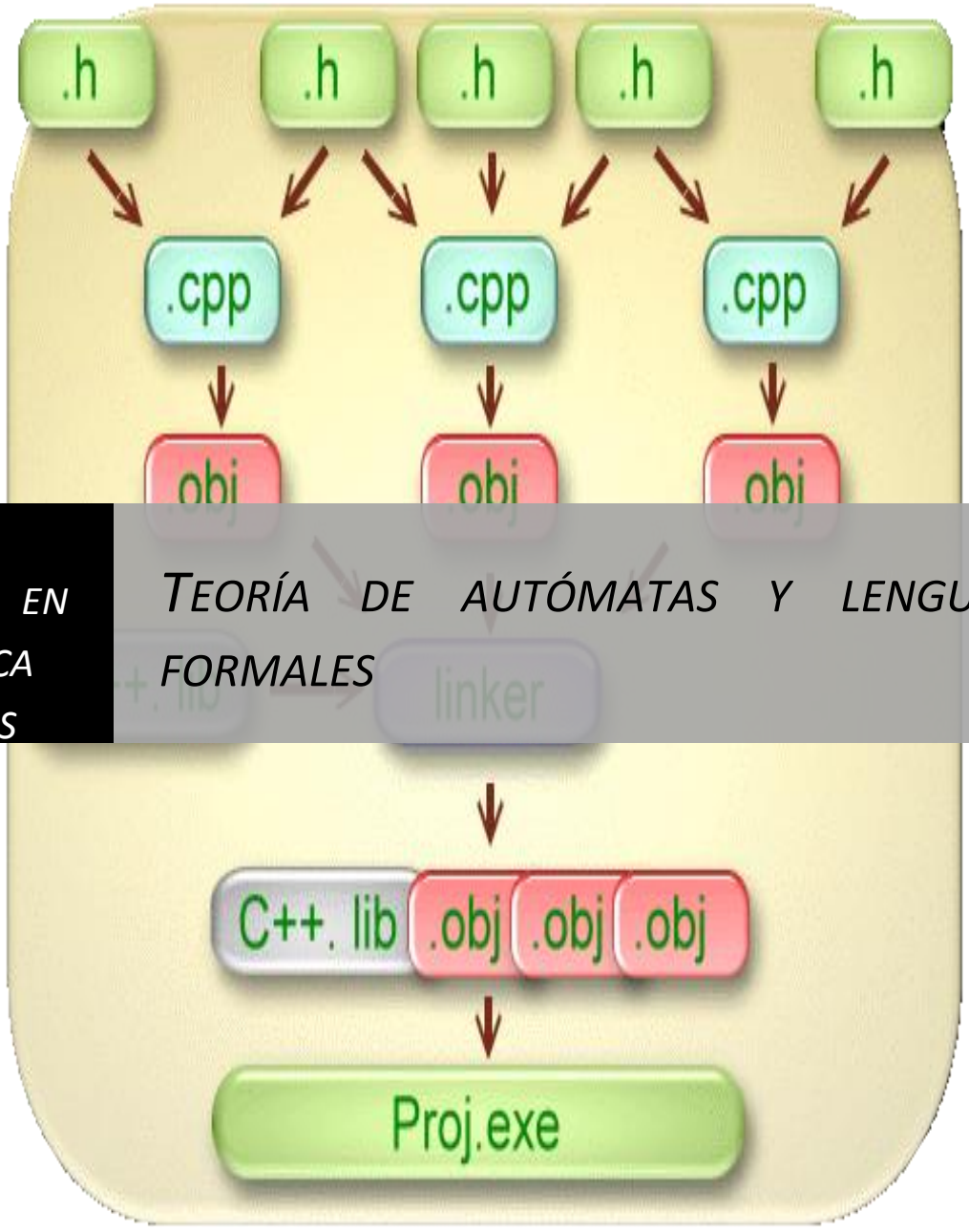

INGENIERÍA TÉCNICA EN INFORMÁTICA DE SISTEMAS

TEORÍA DE AUTÓMATAS Y LENGUAJES FORMALES

Enunciado de la práctica

El objetivo de la práctica de esta asignatura es diseñar e implementar un traductor de Pseudocódigo utilizando las herramientas LEX y YACC.

EL trabajo práctico se dividirá en cuatro fases:

Especificación del lenguaje fuente

Debido a que el pseudocódigo no es un lenguaje de programación, sino una adaptación del lenguaje natural a los rasgos de los lenguajes de programación, la primera tarea consistirá en redactar una especificación del pseudocódigo que se va a utilizar. Como ejemplo se puede utilizar la especificación detallada en http://pseint.sourceforge.net/pseudocodigo.php, aunque el alumno podrá modificarla como considere más oportuno. Como mínimo ***para optar a aprobar*** *la práctica el pseudocódigo deberá contener los siguientes elementos:*

1. *Tipos de datos: Numéricos y constantes de cadena.*
2. *Operadores: Operadores aritméticos básicos´(+,-,*./ y módulo), lógicos (Y, O y NO) y relacionales (MAYOR_QUE, MENOR_QUE, IGUAL, DISTINTO)*
3. *Sentencias:*
    a. *Asignación, que almacena el valor de una expresión en una variable.*
    b. *Condicionales, de tipo SI… y de tipo SI…. SINO…*
    c. *Bucles: De tipo MIENTRAS, PARA y REPETIR… HASTA QUE.*
    d. *Entrada/ salida: Funciones LEER y ESCRIBIR que leen desde teclado y escriben por pantalla el valor de una variable o constante*

Aquellos alumnos que quieran optar a una calificación de ***notable*** *en las prácticas, además de los elementos anteriores su práctica deberá contener:*

1. *Tipos de datos: Arrays unidimensionales. Variables de cadena*
2. *Operadores para concatenar cadenas y concatenar valores numéricos a cadenas.*
3. *Sentencias:*
    a. *Condicionales de selección múltiple*

Aquellos alumnos que quieran optar a una calificación de ***sobresaliente*** *en las prácticas, además de los elementos anteriores su práctica deberá contener:*

1. *Tipos de datos: Arrays n-dimensionales.*

2. *Permitir la existencia de funciones o sub-programas, a los cuales se les puedan pasar parámetros (al menos por valor) y puedan devolver un resultado.*

El resultado de la especificación será un documento en el que el alumno describirá de manera detallada los elementos del lenguaje para el que va a construir el traductor

Construcción del analizador léxico

Una vez realizada la especificación, la siguiente fase consistirá en construir un analizador léxico que permita detectar todos los TOKENS o elementos del lenguaje que anteriormente se ha definido. Para ello se utilizará la herramienta LEX, que permite construir analizadores léxicos a partir de definiciones de expresiones regulares.

El resultado de esta fase será un fichero con la especificación del analizador léxico en formato lex.

Construcción del analizador sintáctico

En primer lugar hay que dedicar tiempo a escribir la gramática del lenguaje especificado, haciendo uso de todos los conocimientos adquiridos en las clases de teoría y prácticas. Esta gramática es el eje de la práctica y debe estar cuidadosamente diseñada, abarcando todas las posibles sentencias que pueden aparecer en un programa fuente. Los errores en el diseño de esta gramática obligarán a volver sobre ella más adelante, lo que conlleva pérdidas de tiempo y esfuerzo. Es conveniente diseñar la gramática primero con "lápiz y papel" para posteriormente implementarla en yacc.

La implementación de la gramática utilizando yacc se hace especificando cada una de las reglas de producción de dicha gramática. EL resultado de dicha fase será un fichero fuente de yacc con dicha especificación.

Generación de código

En esta fase se dotará a la gramática previamente desarrollada de una "semántica", asociando una acción a cada una de las reglas de producción de la gramática. Dicha acción está escrita en lenguaje C y esencialmente consistirá en generar un fragmento del programa objeto.

El programa objeto, resultado de la compilación podrá estar escrito en cualquier lenguaje de programación. En principio el resultado será un programa en ANSI-C, ***compilable*** *con el compilador GNU de c (gcc) aunque a elección del alumno el resultado puede estar en cualquier*

otro lenguaje de alto nivel (java, C++, Pascal, etc.) o a nivel de ensamblador (en este caso se recomienda utilizar algún ensamblador "sencillo" como ENS2001[1] o JASMIN[2])

[1] *http://usuarios.multimania.es/ens2001/*

[2] *http://jasmin.sourceforge.net/*

Planificación y Entregas

A modo de guía se propone como una planificación razonable para llevar a cabo la práctica la siguiente planificación:

28-Fabrero	*Especificación del lenguaje*
7-Marzo	*Especificación del lenguaje*
14-Marzo	*Analizador léxico*
21-Marzo	*Analizador léxico*
28-Marzo	*Analizador léxico*
4-Abril	*Analizador Sintáctico*
11- Abril	*SEMANA SANTA*
	PRIMERA ENTREGA
18- Abril	*Analizador Sintáctico*
25- Abril	*Analizador Sintáctico*
2-Mayo	*Generación de código*
9-Mayo	*Generación de código*
16-Mayo	*Generación de código*
23-Mayo	*Generación de código*
	(junio) SEGUNDA ENTREGA

- ***Primera entrega****: Especificación del lenguaje y del analizador léxico. Esta entrega no será evaluada aunque será obligatoria para optar a la presentación de la práctica final.*
- ***Segunda entrega****: En traductor completo (especificación léxica, sintáctica y semántica) además de todos los ficheros auxiliares necesarios para compilarlo. Se incluirá la especificación del lenguaje, ejemplos para probar el traductor y una memoria en la que se detallará los aspectos mas relevante de la práctica realizada, los cambios hechos en*

la especificación del lenguaje y en el analizador léxico con respecto a la primera entrega, así como las instrucciones necesarias para compilar el traductor, ejecutarlo y probarlo.

Evaluación

La evaluación de la práctica se hará mediante una presentación y defensa ante el profesor. Dicha presentación será calificada cualitativamente como "mal", "bien" y "muy bien" dependiendo del conocimiento que demuestre el alumno de la práctica que ha realizado. Todo alumno que no supere esta presentación (calificación "mal") no superará la práctica y tendrá que presentarla en la convocatoria de septiembre.

Los alumnos que la superen, en función de las funcionalidades que desarrolle su traductor y de la defensa oral (ver enunciado de la práctica/ especificación del lenguaje) tendrán la siguiente nota:

Funcionalidad	***Nota defensa***	***Nota práctica***
No llega a la funcionalidad mínima		*Suspenso*
Funcionalidad mínima	*Bien*	*5*
	Muy bien	*6*
Funcionalidad de "notable"	*Bien*	*7*
	Muy bien	*8*
Funcionalidad de "sobresaliente"	*Bien*	*9*
	Muy bien	*10*

*Sin embargo para aquellas prácticas que a juicio del profesor se consideren **copiadas** (sean "origen" o "destino" de copia) se calificarán con un **suspenso**, lo que implica el suspenso en la asignatura.*