1.-) Considere la gramática ambigua

$S \rightarrow AS \mid \mathbf{b}$
$A \rightarrow SA \mid \mathbf{a}$

(a) Construya un NFA en el que cada estado sea un item-LR(0).
(b) Demuestre que el grafo de transiciones goto del DFA de items-LR(0) para esta gramática es el mismo que el del DFA construido a partir del NFA utilizando la construcción de subconjuntos.
(c) Construya la tabla de análisis sintáctico utilizando el algoritmo SLR.
(d) Muestre todos los movimientos admitidos por la tabla de (c) con la entrada abab.
(e) Construya la tabla de análisis sintáctico LR(1).
(f) Construya la tabla de análisis sintáctico LALR.

2.-) Construya la tabla de análisis sintáctico SLR para la gramática:

bexpr $\rightarrow$ bexpr **or** bterm | bterm
bterm $\rightarrow$ bterm **and** factor | bfactor
bfactor $\rightarrow$ **not** bfactor | **(** bexpr **)** | **true** | **false**

3.-) Considere la gramática:

$S \rightarrow CC$
$C \rightarrow \mathbf{a}C \mid \mathbf{b}$

(a) Escriba la tabla de análisis sintáctico LR(1).
(b) Escriba la tabla de análisis sintáctico LALR.

4.-) Considere la siguiente gramática:

$E \rightarrow E + T \mid T$
$T \rightarrow T\ F \mid F$
$F \rightarrow F^* \mid \mathbf{a} \mid \mathbf{b}$

(a) Construya la tabla de análisis sintáctico SLR.
(b) Construya la tabla de análisis sintáctico LR(1).
(c) Construya la tabla de análisis sintáctico LALR.

5.-) Considere la gramática:

$S \rightarrow \mathbf{a}A\mathbf{d} \mid \mathbf{b}B\mathbf{d} \mid \mathbf{a}B\mathbf{e} \mid \mathbf{b}A\mathbf{e}$
$A \rightarrow \mathbf{c}$
$B \rightarrow \mathbf{c}$

Construya la tabla de análisis sintáctico LALR.

6.-) Construya una tabla de análisis sintáctico SLR para la gramática del ELSE ambiguo:

stmt $\rightarrow$ **if** expr **then** stmt
| **if** expr **then** stmt **else** stmt
| other

considerando expr como un terminal. Resuelva el conflicto de acciones de análisis sintáctico de la forma habitual.

1.-) Construya una tabla de análisis sintáctico SLR para la gramática:

$$E \rightarrow E + E \mid E * E \mid (E) \mid \mathbf{id}$$

2.-) Escriba un programa en YACC que tome expresiones aritméticas como entrada y produzca la correspondiente expresión postfija como salida.

3.-) Escriba una calculadora en YACC que evalúe expresiones booleanas.

4.-) Escriba un programa YACC que tome como entrada una expresión regular y produzca un árbol de análisis sintáctico como salida.

5.-) Escriba un programa YACC que reconozca especificaciones de gramáticas escritas en BNF.

6.-) Escriba un programa YACC que reconozca especificaciones de gramáticas escritas en EBNF.

7.-) Escriba un programa YACC que reconozca polinómios en X de grado menor o igual que 2 con coeficientes enteros. Por ejemplo:

-16X^2 + x + 9 ; 30X^2 + 7 ; 17X^2 - 4x ; 8x ; -7 ; 10 ; 0

Se han de reconocer listas de polinomios separadas por retornos de carro.

8.-) Añádale al programa del ejercicio anterior los atributos, declaraciones y acciones semánticas necesarias para evaluar un polinomio en un punto. Para ello, declarar una variable global que contendrá el valor del punto y añadir una nueva regla a la gramática que permita asignale distintos valores. Por ejemplo :

= 1 ; = 5

Aunque 0 es un polinomio, el coeficiente 0 no aparece en ninguno. Un coeficiente de 1 se suprime siempre.

9.-) Considere la gramática que describe el lenguaje de las expresiones lógicas cuyas proposiciones son las formadas por todas las letras minúsculas; con las operaciones de conjunción (/\), disjunción (\/), negación (~), y agrupaciones entre paréntesis. Por ejemplo:

~~p; ~p /\ ~q; ~(p /\ q) \/ ~r /\ s

(a) Escriba un programa YACC que reconozca este lenguaje.

(b) Añada al programa anterior los atributos, declaraciones y acciones semánticas necesarias para evaluar listas de de expresiones separadas por punto y comas (;) y muestre el resultado por pantalla. Considere también la posibilidad de asignar algún valor a las variables, utilizar en este caso el operador (=). Por ejemplo :

p = TRUE; q = FALSE

Por defecto se ha de considerar que cada variable proposicional tiene asignado el valor FALSE. Para las expresiones del ejemplo se debería imprimir los siguiente:

TRUE; FALSE; TRUE

Considere una máquina RAM. Dicho modelo se caracteriza por poseer un único acumulador y un juego de instrucciones que no se modifican a si mismas. Una RAM consta además, de una cinta de entrada de sólo lectura, una cinta de salida de sólo escritura y el siguiente juego de instrucciones:

```
LOAD      operando
STORE     operando
ADD       operando
SUB       operando
MULT      operando
DIV       operando
READ      operando
WRITE     operando
JUMP      etiqueta
JGTZ      etiqueta
JZERO     etiqueta
HALT
```

Un *operando* puede tener las siguientes formas : "=i" o "i" o "*i" en cualquiera de los casos "i" es un entero positivo. Una *etiqueta* es una cadena de caracteres y números que empieza siempre por un caracter.

Escriba un programa YACC que reconozca programas en ensamblador para la RAM. En cada línea del programa ensamblador sólo puede aparecer una instrucción. Una instrucción puede tener o no una etiqueta. En caso de que aparezca una instrucción etiquetada, la etiqueta, aparece al principio de la línea seguida del símbolo ":". Es un error que los operandos de las instrucciones STORE y READ empiecen con un "=". Se considerará un comentario toda aquella cadena que comience con un ";" hasta el final de línea.

Considere el siguiente ejemplo de entrada:

```
        READ    1       ; en la celda 1 se almacena el número
                        ; de elementos a leer "n"
        LOAD    =0
        STORE   2       ; inicializar contador
test:   LOAD    1       ; n
        SUB     2       ; n-contador
        JZERO   end     ; while contador <= n
        LOAD    2
        ADD     =6
        STORE   5
        READ    *5
        LOAD    2       ; incrementar el contador
        ADD     =1
        STORE   2
        JUMP    test
end :   HALT
```