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

Examen de Traductores, Intérpretes y Compiladores.
Convocatoria extraordinaria de Septiembre de 1998.
3er Curso de I.T. Informática de Sistemas.

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

TEORÍA

1.- Escribir en pseudo-código el algoritmo del análisis sintáctico descendente con retroceso.

2.- a) ¿Qué dos directrices emplea LEX cuando encuentra ambigüedad a la hora de encajar patrones?
b) ¿Qué directrices hay que seguir en YACC a la hora de utilizar atributos?

3.- a) ¿Qué problema puede producir la siguiente gramática cuando es analizada por un analizador LALR(1)?:

l_decl --> ε | l_decl decl
decl --> **ID :** tipo | **ID ,** decl
tipo --> **INT** | **REAL**

b) Sustituir la gramática por otra equivalente y que no tenga dicho problema.
c) La nueva gramática que has propuesto, ¿permite hacer fácilmente el chequeo de tipos?

4.- Escribir una gramática lo más completa posible que permita reconocer la declaración de un registro en MODULA-2.

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

Examen de Traductores, Intérpretes y Compiladores.
Convocatoria extraordinaria de Septiembre de 1998.
3er Curso de I.T. Informática de Sistemas.

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

PRÁCTICA

Se desea construir un pequeño intérprete en notación polaca inversa, que funciones con una pila que puede contener tanto valores numéricos como de tipo cadena de caracteres. Esta pila está implementada mediante la siguiente estructura:

```
#define TOPE 100
struct {
   int longitud;
   struct {
      char tipo; // 'n'-numero, 'c'-cadena, 'm'-marca
      union {
         int numero;
         char * cadena;
      } dato;
   } array[TOPE];
} pila;
```

Las operaciones que pueden hacerse con la pila son:

1.- Insertar un valor numérico. Para ello basta con teclear el valor numérico.
2.- Insertar una cadena de caracteres. Para ello basta con teclear la cadena encerrada entre comillas (no se permiten los caracteres de escape).
3.- Duplicar el elemento que hay en la cima de la pila. (Mediante la palabra reservada DUP).
4.- Sacar el elemento que hay en la cima de la pila. (Mediante la palabra reservada POP).
5.- Visualizar el elemento que hay en la cima de la pila. (Mediante el símbolo =).
6.- Visualizar el tipo del elemento que hay en la cima de la pila. (Mediante la palabra reservada TYPE).
7.- Copiar en la cima de la pila al elemento situado en la posición **p**, donde la posición 0 la ocupa la cima de la pila, la **1** el inmediatamente debajo, etc. (Mediante la palabra reservada COPY **p**).
8.- Sumar y multiplicar (Mediante **+** y *****). Hay que considerar que sólo pueden multiplicarse valores numéricos, pero que, sin embargo, la suma puede hacerse entre valores numéricos, de texto, o mezcla de ambos. Cuando en una suma interviene algo de tipo texto, el **+** toma el significado de concatenación. Si aparece un texto y un número, el número se pasa a texto, y se concatena.
9.- Puede ponerse una marca en la pila mediante la palabra reservada MARK.
10.- Contar (sacando el resultado por pantalla) el número de elemento que hay desde la cima de la pila hasta la 1º marca colocada (mediante COUNTTOMARK).
11.- Eliminar todos los elementos que haya desde la cima de la pila hasta la 1º marca colocada (mediante CLEARTOMARK).

El siguiente ejemplo ilustra la pila cuando se hacen distintas operaciones:

INPUT	PILA	OUTPUT
98	98	
2	98, 2	
+	100	
"HOLA"	100, "HOLA"	
DUP	100, "HOLA", "HOLA"	
+	100, "HOLAHOLA"	
+	"100HOLAHOLA"	
5	"100HOLAHOLA", 5	
7	"100HOLAHOLA", 5, 7	

*	"100HOLAHOLA", 35	
POP	"100HOLAHOLA"	
23	"100HOLAHOLA", 23	
12	"100HOLAHOLA", 23, 12	
MARK	"100HOLAHOLA", 23, 12, Marca	
TYPE	"100HOLAHOLA", 23, 12, Marca	Marca.
34	"100HOLAHOLA", 23, 12, Marca, 34	
45	"100HOLAHOLA", 23, 12, Marca, 34, 45	
COUNTTOMARK	"100HOLAHOLA", 23, 12, Marca, 34, 45	2
CLEARTOMARK	"100HOLAHOLA", 23, 12	
TYPE	"100HOLAHOLA", 23, 12	Numerico.
COPY 1	"100HOLAHOLA", 23, 12, 23	
^Z		La pila no esta vacia.

Se pide:

- Hacer el programa LEX correspondiente.
- Hacer el programa YACC correspondiente.

teniendo en cuenta las siguientes consideraciones:

- Las marcas ocupan un lugar en la pila como cualquier otro elemento, y no pueden ser operadas (+ y *).
- En todo momento hay que chequear que la pila no se desborde.
- Si cuando el usuario finalice de trabajar, la pila no está vacía, se le deberá indicar con un mensaje por pantalla.
- Se debe liberar memoria con *free()* siempre que sea posible.

Para ello se da el siguiente esqueleto:

Fichero Exa982l.lex

```
%%
```

Fichero Exa982y.yac

```
%{
#include <stdio.h>
#include <string.h>

#define TOPE 100
```

```
struct {
   int longitud;
   struct {
      char tipo; // 'n'-numero, 'c'-cadena, 'm'-marca
      union {
         int numero;
         char * cadena;
      } dato;
   } array[TOPE];
} pila;

void pop() /* Extrae un elemento de la pila, liberando memoria si es necesario. */ {


}


%}

%union {


}


%%

prog : /* Epsilon */
    | prog sent
    | prog error  {                    }
    ;
sent : NUM          {


                }
    | CADENA       {


                }
    | MARK         {
```

}

| DUP {

}

| POP {

}

| IGUAL {

}

| COUNTTOMARK {

}

| CLEARTOMARK {

```
                }
    | TYPE       {
```

```
                }
    | COPY NUM    {
```

```
                }
    | '+'        {
```

```
                }
    | '*'       {
                }
    ;
%%
#include "exa982l.c"


void yyerror(char * s) {
  printf("%s\n", s);
}
```