Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ciencias y Sistemas
Organización de Lenguajes y Compiladores 1
Segundo Semestre 2020
Ing. Mario Bautista, Ing. Manuel Castillo, Ing. Kevin Lajpop
Auxiliares: Luis Mazariegos, Huriel Gómez, Fernando Paz, José Puac

Proyecto 1

ML WEB

Objetivos

General

- Introducir al estudiante al diseño de un analizador léxico que pueda recuperarse de errores.

Específico

- Introducir al estudiante al uso de lenguajes utilizados actualmente como Python.
- Utilizar los conocimientos adquiridos en clase para realizar el análisis léxico de los lenguajes.

Descripción

En la Escuela de Ciencias y Sistemas se propuso un proyecto de un intérprete que englobe los principales lenguajes orientados al diseño web: HTML, CSS y Javascript.
Por lo cual, se le solicita que desarrolle una prototipo para la detección de errores léxicos dentro de la estructura de un proyecto web, dado que al momento de programar en los lenguajes mencionados anteriormente existen caracteres que no son reconocidos por el lenguaje y estos deben ser reportados.

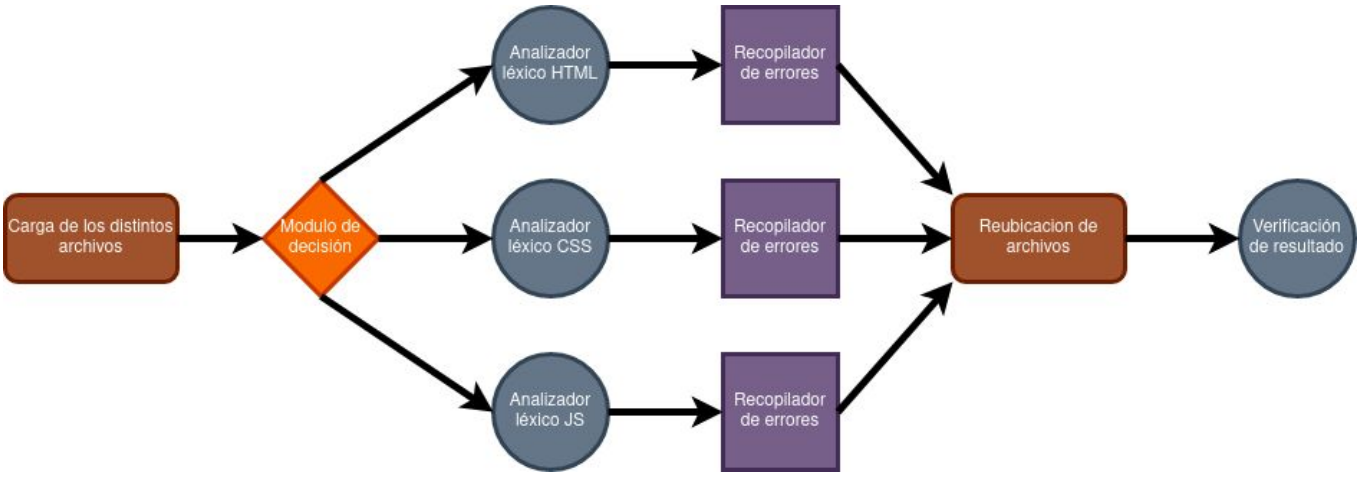
La idea principal del proyecto es que se genere un analisis lexico de cada uno de los distintos archivos de entrada los cuales pueden ser de tipo Marcado como en el caso de HTML, de tipo diseño como CSS o de tipo interpretado como JavaScript.

Se debe realizar un analizador léxico por lenguaje ya que en cada uno de ellos existen distintos tipos de caracteres reconocidos. Y como salida se espera la construcción del proyecto en directorios los cuales estarán identificados por medio de paths las cuales serán tomadas y almacenadas por cada archivo dentro de la ejecución del programa, reportando todos los errores léxicos encontrados.

Cabe mencionar que al momento de reconocer un error léxico el análisis no debe detenerse por ningún motivo, sino que debe tener la capacidad de recuperarse y seguir analizando. La idea es lograr obtener todos los errores.

Flujo del programa

A continuación se detalla el flujo a considerar para la elaboración del proyecto:



La decisión de analizar un archivo dependera de la extensión del mismo, siendo estas:

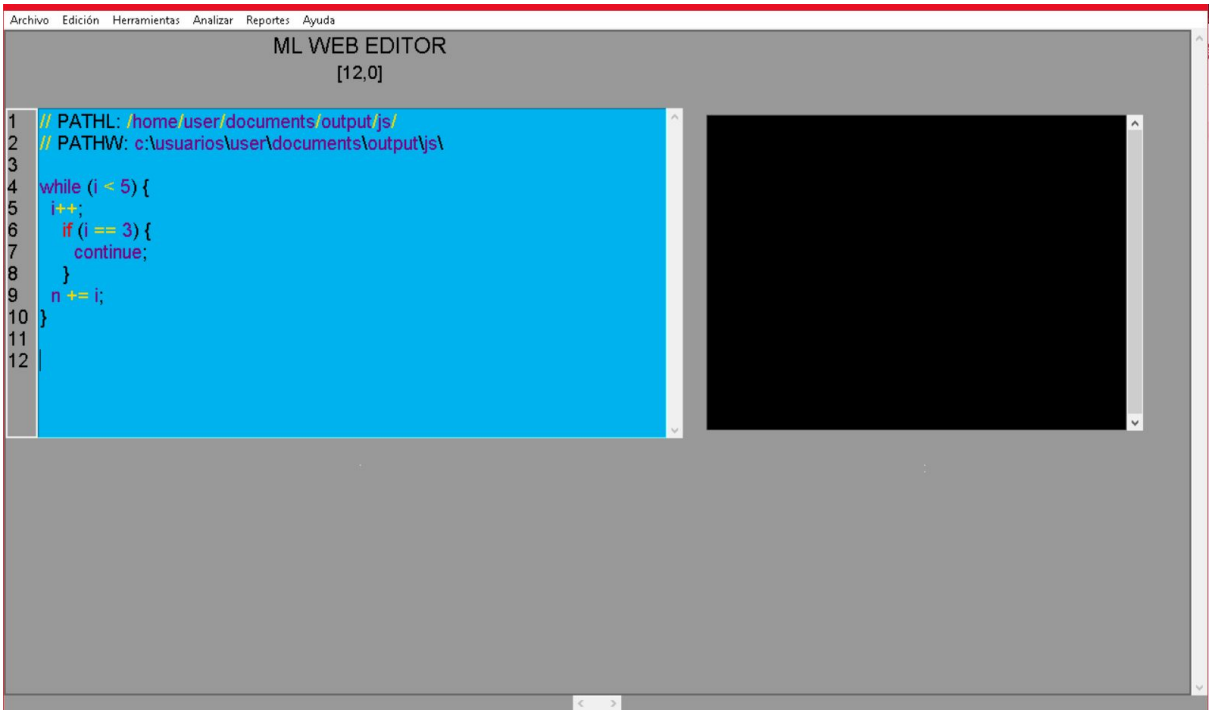
- .html
- .css
- .js

La recopilación de errores debe hacerse de manera independiente, por lo que deberán generarse 3 archivos en formato HTML para el reporte de errores.

Al referirse a reubicación de archivos indica que cada archivo tendrá una ruta al inicio del mismo en forma de comentario la cual indica el directorio donde deberá guardarse. Esto con el fin de que al ejecutar el index.html este funcione correctamente.

Interfaz Gráfica

La solución debe de ser fácil de entender por lo cual se pide que esta cuente con una interfaz gráfica para lo cual se le recomienda utilizar alguna librería para generar una por ejemplo: TKinder.



*PATHL -> Path Linux
*PATHW -> Path Windows

Esta deberá de cargar los archivos mediante un menú, el cual será utilizado para leer el directorio que contiene los distintos archivos a analizar. Los reportes deben de construirse al finalizar el análisis.
También debe de poseer una consola, para indicar los errores generados mediante el análisis del archivo de entrada.

El menú debe de tener por lo menos las siguientes opciones:

- Nuevo
- Abrir
- Guardar
- Guardar Como
- Ejecutar Análisis
- Salir

También debe de tener la capacidad de resaltar el texto, según las diversas opciones que se detallan a continuación:

- Palabras Reservadas: Rojo
- Variables: Verde
- String y Char: Amarillo
- Int y Boolean: Azul
- Comentarios: Gris
- Operadores: Naranja
- Otro: Negro

*Para HTML: El texto que esté entre etiquetas (<H1>TEXTO NEGRO</H1>) debe de ser negro, no se debe tomar como un identificador.

Y por último, indicar la línea y la columna en la que se encuentra posicionado el cursor, esto con la finalidad de detectar con mayor facilidad los errores reportados tras analizar la entrada.

Analizador para JavaScript

Tomando el cuenta la sintaxis utilizada en este lenguaje debe de identificar los distintos caracteres propios del mismo, el resto de caracteres se tomarán como errores léxicos.

1. Comentarios

Dentro del lenguaje se manejan comentarios de 2 tipos, los unilinea y los multilínea, dentro de estos pueden venir caracteres que no estén definidos dentro del lenguaje pero no deberán marcarse como error ya que son comentarios. A continuación se muestran los ejemplos de comentarios:

Unilinea	//Este es un comentario @ unilinea
Multilínea	/* Este es un comentario @ multilínea */

2. Variables

El lenguaje javascript no realiza una distinción explícita de los tipos de variable, ya que lo realiza mediante el tipado dinámico.

Int	var int = 1
String	var string = “4”
Char	var char = ‘a’
Boolean	var boolean = true
Type	var ...

3. Asignación y declaración de variables

En lenguaje javascript debemos de utilizar la palabra reservada ‘var’ al momento de realizar la declaración de una variable mientras que para la asignación no es necesario debido a que esta ya se encuentra almacenada en la tabla de símbolos.

Declaración	var x = y
Asignación	x *= 5

4. Sentencias de control

a. If

if (EXPRESIÓN)
SENTENCIAS
[ELSE IF/ ELSE]
	var edad = 18;
	
	if(edad >= 18) {
	console.log("Eres mayor de edad");
	} else {
	console.log("Todavía eres menor de edad");
	}

5. Sentencias de repetición (ciclos)

a. For

for (EXPRESIÓN; CONDICION; EXPRESIÓN)
SENTENCIAS
	for (var i = 0; i < 9; i++) {
	n += i;
	mifuncion(n);
	}

b. While

while (*CONDICIÓN*)
SENTENCIAS
	while (n < 3) {
	n ++;
	x += n;
	}

c. Do ... while

do
  SENTENCIAS
while (CONDICIÓN)
	do {
	  i = i + 1;
	  result = result + i;
	} while (i < 5);

6. Sentencias de escape

Estas son utilizadas para el control del flujo de ejecución, en el caso de javascript su implementación es la siguiente.

Continue	while (i < 5) {
	  i++;
	  if (i == 3) {
	  continue;
	  }
	  n += i;
	}

Break	while (true) {
	  z += 1;
	  if (z === 10 && x === 10) {
	  break;
	  } else if (z === 10) {
	  break;
	  }
	  }

Return	return EXPRESIÓN; return false; return x; return x + y / 3;

7. Métodos y funciones

Para ambos se utiliza la misma sintaxis la diferencia será al momento de retornar un valor.	function nombre (PARÁMETROS) SENTENCIAS
Otra forma de implementación es mediante la asignación a una variable.	Nombre: function(PARAMETROS) SENTENCIAS
Y tenemos la implementación mediante la estructura lambda.	OBJETO.MÉTODO((PARÁMETRO) => SENTENCIAS);

8. Clases y método constructor

constructor (PARÁMETROS)
  SENTENCIAS
	constructor(alto, ancho) {
	  this.alto = alto;
	  this.ancho = ancho;
	}

class nombre
  SENTENCIAS
	class Rectangulo {
	constructor(alto, ancho) {
	  this.alto = alto;
	  this.ancho = ancho;
	}
	}

9. SENTENCIAS

Las sentencias son un conjunto de instrucciones que conforman el funcionamiento interno de un bloque de ejecución.

SENTENCIAS	{
	  Código
	}

10. PARÁMETROS

Son el listado de variables que se espera sean enviados durante la llamada a un metodo o funcion.

PARÁMETROS	function square(number1, number2) {
	  return number1 * number2;
	}

11. EXPRESIÓN

Con expresión se refiere a un conjunto de operaciones que pueden ser reducidas mediante la interpretación, entre los tipos de operación tenemos las Aritméticas, Relacionales y lógicas.

a. Aritméticas

Suma	EXP + EXP	4 + 5 x + y z + " a "
Resta	EXP - EXP	4 - 5 x - y
Multiplicación	EXP * EXP	4 * 5 x * y
División	EXP / EXP	4 / 5 x / y
Potencia	Math.pow(EXP, EXP)	Math.pow(7, 3); // 343

b. Relacionales

Igual	EXP == EXP	4 == 5
Distinto	EXP != EXP	4 != 5
Mayor que	EXP > EXP	3 > 2
Menor que	EXP < EXP	3 < 2
Mayor igual que	EXP >= EXP	x >= (w-4)
Menor igual que	EXP <= EXP	x <= (w-4)

c. Lógicas

Conjunción	EXP && EXP	true && true
Disyunción	EXP || EXP	false || true
Negación	!EXP	!true

d. Paréntesis

Se utiliza para definir de manera explícita la jerarquía de una operación sobre otra.	(EXP)	(4 + 5) // 1

12. Definición de rutas

Dentro de cada archivo .js al inicio se tendra un comentario especial el cual contendrá un path destino del directorio donde deberá ser almacenado.

Path	Ejemplo:
	
	// PATHL: /home/user/documents/output/js/
	// PATHW: c:\usuarios\user\documents\output\js\

NOTA: Es importante tomar en cuenta que al momento de realizar el análisis léxico deberá eliminar los errores obtenidos de cada archivo JS. Esto debido a que al realizar la reubicación del archivo este deberá estar limpio de errores para realizar la prueba de su funcionalidad.

Reconocimiento de la estructura de una operacion aritmetica

Como tema introductorio al análisis sintáctico se le pide la implementación de un analizador básico y sencillo el cual consiste en la verificación de una cadena de entrada la cual será un conjunto de operación aritméticas con las cuales deberá de poner en práctica la jerarquía operacional de los paréntesis.

Deberá reportar si la operación es correcta o si contiene errores. A continuación se muestran algunos ejemplos de las entradas a evaluar:

((4 - 6 *(1/8)/2)+(6-9*())-(5)*(3*x)/(var1))	Correcto
(7/6*2)-(var1 * exp1/r)/(6-8))	Incorrecto

Analizador para CSS

Un archivo CSS (Cascading Style Sheets) es un documento de texto también conocido como hoja de estilo, el cual permite en conjunto con un html crear páginas web más atractivas para el usuario.

A continuación se detalla la sintaxis que dicho lenguaje de maquetación, para lo cual se le solicita que por medio de un análisis léxico identifique los caracteres que son permitidos en dicho lenguaje, así como también reporte como errores léxicos aquellos que no están permitidos.

1. Comentarios

Los comentarios en un CSS son usados para añadir notas explicativas o prevenir que el navegador interprete partes de la hoja de estilos. Dentro del dicho lenguaje el manejo de comentarios es el mismo para los comentario unilinea y multilínea.
Dentro de estos pueden venir caracteres que no estén definidos dentro del lenguaje los cuales no se deberán marcar como error.

A continuación se ejemplifica en uso de los comentarios:

Unilinea y Multilínea	/* %--- Comentario de una sola línea ---$ */ /* Este es un comentario @ multilínea encontrado dentro de un archivo CSS >:) */

2. Estructura General del Archivo

El archivo CSS se basa en un conjunto de reglas contenidas dentro de un bloque, ejemplo:

<SELECTOR1> {
  <LISTA_REGLAS>
}
 	
<SELECTOR2> {
  <LISTA_REGLAS>
}
 	
….
 	
<SELECTORN> {
  <LISTA_REGLAS>
}
	/* Inicio deL archivo css */
	
	p {
	  color: red;
	}
	
	/* pueden venir 1
	o n reglas */
	
	miLabel {
	  color: red;
	  font-size: 5em;
	}
	
	/* fin del archivo */

3. Selector

El selector son uno o varios elementos del HTML que vamos a seleccionar para aplicarle un estilo según las reglas que deseemos aplicar y analizar separados por comas o espacios en blaco.

<SELECTORES> {
  <LISTA_REGLAS>
}
	p {
	  <LISTA_REGLAS>
	}

Un selector puede definirse de varias maneras:

* { … }	* { /* este es un selector universal */
	  color: red;
	  font-size: 5em;
	}

<ID> { … }	div {
	  background-color:#dddddd;
	  color: gray
	}
	h1, h2, h3, h4, h5, h6 {

	margin-top: 0; margin-bottom: 0.5rem; }

#<ID>	#texto-centrado {
	text-align: center;
	}
	
	#link enlace-web {
	color: yellow;
	font-size: 4em;
	position: absolute;
	top: 600px
	}

.<ID>	**.**texto-rojo {
	color: red;
	}

<ID>**:**<ID>	a:hover { … }
<ID>**::**<ID>	*, *::before, ax:after { … }
<ID>#<ID>	div#container{ … }
<ID>#<ID> <ID>	div#header h1, h2{ … } div#content p{ … }

- Un <ID> o un identificador es una secuencia de caracteres alfabéticos [A-Z a-z] incluyendo el guión medio [-] o dígitos [0-9] que comienzan con un carácter alfabético o guion medio.

4. Reglas

Las reglas son propiedades con un valor asociado, las cuales se aplican sobre los componentes del HTML. Dentro de un bloque pueden haber n reglas las cuales finalizan con un "**;**", **a excepción de la última regla que puede o no tener el ";"**.

<PROPIEDAD> **:** <VALOR> **;**
.
.
.
<PROPIEDAD> **:** <VALOR>
	h1, h2 {
	background-color: #00ff00;
	}
	
	body {
	background-color: #00ff00
	}

	div1 {
	color: purple;
	background-color: #d8da3d;
	width: 200px
	}
	
	div2 {
	color: purple;
	background-color: #d8da3d;
	}

- Propiedades: Estos componentes son los encargados de especificar que es lo que se modificara el componente del HTML al que se hace referencia, asignándole los valores que el usuario desee.

color	background-color	background-image
border	Opacity	background
text-align	font-family	font-style
font-weight	font-size	font
padding-left	padding-right	padding-bottom
padding-top	padding	display
line-height	width	height
margin-top	margin-right	margin-bottom
margin-left	margin	border-style
display	position	bottom
top	right	left
float	clear	max-width
min-width	max-height	min-height

- Valores: Estos pueden ser un único valor asignado o una lista de los mismos separados por espacios en blanco o por comas.
  Los valores asociados y permitidos para cada una de las propiedades se describen a continuación:

Unidades de Medida: px, em, vh, vw, in, cm, mm, pt, pc.	15px -15 em
Identificadores y números:	relative -500 50.35 inline-block
Porcentajes:	75%, 12.25%

Colores	red
	#fF0000
	rgba(255,0,0);

url: contiene una cadena de texto delimitada por comillas dobles, dentro de la cual puede existir cualquier carácter.	url("data:image/svg+xml,%3csvg xmlns='http://www.w3.org/2000/svg' width='8' height='8'");
Cadenas: son cadenas de texto delimitada por comillas dobles, dentro de la cual puede existir cualquier carácter.	content: "\2014\00A0";

Ejemplo:

```
.ejemplo{

   margin-top: 0;
   margin-bottom: 0.5em;
   line-height: inherit;
   position: relative;
   font-size: 75%;
   color: #333;
   border-top: 2px solid rgba(0, 0, 0, 0.1);
   content: "\2014\00A0";
   position: absolute;
   background-image: url(“/home/documents/picture.png");
   background-color: rgba(220, 53, 69, 0.9);
   font: 76% arial, sans-serif;
   background: 0 0 0 0.2rem rgba(51, 51, 51, 0.25);


}
```

Analizador para HTML

HTML es un lenguaje de marcado que se utiliza para el desarrollo de páginas de Internet. Se trata de la sigla que corresponde a HyperText Markup Language, es decir, Lenguaje de Marcas de Hipertexto, que podría ser traducido como Lenguaje de Formato de Documentos para Hipertexto.

EL HTML se encarga de desarrollar una descripción sobre los contenidos que aparecen como textos y sobre su estructura, complementando dicho texto con diversos objetos (como fotografías, animaciones, etc).

Es un lenguaje muy simple y general que sirve para definir otros lenguajes que tienen que ver con el formato de los documentos. El texto en él se crea a partir de etiquetas, también llamadas tags, que permiten interconectar diversos conceptos y formatos.

A continuación se detalla la sintaxis de dicho lenguaje , para lo cual se le solicita que por medio de un análisis léxico identifique los caracteres que son permitidos en dicho lenguaje, así como también reporte como errores léxicos aquellos que no están permitidos.

Etiquetas soportadas:

Descripción	**Ejemplo**
<html> está al inicio de un documento HTML e indica a los navegadores que la página tiene código HTML para que pueda leerlo de esa forma. Siguiendo la sintaxis del lenguaje, la etiqueta de cierre </html> será el último elemento del documento.	<HTML></HTML>
<head> es la etiqueta que se utiliza para el encabezado de la página. Su principal función es contener toda la información del funcionamiento del sitio. Debido a esto, es un código encriptado que las personas que entran a la página no pueden ver.	<HEAD></HEAD>
<title> es la etiqueta que da a tu sitio un nombre para que los usuarios puedan identificarlo. Es el título que puedes ver en las pestañas del navegador.	<TITLE></TITLE>

<body> es la etiqueta que contiene todos los elementos individuales del sitio. O dicho de otra forma, comprende todo el **contenido visible del sitio**. Aquí podrás insertar texto, imágenes, videos o cualquier otra funcionalidad que desees mostrar.	<html> <head> <title>Mi página de ejemplo</title> </head> <body> Aquí va el contenido </body> </html>
Títulos y subtítulos *<h1>* *<h2>…* *<h6>* son etiquetas que funcionan para agregar títulos y subtítulos en el cuerpo del texto, lo cual ayuda a jerarquizar la información. Por cierto, te recomendamos utilizar la etiqueta H1 sólo una vez dentro del contenido.	Esta es una etiqueta H1. Utilízala en el título. Esta es una etiqueta H2. Utilízala en los encabezados de secciones. Esta es una etiqueta H3. Utilízala en sub-secciones. Esta es una etiqueta H4. No se usan muy a menudo. Esta es una etiqueta H5. Esta es una etiqueta H6.
Párrafos Las etiquetas y te ayudarán a organizar el texto en párrafos y saltos de línea.	Tu primer párrafo. Tu segundo párrafo. Un enunciado. Un segundo enunciado (pegado al primero).
Imágenes <img> con esta etiqueta podrás agregar imágenes al cuerpo de tu página. Combinarlo con el atributo src te permitirá especificar la ubicación donde se encuentra la imagen, y el atributo alt te ayudará a darle un título a esa imagen para que lo lean buscadores como Google. La estructura de los atributos es la siguiente: primero viene la palabra o abreviatura que lo define (en este caso src es abreviatura de “source” o fuente), luego el signo igual (=) y al último el modificador del atributo entre comillas dobles (“_”) o simples (‘_’).	<img src="https://images.unsplash.com/photo-1535378917042-10a22c95931a">

<table>
<tr>
<td>Hipervínculos

<a> te permite insertar un hipervínculo a la página. Por ejemplo, el link hacia tus redes sociales o hacia otro sitio web con el que desees conectar tu página.</td>
<td>Visita el blog de GoDaddy</td>
</tr>
<tr>
<td>Listas e índices

 sirve para agregar listas numeradas y <ul> para agregar bullets, los cuales mejoran la legibilidad de tus textos.</td>
<td>
Elemento 1
Elemento 2
Elemento 3
Elemento 4
</td>
</tr>
<tr>
<td>Estilo

Aunque usualmente se ubica dentro de la etiqueta <head>, con el atributo <style> puedes definir el estilo de tu contenido en términos de:

● Color
● Tamaño de fuente
● Tipografía, etc.</td>
<td> Hola Mundo </td>
</tr>
<tr>
<td><table>
<tr><th>Etiquetas</th><th>Descripción</th></tr>
<tr><td><table></td><td>Definición de una tabla.</td></tr>
<tr><td><th></td><td>Definición de cabecera de tabla.</td></tr>
<tr><td><tr></td><td>Define una fila en la tabla.</td></tr>
</table></td>
<td><html>
<head>
<title>Ejercicios prácticos HTML5</title>
</head>
<body>
<table border="1">
<caption>Titulo de la tabla</caption>
<tr>
<th>Col.Cab.1</th>
</tr>
<tr>
<td>Celda 1</td>
<td>Celda 2</td>
</tr>
</table>
</body>
</html></td>
</tr>
</table>

<td>	Define una celda en la tabla.
<caption>	Define el nombre o título de la tabla.
<colgroup>	Especifica un grupo de una o más columnas para aplicar formato.
<col>	Especifica las propiedades de una columna de las columnas definidas en un elemento colgroup
<thead>	Define la cabecera de la tabla
<tbody>	Define el cuerpo de la tabla
<tfoot>	Define el pie de la tabla

Reportes

Para tener derecho a calificación deberá cumplir con la entrega de los siguientes reportes:

Árbol generado mediante la identificación de caracteres (JavaScript)

Para este reporte debe de graficar el árbol generado al realizar el reconocimiento de caracteres en lenguaje Javascript, esto es posible debido a que debe de crear estados para el análisis del mismo, el árbol debe de graficar como nodos aquellos estados por los cuales pasó el analizador léxico, incluyendo el estado sumidero donde se hace el reconocimiento y recuperación de errores léxico.

Esta gráfica puede ser creada mediante cualquier herramienta o librería por ejemplo graphviz.

Bitácora de recorrido (CSS)

Para este reporte se deberá implementar una bitácora del recorrido de los estados por los cuales pasó el analizador léxico, mostrando los cambios de estados y los tokens que se aceptaron o errores encontrados. Deberá mostrarse en una consola de texto en la aplicación.

Reporte de Errores Léxicos

La aplicación debe estar en la capacidad de recuperarse de los errores léxicos, para poder corregir de manera eficiente los archivos.

No.	Línea	Columna	Descripción
1	23	5	El carácter ‘@’ no pertenece al lenguaje.
2	10	1	El carácter ‘¬’ no pertenece al lenguaje.
3	...	...	…….

- El reporte para cada archivo analizado debe ser en formato HTML y de existir errores deberá ser visualizado al momento de terminar el análisis.

Reporte de Análisis Sintáctico

Este reporte es la representación del análisis realizado en la sección de reconocimiento de operaciones aritméticas del lenguaje Javascript.

No.	Línea	Operación	Análisis
1	23	((4 - 6 *(1/8)/2)+(6-9*())-(5)*(3*x)/(var1))	Correcto
2	10	(7/6*2)-(var1 * exp1/r)/(6-8))	Incorrecto

Directorios

Cuando realice el análisis de los archivos las rutas deberá obtenerlas y generar las que se acomoden a su respectivo Sistema Operativo teniendo en cuenta que la base del directorio que contendrá el proyecto de salida inicia con la carpeta **output**.

Entregables

Para tener derecho a calificación deberá presentar lo siguiente:

1. Código fuente de la solución.
2. Manual técnico y de usuario.
3. Archivo ejecutable.
4. Repositorio git (Github, Gitlab, BitBucket) con nombre OLC1_Proyecto1_#carné.

Consideraciones

---

1. Trabajar de Manera Individual.
2. Debe poder analizar al menos 2 de los 3 lenguajes para tener derecho a calificarse.
3. Tendrá 20 minutos para la calificación, una vez terminado este tiempo se colocará la nota que se haya obtenido al momento.
4. Copias de proyectos tendrán de manera automática una nota de 0 puntos y serán reportados a la Escuela de Ciencias y Sistemas los involucrados.
5. No se permite el uso de herramientas de análisis. Para el análisis léxico deberá utilizarse estados.
6. Lenguaje de programación Python.

7. El sistema operativo y el IDE son libres.
8. Los archivos de entradas serán proporcionados el día de la calificación.
9. No habrá prórroga y no se recibirán entregas tarde.
10. El proyecto debe contener su número de carné. ej: [OLC1]P1_#carné.
11. Durante la calificación se realizarán preguntas sobre el código para verificar la autoría de este, de no responder correctamente la mayoría de las preguntas se reportará la copia.
12. Fecha de entrega: Martes 15 de septiembre de 2020 antes de las 11:59 PM.

Nota: Tomar en cuenta que la entrega será mediante Uedi y que al concluir el limite de tiempo de entrega este ya no dejará subir el entregable.