PROCESADORES DE LENGUAJE - Curso 2012-2013

Práctica 1: Implementación de un analizador léxico y un analizador sintáctico predictivo recursivo

Un departamento pide a sus profesores que siguiendo un determinado formato especifiquen la información de la asignatura de la que son responsables. Los profesores deben crear un fichero para cada asignatura que contenga la información sobre el código del módulo, el título de la asignatura, el profesor responsable, el curso, los objetivos, el temario y la bibliografía. Dicho formato debe seguir unas determinadas reglas que se describen en las siguientes secciones de la práctica. El director del departamento pide a un alumno de cuarto curso de la asignatura de Procesadores de Lenguaje de la Titulación de Ingeniería Informática que realice un programa que, a partir de los ficheros creados por los profesores, genere una página Web con la información correspondiente de cada asignatura.

La práctica se divide en tres partes. La primera corresponde a la especificación y reconocimiento de la estructura léxica del lenguaje, donde se especifican los componentes léxicos a reconocer haciendo uso de expresiones regulares y donde se tendrá que diseñar e implementar el autómata finito determinista (AFD) para su reconocimiento. En la segunda parte se trata de analizar si la estructura de un fichero fuente con el formato de un programa de una asignatura se corresponde con la sintaxis definida para ese lenguaje haciendo uso del método de análisis sintáctico descendente predictivo recursivo. En la tercera parte se deberán añadir ciertas instrucciones de código (acciones semánticas) durante el proceso de análisis sintáctico para generar una página Web en HTML con la información sobre la asignatura.

Al final del enunciado de la práctica se presenta un ejemplo de fichero de entrada y la página HTML generada.

1. Parte I: Análisis Léxico

Se trata de implementar un analizador léxico haciendo uso de un Autómata Finito Determinista. Debe existir una función que devuelva el siguiente componente léxico del fichero texto fuente. Esta función debe devolver el tipo de componente léxico en forma de constante entera, y su lexema, en forma de cadena. En esta primera parte de la práctica, y con la finalidad de comprobar que funciona correctamente, esta función será llamada desde el programa principal, que solicitará nuevos componentes léxicos hasta que se agote el texto del fichero de entrada. El programa deberá imprimir una lista de componentes léxicos de la forma $(tipo\_compenente\_lexico, lexema)$ como resultado, deteniéndose en el caso de un error léxico e indicando la línea y columna del texto fuente donde éste se produjo.

La distinción entre identificadores y palabras reservadas debe hacerse usando una tabla de palabras reservadas previamente inicializada.

1.1. Especificación léxica

Los componentes léxicos a reconocer son los siguientes:
Constantes numéricas enteras:
$TKN\_NUMINT = digito^+$

Identificadores:
$TKN\_ID = letra\ (letra \mid digito)^*$, siendo
$digito = 0|1|\ldots|9$

$letra = a|b|\ldots|z|A|B|\ldots|Z$

Símbolos ortográficos:
$TKN\_ORT, ; = (){}/ * + - . :$ ¿ ? ¡ ! "<> @# (y otros opcionalmente)

Palabras reservadas:

TKN_MODULO	"MODULO"
TKN_TITULO	"TITULO"
$TKN\_PROFESORES$	"PROFESORES"
TKN_PROF	"PROF"
TKN_CURSO	"CURSO"
TKN_ANYO	"AÑO"
$TKN\_OBJETIVOS$	"OBJETIVOS"
TKN_OBJ	"OBJ"
$TKN\_TEMARIO$	"TEMARIO"
$TKN\_BEGINITEM$	"BEGINITEM"
TKN_ITEM	"ITEM"
$TKN\_ENDITEM$	"ENDITEM"
$TKN\_BIBLIOGRAFIA$	"BIBLIOGRAFIA"
TKN_BIB	"BIB"

2. Parte II: Análisis Sintáctico

Se trata de implementar un analizador sintáctico direccional determinista descendente predictivo recursivo a partir de una gramática LL(1). El programa deberá imprimir como resultado la lista de producciones que generan la derivación más a la izquierda del fichero de entrada. En el caso de que haya errores sintácticos, el programa debe proporcionar la información sobre la línea y columna del texto original donde se produjo el error.

El lenguaje a reconocer viene especificado a continuación. Se trata de un lenguaje para la generación de un programa de una asignatura donde aparece la información del modulo, título de la asignatura, profesor responsable, curso, objetivos, temario y bibliografía. El fichero fuente contiene la información de un único programa de una asignatura.

2.1. Especificación sintáctica

Las consideraciones a tener en cuenta para la definición de la estructura de un fichero de este tipo son:

- Un fichero fuente consta de las siguientes partes: el modulo de la asignatura, el título, la lista de profesores, el curso, el año, la lista de objetivos, la lista de temas, la lista de bibliografía. Estas partes deben existir siempre, y en ese orden, y van introducidas por su correspondiente componente léxico de palabra reservada.
- Cada objetivo viene introducido por la palabra reservada OBJ, cada profesor por la palabra $PROF$, cada punto del temario por la palabra $ITEM$ y cada referencia bibliográfica por la palabra BIB.
- No pueden existir campos vacíos. Siempre debe existir el código del modulo, el título, el profesor, el curso y haber al menos un objetivo, un tema y una referencia bibliográfica.

- Después de cada componente léxico de palabra reservada debe aparecer al menos una palabra.
- El temario tiene una lista de temas. Cada tema puede tener una lista de subtemas. Cada subtema puede tener a su vez una lista de nuevos subtemas. El nivel de anidamiento de subtemas es ilimitado. Pueden no existir subtemas.
- Cada una de las partes del programa de la asignatura termina al comenzar la siguiente.

Se dotará al reconocedor de un mecanismo de recuperación de errores sintácticos basado en el método de recuperación en modo de pánico. Cuando se produzca un error, se debe indicar qué se ha encontrado y qué se esperaba en ese lugar.

3. Parte III: Generación de la página HTML

A partir de la gramática definida en el apartado anterior se deberán añadir las acciones semánticas necesarias para que se genere la página Web conforme se va realizando el análisis sintáctico. Estas acciones se intercalaran en posiciones concretas de las reglas de la gramática.

A continuación, se pueden encontrar las órdenes básicas que podéis usar para generar la página Web. Es suficiente con conocer la siguiente información:

- Una página comienza con la orden <HTML> y acaba con la orden </HTML>.
- Consta de una cabecera <HEAD> ... </HEAD> y un cuerpo <BODY> ... </BODY>.
- La cabecera contiene la orden <TITLE> ... </TITLE> que indica el título de la página y el nombre con el que se guarda en el *bookmarks*.
- El cuerpo contiene la información de la página.
- Se puede poner encabezados con diferentes tamaños de letras con la orden <H1> ... </H1>,<H2> ... </H2>
- Una lista no ordenada se construye con la orden <UL> ... </UL>. Cada item de la lista viene definido por la orden <LI> ... </LI>. Para las listas ordenadas se utiliza <OL> ... </OL>. Se pueden anidar listas dentro de listas.
- Se puede añadir un retorno de carro con la orden
.

4. Resultados

Se pide:

- Implementar el AFD que permite reconocer la estructura léxica de este lenguaje.
- Diseñar una gramática que genere el formato del programa de la asignatura. Comprobar que es LL(1).
- Implementar el analizador descendente sintáctico predictivo recursivo que reconozca si el formato de un programa de una asignatura es correcto o no. El programa debe imprimir la serie de producciones utilizadas en el proceso de derivación.
- Implimentar el mecanismo de recuperación de errores sintácticos en modo de pánico, indicando qué se ha encontrado y qué se esperaba en ese lugar en el caso de errores sintácticos.

- En caso de error léxico o sintáctico se debe informar de la fila y columna dónde se ha producido.
- Implementar las acciones semánticas en la gramática necesarias para generar la página Web del programa de la asignatura. La página Web debe ser lo más sencilla posible. La numeración de los subtemas vendrá dada de la forma tema.subtema.subsubtema.etc. Ejemplo: tema 2, apartado 3, subapartado 1, se numeraría como 2.3.1

DURACIÓN: 3 sesiones
ENTREGA:
Todos los grupos el día 10 de Diciembre en Aula Virtual.

Ejemplo: fichero de entrada con la información sobre la asignatura

```
MODULO 13048
TITULO Procesadores de Lenguaje
PROFESORES
PROF Elena Díaz
PROF Ariadna Fuertes
CURSO 4
AÑO 2012
OBJETIVOS
OBJ Conocer la estructura de un traductor
OBJ Mostrar el campo de aplicabilidad y relación con otras áreas
OBJ Implementar un traductor para un sencillo lenguaje de programación
TEMARIO
BEGINITEM
ITEM Introducción al proceso de traducción
    BEGINITEM
    ITEM ¿Qué es un traductor?
    ITEM Estructura de datos en un traductor
    ITEM Aplicaciones de los sistemas de traducción
    ENDITEM
ITEM Análisis Léxico
    BEGINITEM
    ITEM Funciones del Análisis Léxico
    ITEM Implementación de un analizador léxico
        BEGINITEM
        ITEM Dirigido por tabla
        ITEM Mediante bucles anidados
        ENDITEM
    ITEM Generación automática de analizadores léxicos: FLEX
    ENDITEM
ITEM Análisis Sintáctico
    BEGINITEM
    ITEM Funciones del Análisis Sintáctico
    ITEM Implementación de un analizador sintáctico
        BEGINITEM
        ITEM Descendente
            BEGINITEM
            ITEM Predictivo Recursivo
            ITEM Con tabla
            ENDITEM
        ITEM Ascendente
            BEGINITEM
            ITEM Método LR(0)
            ITEM Método SLR(1)
            ITEM Método LR(1)
            ENDITEM
        ENDITEM
    ITEM Generación automática de analizadores sintácticos: BISON
    ENDITEM
ENDITEM
BIBLIOGRAFIA
```

BIB Aho, A.V., Sethi, R., Ullman. Çompiladores: Principios, Técnicas y Herramientas". McGraw-Hill, 1990.
BIB Louden, K.C. Çompiler Construction: Principles and Practice". PWS Publishing Company, 1997.

Código HTML generado

```
<HTML>
<HEAD>
<TITLE> Temario Modulo 13048 </TITLE>
</HEAD>
<BODY>
<H2> Modulo 13048 </H2>
<H2> Título </H2> Procesadores de Lenguaje
<H2> Profesores </H2> Elena Díaz, Ariadna Fuertes
<H2> Curso 4 </H2>
<H2> Año 2012 </H2>
<H2> Objetivos </H2>
<UL>
<LI> Conocer la estructura de un traductor </LI> <BR>
<LI> Mostrar el campo de aplicabilidad y relación con otras áreas </LI> <BR>
<LI> Implementar un traductor para un sencillo pero representativo lenguaje de programación
</LI> <BR>
</UL>
<H2> Temario </H2>
<OL>
<LI> Introducción al proceso de traducción </LI> <BR>
<OL>
<LI> ¿ Qué es un traductor ? </LI> <BR>
<LI> Estructura de datos en un traductor </LI> <BR>
<LI> Aplicaciones de los sistemas de traducción </LI> <BR>
</OL>
<LI> Análisis Léxico </LI> <BR>
<OL>
<LI> Funciones del Análisis Léxico </LI> <BR>
<LI> Implementación de un analizador léxico </LI> <BR>
<OL>
<LI> Dirigido por tabla </LI> <BR>
<LI> Mediante bucles anidados </LI> <BR>
</OL>
<LI> Generación automática de analizadores léxicos : FLEX </LI> <BR>
</OL>
<LI> Análisis Sintáctico </LI> <BR>
<OL>
<LI> Funciones del Análisis Sintáctico </LI> <BR>
<LI> Implementación de un analizador sintáctico </LI> <BR>
<OL>
<LI> Descendente </LI> <BR>
<OL>
<LI> Predictivo Recursivo </LI> <BR>
<LI> Con tabla </LI> <BR> </OL>
<LI> Ascendente </LI> <BR>
<OL>
```

<LI> Método LR (0) </LI>
<LI> Método SLR (1) </LI>
<LI> Método LR (1) </LI>
</OL>
</OL>
<LI> Generación automática de analizadores sintácticos : BISON </LI>
</OL>
</OL>
<H2> Bibliografia </H2>
<UL>
<LI> Aho , A . V . , Sethi , R . , Ullman . Çompiladores : Principios , Técnicas y Herramientas ". McGraw - Hill , 1990 . </LI>
<LI> Louden , K . C . Çompiler Construction : Principles and Practice ". PWS Publishing , 1997 . </LI>
</UL>
</BODY>
</HTML>