PROCESADORES DE LENGUAJE (LABORATORIO) Septiembre 2004 TIEMPO: 90 minutos

1. (2 ptos.) Un compilador de PASCAL admite la utilización de constantes de tipo array, permitiendo su inicialización en la sección correspondiente indicando agrupadamente cada uno de los elementos que lo componen, así, por ejemplo, son válidas las siguientes definiciones:

   alfa: array[1..2] of char = ('a', 'd') ;
   beta : array[1..2,1..2,1..3] of integer = (((1,2,3),(4,5,6)) , ((7,8,9),(1,2,3))) ;
   gamma, kappa : array[1..4,1..2] of integer = ((1,1) , (2,2) , (3,3) , (4,4)) ;

Se pide construir la Gramática LL(1) que aceptaría arrays inicializados teniendo en cuenta las siguientes consideraciones:

- Nuestra gramática debe considerar al menos una instrucción de inicialización, pudiendo haber más.
- En cada instrucción de inicialización se puede inicializar más de una variable separándolas por comas.
- Los tipos básicos que deben considerarse son sólo char e integer y se consideran palabras reservadas (TKN_CHAR y TKN_INT).
- El único constructor de tipos que debe tenerse en cuenta es el correspondiente a los arrays (TKN_ARRAY).
- Como datos recibidos de la parte léxica tenemos: las letras que será incluirán las comillas de apertura y cierre y el carácter, por ejemplo la 'a' (se corresponden con el token TKN_LETRA), los números enteros (TKN_NUM) y los identificadores que son los nombres de las variables (TKN_ID), teniendo además los tokens = (TKN_IGUAL), el ; (TKN_PTOC), los : (TKN_DOSPTOS), el toquen . (TKN_PTO), los corchetes [(TKN_ACOR) y] (TKN_CCOR), la , (TKN_COMA) y los paréntesis de apertura (TKN_APAR) y cierre (TKN_CPAR). Además está la palabra reservada of (TKN_OF).

2. (1 pto.) Indica brevemente cómo implementaste en la práctica 2, la comprobación semántica "declaración de objetos y funciones antes de uso".

3. (2,5 ptos.) Dado el siguiente conjunto de instrucciones de la práctica 2, hacer un programa en Bison que calcule una pequeña estadística sobre **los objetos declarados**. La estadística dará como resultado :

- el número medio de vértices por objeto y,
- el número medio de operaciones de giro y movimientos realizados por objeto.

Programa → DeclaracionObjetos DefEscenario

DeclaracionObjetos → **OBJETOS** DeclObjetos ListaDeclObjetos
ListaDeclObjetos → DeclObjetos ListaDeclObjetos | ε
DeclObjetos → **ID** ListaObjetos **NVERTICES NUM POSICIONES** PosicVertices **TEXTURA ID**
ListaObjetos → **, ID** ListaObjetos | ε

PosicVertices → **NUM NUM NUM** ListaPosicVertices
ListaPosicVertices → **; NUM NUM NUM** ListaPosicVertices | ε
DefEscenario → **ESCENARIO INICIO** Sentencia ListaSents **FIN**
ListaSents → Sentencia ListaSents | ε

Sentencia → **CrearObjeto ID** DatoN | `/* Crea un nuevo objeto ID con NUM vértices */`
Girar ID DatoN |`/*Gira el objeto ID tantos grados como indica DatoN con respecto al eje Z*/`
Dibujar ID |`/* Dibuja el objeto indicado*/`
Borrar ID |`/* Elimina el objeto de la escena */`
MoverX ID PosicionArtimetica | `/*Le suma a las coordenadas X,Y ó Z del */`
MoverY ID PosicionArtimetica |`/*objeto ID el valor resultante de las */`
MoverZ ID PosicionArtimetica |`/* operaciones aritméticas indicadas */`

4. (4 ptos.) Sea la siguiente gramática :

Gramática:

Programa → Operacion Operaciones
Operaciones → Operacion Operaciones | ε
Operacion → ID = ExpresionArit ; |
leer ID ; | `/* lee valor de variable del teclado */`
escribir ExpresionArit ; `/*escribe el valor */`
ExpresionArit → ExpresionArit + ExpresionArit | ID | NUM | - NUM

Se pide:

- Compactar la gramática y escribir las producciones en PCCTS (el apartado donde se define la clase sintáctica).
- Añade los operadores del PCCTS necesarios para la generación del árbol sintáctico y enumera cuales son las consideraciones a tener en cuenta para que el PCCTS genere dicho árbol de forma automática.
- Dibuja el árbol de salida que se genera si la entrada es:

  a = 3 + -2 ; b = a + 7 + 5 ;

- Escribe la función "genera_código" teniendo en cuenta que los cúadruplos de salida para el ejemplo anterior son los siguientes:

```
NEG  t0  2
SUMA t1  3  t0
ASIGNA a  t1
SUMA t2  a  7
SUMA t3  t2  5
ASIGNA t4  b
```