Proyecto 2. Valor 2%

Temas de Estudio

Tema 4 Análisis sintáctico.

Tema 5 Traducción orientada por la sintaxis.

Tema 6 Generación de código intermedio.

Objetivo

Aplicar los conocimientos adquiridos en el curso, abordando cada uno de los temas mencionados, estableciendo conexiones entre los conceptos hacia la programación, con el objetivo de profundizar en las ideas fundamentales relacionadas con un compilador.

Software de Desarrollo

El proyecto debe ser realizado en el lenguaje de programación Java, utilizando como IDE NetBeans en modo carácter. No se debe usar el modo gráfico.

Desarrollo

- El proyecto debe tener independencia física en la ejecución, lo que quiere decir que el programa debe poder ser ejecutado en cualquier máquina y en cualquier carpeta.

- No se permite el uso de librerías de terceros, que son fragmentos de código desarrollados por otras personas y encapsulados en componentes que podrían utilizarse como referencia en la solución del problema a través del código fuente.

- Por lo tanto, se sigue el mismo enfoque: leer un archivo de extensión **.html**, el cual va a contener un código HTML y JavaScript; debe realizar una copia del archivo original y enumerar las líneas para poder evidenciar en un archivo con extensión **.txt** los respectivos errores, haciendo referencia a dicha línea, lo anterior, según los requerimientos a evaluar en la tabla de abajo.

- Se requiere leer un archivo con extensión **.html** el cual contendrá código HTML y JavaScript

#	Requerimientos a evaluar
1	**ARCHIVO ERRORES:** Es necesario crear una copia del archivo original, pero con extensión **.txt**, este archivo contendrá el contenido del archivo **.html** pero con las líneas enumeradas en formato de 4 dígitos por ejemplo, 0001 y el contenido de la línea. Y en este archivo especificar los errores indicando la línea donde está el error, puede ser debajo de la línea del error, haciendo referencia a un número de error y la línea del error, con la respectiva descripción del error. Es importante indicar que se deben de identificar todos los errores y en una línea separada
2	**IDENTIFICADORES** En JavaScript, los identificadores son nombres que se utilizan para nombrar variables, funciones y otros elementos del programa. Para que un identificador sea válido y cumpla con las convenciones de nomenclatura, debe seguir ciertos patrones y reglas. Aquí están las reglas y patrones comunes para determinar un identificador correcto en JavaScript: • Deben comenzar con una letra (a-z, A-Z) o un guion bajo (_) o un carácter especial del sistema de escritura del lenguaje (como los caracteres Unicode que representan letras de otros idiomas). • Después del primer carácter, se pueden usar letras, dígitos (0-9), guiones bajos o caracteres especiales del sistema de escritura. • No se pueden usar espacios en blanco ni caracteres especiales como "-", "+", "*", "/", etc. • JavaScript es sensible a mayúsculas y minúsculas, por lo que "miVariable" y "mivariable" se considerarán identificadores diferentes. • No se pueden utilizar palabras reservadas de JavaScript como identificadores • Los identificadores van después de los comandos let o bien var
3	**CONSTANTES** Para definir una constante en JavaScript, debes seguir algunas reglas y convenciones. Aquí tienes las reglas básicas para definir constantes en JavaScript: • El nombre de las constantes aplica las mismas reglas de los IDENTIFICADORES • Debes utilizar la palabra clave **const** para declarar una constante • Debes asignar un valor a la constante en el mismo momento en que la declaras. No puedes declarar una constante sin asignarle un valor • Una vez que asignas un valor a una constante, no puedes cambiar ese valor en el futuro. Intentar hacerlo generará un error • las constantes deben declararse antes de ser utilizadas, y no pueden declararse después de las variables (var, let) en el mismo ámbito. Esto se debe a las reglas de alcance (scope) en JavaScript

4	**ASIGNACIÓN**
	En JavaScript, puedes asignar valores a una variable utilizando varios operadores de asignación. Los operadores de asignación se utilizan para tomar un valor y asignarlo a una variable:
	• Operador de Asignación (=): Este es el operador de asignación más básico y se utiliza para asignar un valor a una variable, ejemplo:
	▪ let numero = 42;
	• Operadores de Asignación Compuesta: Estos operadores realizan una operación y luego asignan el resultado a la variable, ejemplo:
	▪ let total = 10;
	▪ total += 5; // Ahora, 'total' es igual a 15.
	• Operador de Resta y Asignación (-=): Resta un valor de la variable existente, ejemplo:
	▪ let saldo = 100;
	▪ saldo -= 30; // Ahora, 'saldo' es igual a 70
	• Operador de Asignación en Cadena (=) o (+=): Puedes asignar valores a múltiples variables en una sola línea, ejemplo:
	▪ let a, b, c;
	▪ a = b = c = 42;

	Se puede asumir que todo lo que viene a la derecha del último igual en caso que tenga más de uno es correcto. Pero se deben validar que todos los tipos de datos de las variables que aparecen a la derecha del igual coinciden con el tipo de datos de la variable que aparece a la izquierda del igual.

5	**DEFINIR FUNCIONES**
	Para definir una función en JavaScript, debes seguir ciertas reglas y convenciones. Aquí te proporciono las reglas básicas para definir una función en JavaScript
	• Puedes declarar una función en JavaScript utilizando la palabra clave function seguida por el nombre de la función y un conjunto de paréntesis.
	• Después de los paréntesis, se utiliza un bloque de código en llaves { } para definir el cuerpo de la función.
	• Debe haber al menos una línea de comando dentro del bloque del cuerpo de la función
	• Los nombres de las funciones deben seguir las mismas reglas de nomenclatura que las variables
	• Se debe de verificar que los parámetros que recibe la función están correctos y deben de estar contenidos por la apertura y cierre de las etiquetas `<script>` `</script>`
	Ejemplo:
	`<script>`
	function calcularEdad()
	{
	Otros comandos
	}
	`</script>`

6	**ENTRADA DE DATOS** Puedes utilizar formularios HTML para recopilar datos del usuario. Para acceder a los datos del formulario, puedes utilizar JavaScript. Por ejemplo, puedes usar document.getElementById En HTML el control para captura de datos de entrada es el control determinado por la etiqueta <input>, se debe validar lo siguiente: • Que la etiqueta <input> se compone de los atributos type y id • Type: especifica el tipo de campo de entrada que se debe mostrar. Puede tomar valores como "text" que es campo de texto, “password” es campo de contraseña, "email" es campo de correo electrónico, "number" es campo numérico • Id: Asigna un identificador único al campo de entrada. Esto es útil para acceder al campo mediante JavaScript o para asociarlo con etiquetas de <label>. El atributo id debe ser único en toda la página. Ejemplo: <input type="date" id="fechaNacimiento"> Para obtener datos de entrada del usuario considerando document.getElementById("").value, puedes usar este enfoque para acceder a los valores de los elementos de formulario en una página web: • Se debe validar la estructura document.getElementById("ID").value

	• El valor ID que está dentro de las comillas corresponde al ID del control HTML que obtiene el valor determinado por el usuario, se debe validar que ese control esté definido Ejemplo: const nombre = document.getElementById("nombre").value;
7	**SALIDA DE DATOS** Se debe validar que el comando para salida de datos siga esta sintaxis: • document.getElementById(" id del control").innerHTML = resultado; Lo que se indica entre comillas dobles corresponde al ID del control donde se va a mostrar la información. Lo que está a la derecha del igual puede asumirse que está correcto. Debe de seguir la sintaxis indicada, por lo tanto, debe existir el id del control
8	**ESTRUCTURA HTML** Todas las etiquetas tienen una etiqueta de apertura y otra etiqueta de cierre a excepción de la etiqueta **<!DOCTYPE html>** La etiqueta **<!DOCTYPE html>** debe estar determinada de la manera indicada, cualquier omisión de su conformación es un error. La etiqueta **<!DOCTYPE html>** siempre estará al inicio del código HTML. Las etiquetas y el orden correcto de la posición de ellas son la siguiente estructura: <!DOCTYPE html> <html> <head> <title></title> </head> <body> </body> </html>

A continuación, te presentamos ejemplos de programas en HTML y JavaScript que podrían ser utilizados para evaluar el Proyecto 2. No obstante, es importante tener en cuenta que podrían existir espacios en blanco redundantes o líneas innecesarias que tu programa debe ser capaz de gestionar de manera adecuada. Además, con el propósito de realizar pruebas, el tutor incorporará deliberadamente errores entre las líneas para asegurarse de que tu programa genere los mensajes de error correspondientes

Ejemplo 1:

```
<!DOCTYPE html>
<html>
<head>
    <title>Calculadora de Edad</title>
</head>
<body>
    <h1>Calculadora de Edad</h1>

    <label for="nombre">Nombre Completo: </label>
    <input type="text" id="nombre" placeholder="Nombre Completo">

    <label for="fechaNacimiento">Fecha de Nacimiento: </label>
    <input type="date" id="fechaNacimiento">

    <button onclick="calcularEdad()">Calcular Edad</button>

    <p id="resultado"></p>

    <script>
        function calcularEdad() {
          const nombre = document.getElementById("nombre").value;
          const fechaNacimiento = new Date(document.getElementById("fechaNacimiento").value);
          const fechaHoy = new Date();
          const edad = fechaHoy.getFullYear() - fechaNacimiento.getFullYear();

          let resultado = `${nombre}, tienes ${edad} años de edad. Eres `;
          if (edad < 18) {
               resultado += "menor de edad.";
          } else if (edad >= 18 && edad < 65) {
                resultado += "mayor de edad.";
          } else {
               resultado += "adulto mayor.";
          }

            document.getElementById("resultado").innerHTML = resultado;
        }
    </script>
</body>
</html>
```

Ejemplo 2:

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Ejemplo JavaScript</title>
    <script>
        // Función en JavaScript
        function realizarOperaciones() {
            // Obtener el valor del campo de entrada
            var numero = parseFloat(document.getElementById('numeroEntrada').value);

            // Constante
            const constante = 5;

            // Operador de Asignación (=)
            var resultadoAsignacion = numero;

            // Operadores de Asignación Compuesta
            resultadoAsignacion += constante; // Suma y asignación
            resultadoAsignacion *= 2; // Multiplicación y asignación

            // Operador de Resta y Asignación -=
            resultadoAsignacion -= 3;

            // Operador de Asignación en Cadena (=) o (+=)
            let Num1, Num2, Num3;
            Num1 = Num2 = Num3 = 42;

            // Mostrar el resultado en el área de salida
            var cadenaResultado = "El resultado final es: " + resultadoAsignacion + "<br>";
            cadenaResultado += "Num1 = " + Num1 + ", Num2 = " + Num2 + ", Num3 = " + Num3;

            document.getElementById('resultadoSalida').innerHTML = cadenaResultado;
        }
    </script>
</head>
<body>
    <h1>Ejemplo JavaScript con Operaciones de Asignación</h1>

    <!-- Formulario de entrada -->
    <label for="numeroEntrada">Ingrese un número:</label>
    <input type="text" id="numeroEntrada">
    <button onclick="realizarOperaciones()">Realizar Operaciones</button>

    <!-- Área de salida -->
    <div id="resultadoSalida"></div>
</body>
</html>
```

Honestidad Académica

https://audiovisuales.uned.ac.cr/play/player/23048

Nota Importante

Cada estudiante es responsable del contenido que entrega, si no es el archivo correcto, no podrá entregarlo posterior a la fecha establecida.

Si el contenido del archivo coincide con algún otro estudiante, o se comprueba que no es de su autoría, se aplicaría lo indicado en la plataforma en el documento Lineamientos ante casos de plagio

Indicaciones Importantes

- Es obligatorio que incluya todo el directorio donde se encuentra el Proyecto 2.

- Este proyecto debe estar desarrollado en NETBEANS que es la herramienta oficial de la asignatura.

- El programa debe ser modular, utilizando de la mejor manera funciones definidas por usted.

- Los trabajos deben realizarse en forma individual. Dentro del código del programa debe de indicar la documentación que explique cómo fue realizado el programa.

- Si utiliza código de algún ejemplo del libro, o de otra fuente que no sea de su autoría, debe de indicarlo.

- Comprima todos los archivos en un solo archivo .zip o .rar.

- Nombre del archivo que envía: debe ser nombre y primer apellido del estudiante, y nombre del proyecto. Ejemplo: **JuanRojas-Proyecto2**

- La entrega de Proyecto 2 en las fechas establecidas en la plataforma de aprendizaje en línea Moodle en el apartado que se indique.

- Si no concluyó a tiempo este proyecto, debe entregar lo que pudo hacer e incluir una carta explicando las razones por las cuales no finalizó.

Rúbrica de Evaluación

Criterio	Cumple de manera excelente lo indicado en la evaluación	Cumple medianamente lo indicado en la evaluación	Cumple en contenido y formato, pero los aportes no son significantes	No cumple o no presenta lo solicitado
Realización punto #1	5	3	2	1
Realización punto #2	10	5	3	1
Realización punto #3	10	5	3	1
Realización punto #4	15	10	5	1
Realización punto #5	15	10	5	1
Realización punto #6	10	5	3	1
Realización punto #7	10	5	3	1
Realización punto #8	25	15	10	1
TOTAL	**100**			