# PROCESADORES DEL LENGUAJE

Práctica Final Primera Entrega:


**Grupo:** 113-leal-zhu
**ID de grupo de prácticas:** 81
**Nombre de los participantes:**

- Liang Ji Zhu
- Ignacio Leal Sánchez

**Correos electrónicos de los participantes:**

- 100495723@alumnos.uc3m.es
- 100495680@alumnos.uc3m.es

En esta primera parte de la práctica final hemos conseguido imprimir las declaraciones de variables globales en C a Lisp. Se han creado las producciones de los no terminales *var_global*, *cuerpo*, *declaracion*, *valor*, *r_declaración* y *nueva_declaración*. Esto es debido a los problemas de herencia que fue lo que intuímos.

C/C++

```
axioma:     var_global  funcion ';'              { printf ("%s%s\n",
$1.code, $2.code) ; }
                r_axioma                 { ; }
            ;
var_global:   declaracion ';'            { sprintf (temp, "%s\n",
$1.code) ;
                                          $$.code = gen_code (temp) ;
}
              | { $$.code = ""; }

funcion:      MAIN '(' ')' '{' cuerpo '}' { sprintf (temp, "(defun
main ()\n\t%s\n)\n(main);", $5.code) ;
                                          $$.code = gen_code (temp) ;
}
            ;

cuerpo:       sentencia ';' cuerpo       { sprintf (temp, "%s\n\t%s",
$1.code, $3.code) ;
                                          $$.code = gen_code (temp) ;
}
            |                           { ; }
r_axioma:                                { ; }
            |   axioma                   { ; }
            ;

declaracion:   INTEGER  IDENTIF valor  { sprintf (temp, "(setq %s %s",
$2.code, $3.code) ;
                                          $$.code = gen_code (temp) ;
}
            ;

valor:      r_declaracion                   {  sprintf (temp, "%d%s",
0, $1.code) ;
                                          $$.code = gen_code (temp) ;}
            |  '=' NUMBER  r_declaracion { sprintf (temp, "%d%s",
$2.value, $3.code) ;
                                          $$.code = gen_code (temp) ;
}

r_declaracion:  ',' nueva_declaracion         {  $$.code = $2.code ; }
            | { $$.code = ")\n"; }
            ;
```

```
nueva_declaracion:  IDENTIF valor { sprintf (temp, ")\n(setq %s %s",
$1.code, $2.code) ;
                                                $$.code = gen_code (temp) ;
}
```

La función main la hemos creado mediante la terminología de comienzo del main y un No Terminal cuerpo este no terminal es un generador de sentencias puede generar de 0 a el número de sentencias deseado. También formatea el código para legibilidad poniendo cada sentencia en una línea e indentándolo.