Universidad de Málaga
Departamento de Lenguajes y Ciencias de la Computación
Campus de Teatinos, 29071 MÁLAGA

Examen de Traductores, Intérpretes y Compiladores.
Convocatoria ordinaria de septiembre de 2000.
3er Curso de I.T. Informática de Sistemas.

Apellidos, Nombre: _____ ***Calificación:*** _____

TEORÍA

1.- Sea la siguiente gramática:

```
S   –>  D ';' S
    |   g
    ;
D   –>  T id
    |   D ',' id
    ;
T   –>  int
    |   char
    |   T '[' ']'
    ;
```

Indicar cuál de las siguientes tres tablas se corresponde con la gramática anterior:

a)

	id	int	char	[	]	,	;	$	S	D	T
0	R2	D-4	D-5	R2	R2	R2	R2	R2	1	2	3
1								Aceptar			
2						D-7	D-6				
3	D-8			D-9							
4	R5	R5	R5	R5	R5	R5	R5	R5			
5	R6	R6	R6	R6	R6	R6	R6	R6			
6	R2	D-4	D-5	R2	R2	R2	R2	R2	10	2	3
7	D-11										
8	R3	R3	R3	R3	R3	R3	R3	R3			
9					D-12						
10	R10	R10	R10	R10	R10	R10	R10	R10			
11	R4	R4	R4	R4	R4	R4	R4	R4			
12	R7	R7	R7	R7	R7	R7	R7	R7			

b)

	id	int	char	[	]	,	;	$	S	D	T
0	R2	D-4	D-5	R2	R2	R2	R2	R2	1	2	3
1								Aceptar			
2						D-7	D-6				
3	D-8			D-9							
4	R5	R5	R5	R5	R5	R5	R5	R5			
5	R6	R6	R6	R6	R6	R6	R6	R6			
6	R2	D-4	D-5	R2	R2	R2	R2	R2	10	2	3
7	D-11										
8	R3	R3	R3	R3	R3	R3	R3	R3			
9					D-12						
10	R1	R1	R1	R1	R1	R1	R1	R1			
11	R4	R4	R4	R4	R4	R4	R4	R4			
12	R7	R7	R7	R7	R7	R7	R7	R7			

c)

	id	int	char	[	]	,	;	$	S	D	T
0		D-4	D-5						1	2	3
1								Aceptar			
2						D-6	D-7				
3	D-8			D-9							
4	R5	R5	R5	R5	R5	R5	R5	R5			
5	R6	R6	R6	R6	R6	R6	R6	R6			
6	R2	D-4	D-5	R2	R2	R2	R2	R2	10	2	3
7	D-11										
8	R3	R3	R3	R3	R3	R3	R3	R3			
9					D-12						
10	R1	R1	R1	R1	R1	R1	R1	R1			
11	R4	R4	R4	R4	R4	R4	R4	R4			
12	R7	R7	R7	R7	R7	R7	R7	R7			

2.- Indicar los problemas de la siguiente gramática a la hora de un reconocimiento LALR(1):

```
S   :   S ';' S
    |   L
    |   A
    ;
L   :   g
    |   L a b
    |   L a b a
    ;
A   :   A x y
    ;
```

Nota: la gramática está formada por sólo 7 reglas.

3.- Dadas las siguientes funciones,

```
PROCEDURE uno(x : INTEGER)
    PROCEDURE dos(x : INTEGER)
    VAR
        dosA : INTEGER;
    BEGIN
        RETURN tres(x);
    END dos;
VAR
    unoA : INTEGER;
BEGIN
    IF x > 1 THEN BEGIN
        unoA := dos(x);
        RETURN unoA + 2*x;
    END ELSE
        RETURN 23;
END uno;
PROCEDURE tres(x : INTEGER)
VAR
    unoA : INTEGER;
BEGIN
    IF x%2 = 1 THEN
        unoA := uno(x-3);
    ELSE
        unoA := uno(x-1);
    RETURN unoA + 2*x;
END tres
```

indicar la situación de la pila de registros de activación en el nivel más profundo de recursión, cuando se parte de una llamada de la forma: **uno(7);**

Examen de Traductores, Intérpretes y Compiladores.
Convocatoria ordinaria de septiembre de 2000.
3er Curso de I.T. Informática de Sistemas.

Apellidos, Nombre: _____ ***Calificación:*** _____

PRÁCTICA

Se desea construir un lenguaje de programación que trabaje sobre una pila, y que produzca los resultados de forma inmediata. En esta pila se guardan básicamente números, con los que se puede operar a través de las operaciones de suma (+) y producto (*), además de poder eliminar la cima de la pila (**pop**), y visualizar el valor de la cima (**print**). También se quiere utilizar identificadores; para asignar la cima de la pila a un identificador se utiliza la palabra "def" seguida del identificador que se quiera. Para meter el valor de un identificador en la pila, basta con escribir su nombre.

Ej.:

Entrada	Salida	Pila
1		1
2		1,2
+		3
print	3	3
4		3,4
def x		3
print	3	3
x		3, 4
+		7
def x		-
x		7
print	7	7

Se pide:

- Justificar si es o no necesaria una tabla de símbolos. Si lo es, definir la estructura de un símbolo, y suponer las operaciones básicas de **crear()**, **insertar(...)** y **buscar(...)**.
- Crear la estructura de la pila como si fuese un TAD, con las operaciones necesarias para controlarla. Liberar la memoria siempre que sea necesario.
- Crear los programas Lex y Yacc que realizan las operaciones indicadas, SIN acceder directamente a la estructura interna de la tabla ni de la pila, sino SÓLO a través de las operaciones definidas en su interfaz.
- Suponiendo que se desearan introducir bucles en la pila, podría recurrirse a la solución suministrada por lenguajes como PostScript, en los que se engloban entre llaves las acciones que integran el cuerpo del bucle y luego se usa la palabra reservada "for" para que comience su ejecución. El valor inicial, el final y el incremento del contador de bucle se deben haber colocado en la cima de la pila antes de la llave de apertura. En cada iteración, el sistema coloca el contador de bucle en la cima de la pila. Ej:

Entrada	Salida	Pila
0		0
1		0,1
10		0,1,10
2		0,1,10,2
{		
print		...
+		
} for	...	25
print	25	25

Explicar cómo se podría solucionar este problema. Nótese que el cuerpo de un bucle debería ser compilado de alguna manera antes de ser finalmente interpretado cuando se llegue al "for" correspondiente. ¿Sería necesaria alguna modificación al problema ya resuelto con Lex y Yacc?