Instituto Tecnológico de Costa Rica
Escuela de Ingeniería en Computación
Compiladores e Intérpretes
Prof: Andréi Fuentes L.
Mayo, 2013

TAREA PROGRAMADA 3

Introducción

El objetivo de esta tarea es familiarizarse con los conceptos relativos a la fase de análisis semántico de un compilador, mediante la construcción de un analizador semántico para XHTML.

El proyecto podrá será desarrollado a partir de los entregables de la tarea programada I y II, ya que el analizador semántico hará uso del árbol de análisis sintáctico generado por la segunda tarea programada para realizar sus validaciones. El uso del scanner y del parser de la primera y segunda tarea programada, respectivamente, es opcional.

El programa deberá tomar el código fuente, pasarlo por el scanner, enviar los tokens al parser para que genere el árbol, y posteriormente hacerle el análisis semántico al mismo. Ahora, también se puede ejecutar el análisis semántico en la misma corrida que el análisis sintáctico.

En caso de que hayan errores semánticos, el programa deberá imprimir los mensajes de error correspondientes. En caso de que no hayan errores, el analizador deberá imprimir el árbol de análisis sintáctico, incluyendo nodos para los atributos de cada una de las etiquetas.

Acerca del analizador semántico

La idea en esta tarea es que validen que las diferentes etiquetas tengan solamente atributos válidos para dichas etiquetas (por ejemplo, sólo la etiqueta <a> puede tener el atributo href).

Para eso, tomen como base las etiquetas válidas en XHTML (se muestra la lista de etiquetas que deben de manejar más adelante en el documento), y los atributos válidos para cada etiqueta. Si hay atributos que no son válidos en HTML5, entonces deberán mostrar un mensaje de error que indique eso.

Adicionalmente, deberán validar que los valores asignados a cada atributo sean válidos, esto es, correspondan con los valores esperados para dichos atributos.

En caso de que se encuentre algún error, deberán desplegar el mensaje de error apropiado y continuar con el análisis semántico.

Para cada error, deberán mostrar la información de la ubicación del error (líneas y columnas). Los errores deberán ser lo suficientemente explicativos, para facilitar la corrección de los mismos.

Descripción del programa

El programa no deberá usar argumentos de línea de comandos, solamente deberá ser un ejecutable que reciba la entrada del `stdin`, escriba los tokens resultantes al `stdout`, y reporte errores al `stderr`.

Como parte de la entrega, deberán tener un archivo MAKEFILE (en el caso de java podrían usar ANT o MAVEN, de forma alternativa) que tenga todas las instrucciones necesarias para compilar/ configurar el programa, de manera que con solo los archivos entregados, se pueda ejecutar el comando MAKE, y se generen automáticamente los ejecutables de la tarea.

Para invocar el programa, se deberá ejecutar el siguiente comando (asumiendo que el MAKEFILE construye el archivo `analyzer`):

```
./analyzer < archivoFuente
```

Elementos del HTML a validar

Los siguientes son los elementos que deberán manejar en la gramática del HTML.

`Comentarios`	`dl`	`html`	`script`	`title`
`Doctype`	`dt`	`img`	`span`	`p`
`a`	`dd`	`input`	`strong`	`ul`
`b`	`em`	`li`	`style`	`code`
`blockquote`	`embed`	`link`	`table`	`div`
`body`	`pre`	`meta`	`td`	`textarea`
`br`	`form`	`hr`	`th`	`option`
`button`	`h1-h6`	`ol`	`tr`	`head`
`caption`	`head`	`option`	`textarea`	

Pruebas funcionales

Como parte de la entrega de la tarea, tienen que desarrollar y entregar una serie de archivos de prueba que permitan verificar el correcto funcionamiento del analizador semántico. Los archivos de prueba deberán incluir tanto casos válidos (i.e., HTML correcto), como casos inválidos (HTML incorrecto).

Aspectos técnicos

El proyecto podrá ser escrito en C, C++ o Java, y deberá de funcionar en el sistema operativo Linux. En caso de requerir librerías adicionales para compilar y ejecutar el programa, deberán especificarlo en la documentación, ya que de lo contrario se descontarán puntos en la evaluación.

Documentación

La documentación es un aspecto de gran importancia en el desarrollo de programas, especialmente en tareas relacionadas con el mantenimiento de los mismos.

La documentación externa deberá incluir:

- Tabla de contenidos

- Descripción del problema

- Diseño del programa: decisiones de diseño, algoritmos usados, forma de análisis del parse tree, forma de implementación de las funciones de validación, etc.

- Librerías externas usadas: las librerías externas deben ser solamente de funcionalidades secundarias, y no de las funcionalidades claves de la tarea (hacer scanning, parsing, y anáisis semántico del XHTML).

- Análisis de resultados: objetivos alcanzados, objetivos no alcanzados

- Manual de usuario (instrucciones uso, de ejecución y de compilación usando MAKEFILE). Es importante que especifiquen de manera clara todos los pasos que se tienen que efectuar para correr la tarea. Las instrucciones de compilación deben ser de línea de comandos (no de IDE). Deben explicar cómo instalar/configurar las librerías adicionales que usen en la tarea, en caso de haberlas (el MAKEFILE debería hacer eso automáticamente)

- Conclusión personal

Evaluación

Documentación interna	2%
Documentación externa	8%
Funcionamiento del análisis semántico	60%
Revisión de tarea	30%

Aspectos administrativos

- La tarea vale un 15% de la nota del curso

- La tarea se hará en grupos de 2 personas.

- Fecha de entrega: Lunes 17 de Mayo, 9:00 a.m. No se aceptan tareas entregadas después de esa fecha.

- Las dudas con respecto a las tareas deberán ser enviadas a los siguientes correos: andreifu@gmail.com y brivera@ic-itcr.ac.cr

- El medio de entrega será por medio de Github. Los estudiantes deberán enviar un correo a la dirección brivera@ic-itcr.ac.cr, con copia a andreifu@gmail.com, con el link del repositorio de Github en donde se encuentran todos los archivos de la tarea programada. Tanto el nombre del repositorio, como el asunto del correo enviado deberán ser `IC-5701 TP3-nombre1-nombre2-nombre3`

- El código que entreguen debe ser original. En caso contrario, se calificará con nota cero.