UTN FRBA – SSL – Examen Final – 2022-12-19

Apellido, Nombre:		Legajo:		Nota:	

- Resuelva el examen en el documento compartido para edición; no se aceptan documentos adicionales.
- Durante el examen no se responden consultas; si lo necesita, escriba hipótesis de trabajo, las cuales también se evalúan.

1. *(2 puntos)* Diseñe un LF simple. Escriba todas las declaraciones C necesarias para que la variable `t` sea capaz de almacenar y representar cualquier token de ese LF.
2. Dada la siguiente expresión: `m[a][1].n++`
   a. *(1 punto)* Escriba la secuencia de tokens, no lexemas:
   b. *(2 punto)* Dibuje el árbol de expresión asociado:
   c. *(2 puntos)* Escriba declaraciones e inicializaciones que hagan a la expresión semánticamente válida. No use `typedef`:
   d. *(1 punto)* Escriba una expresión equivalente que use los mismos identificadores pero que no tenga efectos de lado:
   e. *(2 puntos)* Indique si es un valor-l o no. Luego haga los cambios a la declaración para que la expresión pase a ser lo contrario a lo que haya respondido.
3. *(Punto extra)* Describa como implementaría la función `ungetc`.

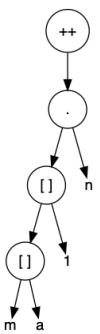
1. Una Resolución

1. Hay muchas resoluciones simple posibles, esta no es la versión más simple pero es una oportunidad para poner en práctica los conceptos:

```
Expresión -> Térmmino | Término + Expresión
Término -> Factor | Factor * Término
Factor -> Constante

Constante = [0-9]+

// Estilo #1 sin alias de tipo
struct {
    enum {
        Numero,
        Adición='+',
        Producto='*'
    }  type;
    int val;
}t;

// Estilo #2 con alias de tipo
typedef enum {
    Numero,
    Adición='+',
    Producto='*'
} TokenType;

typedef int TokenValue;

typedef struct {
    TokenType  type;
    TokenValue val;
} Token;

Token t;
```

   a. `m[a][1].n++` → (**id**, *m*), (**[**), (**id**, *a*), (**]**), (**[**), (**constante**, *1*), (**]**), (**.**), (**id**,*n*), (**++**)

   b.



c.

```
int a=0;
struct{int n;} m[1][2]; // tamaños mínimos para los valores usados
```

d. `m[a][1].n`

e. No es valor-l. Para que sí lo sea: `m[a][1].n`

2. El detalle queda como ejercicio, pero la idea general es que `ungetc` colabore con `getchar` (ó `fgetc`) para mantener registro de los caracteres leídos desde el flujo y los caracteres devueltos al flujo.

v1.0.0-rc.1 2022-12-20