Yacc: Un generador de anal. sintácticos

- Yet Another Compiler-Compiler
- Yacc es una herramienta UNIX para la generación de analizadores sintácticos para gramáticas LALR
- Veremos que, aunque puede trabajar de forma independiente, lo habitual será usarlo en conjunción con Lex
- Esquema de un fuente Yacc:

```
sección de declaraciones
%%
sección de reglas de traducción
%%
sección de rutinas del usuario
```

Yacc: Un generador de anal. sintácticos

- La introducción mediante la construcción de un reconocedor para la gramática

```
IDENTIFICADOR:  como siempre
CONSTENTERA:    sin signo
OPAS:           :=
MAS:            +
APAR:           (
CPAR:           )
NL:             \n
```

```
instrucciones → instrucciones NL instrucción
              | instrucción

instruccion → IDENTIFICADOR OPAS expresion

expresion → termino
          | expresion MAS termino

termino →   IDENTIFICADOR
          | CONSTENTERA
          | APAR expresion CPAR
```

Yacc: Un generador de anal. sintácticos

- La sección de declaraciones

```
/*
      fichero: instrucciones.y
      Ejemplo para int. de Yacc
*/
%{
        ......
#include <stdio.h>
        ......
%}
%start instrucciones
%token IDENTIFICADOR OPAS CONSTENTERA
%token MAS APAR CPAR NL
%%

       reglas de traducción

%%
        rutinas de usuario
```

código C "literal"

símbolo inicial
(si no aparece, izda. primera regla)

terminales
(suministrados por `yylex()`)

Yacc: Un generador de anal. sintácticos

- La sección de producciones (reglas)

```
      sección de declaraciones
%%
instrucciones: instrucciones NL instruccion
          |    instruccion
;
instruccion: IDENTIFICADOR OPAS expresion
;
expresion: expresion MAS termino
     |     termino
;
termino: IDENTIFICADOR
     |   CONSTENTERA
     |   APAR expresion CPAR
;
%%
        rutinas de usuario
```

Yacc: Un generador de anal. sintácticos

```
        sección de declaraciones
%%
        sección de reglas
%%
int yylex()
{
        .........
}

int main()
{
        yyparse();
}
```

habitualmente, no necesario hacerla
(uso conjunto con Lex)

- En lo que sigue:
  - asumiremos trabajo con Lex

Yacc: Un generador de anal. sintácticos

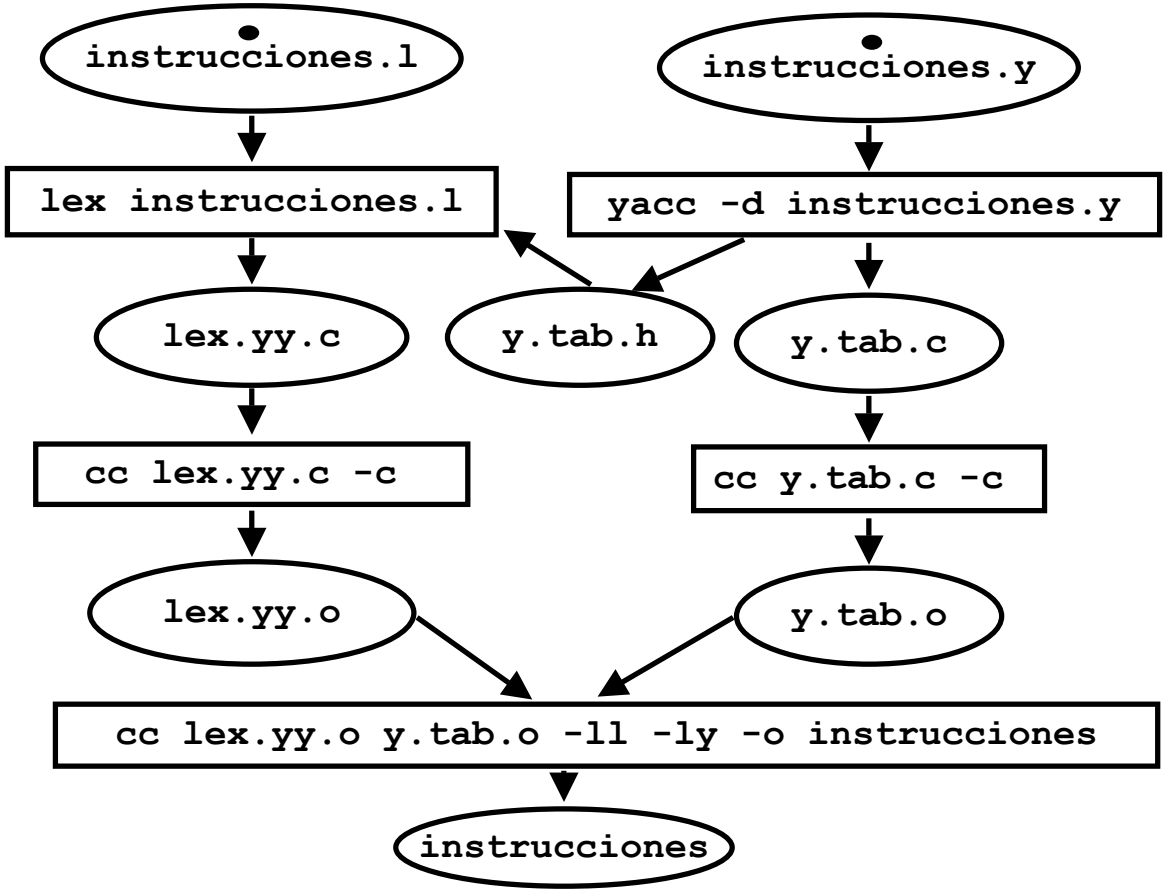
- El fuente del analizador léxico

```
/*
   Analizador léxico para instrucciones
*/
%{
#include <stdio.h>
#include "y.tab.h"
%}
            /*EXPRESIONES REGULARES*/
separador       [\t ]+    /*tab,blanco*/
letra           [a-zA-Z]
digito          [0-9]
identificador {letra}({letra}|{digito})*
constEntera   {digito}+
%%
{separador}       {/*saltarlo*/}
{constEntera}     {return(CONSTENTERA);}
":="              {return(OPAS);}
"+"               {return(MAS);}
{identificador}   {return(IDENTIFICADOR);}
"("               {return(APAR);}
")"               {return(CPAR);}
\n                {return(NL);}
.                 {ECHO;}
%%
```

lo genera Yacc
(`#define..,YYSTYPE..`)

no hay `main()`

Yacc: Un generador de anal. sintácticos



Yacc: Un generador de anal. sintácticos

- Contenido de `'y.tab.h'`

```
#define IDENTIFICADOR   257
#define OPAS            258
#define CONSTENTERA     259
#define MAS             260
#define APAR            261
#define CPAR            257
```

- En casos más generales, tiene mucha más información
  - ya nos irá apareciendo
- Ejemplo de uso:

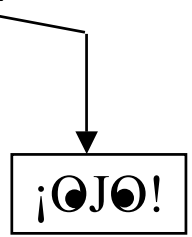
```
vel1  := (vel2+67)+otro
```


```
vel1  := (vel2++67)
```

Yacc: Un generador de anal. sintácticos

- Alternativamente, la sección de reglas se podía haber puesto como

```
    declaraciones (MODIFICAR ADECUADAMENTE)
%%
instrucciones: instrucciones '\n' instruccion
          |    instruccion
;
instruccion: IDENTIFICADOR OPAS expresion
;
expresion: expresion '+' termino
     |     termino
;
termino: IDENTIFICADOR
     |   CONSTENTERA
     |   '(' expresion ')'
;
%%
        rutinas de usuario
```



Yacc: Un generador de anal. sintácticos

- El fuente lex para la nueva versión Yacc cambia ligera-mente

```
/*Analizador léxico para instrucciones*/
%{
#include <stdio.h>
#include "y.tab.h"
%}
            /*EXPRESIONES REGULARES*/
separador           [\t ]+    /*tab,blanco*/
letra               [a-zA-Z]
digito              [0-9]
identificador       {letra}({letra}|{digito})*
constEntera         {digito}+
%%
{separador}       {/*saltarlo*/}
{constEntera}     {return(CONSTENTERA);}
":="              {return(OPAS);}
"+"               {return(MAS);}
{identificador}   {return(IDENTIFICADOR);}
"("               {return(APAR);}
")"               {return(CPAR);}
\n                {return(NL);}
[\+\(\)\n]        {return(yytext[0]);}
.                 {ECHO;}
%%
```