya que depende completamente de la función `list-of-chars->list-of-strings`

Write-HTML

```
(define write-html
    (lambda (input-file output-file)
        (write-file output-file (finalHtml input-file))))
```

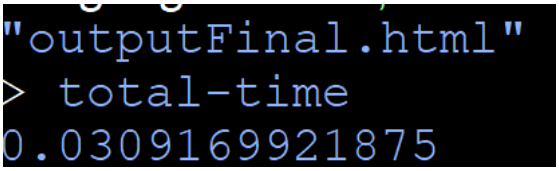
Descripción

Función que recibe un archivo input y genera un archivo tipo html

Complejidad:

O(n^2) ya que depende completamente de la función `list-of-chars->list-of-strings`

Total-time



Reflexión:

Un resaltador de sintaxis si bien es una forma de representar visualmente ciertas características del lenguaje en este caso C#, puede presentar algunas implicaciones éticas que nosotros como programadores tenemos que estar listos para abordarlas.

Primero, al ser una herramienta que analiza texto, tenemos que estar seguros que cumplimos con la protección de datos personales así como datos sensibles, ya que de no estar bien configurado nuestro resaltador, podría informar que ciertos datos corresponden a contraseñas o información sensible.

De igual manera, un resaltador de sintaxis mal programado puede se usado para discriminar ciertos grupos de personas dependiendo de su raza, sexo u orientación sexual, al decidir no resaltarlos o resaltarlos de cierto color que pueda tomarse como burla o discriminación.

Al crear un resaltador de sintaxis nosotros como programadores debemos ser transparentes con las reglas que estamos implementando para crearlo, siguiendo todas las regulaciones y convenciones necesarias para un buen funcionamiento de nuestro sistema.