

Συντακτική Ανάλυση με το Εργαλείο Yacc

Διαλέξεις στο μάθημα: Μεταφραστές II
Γιώργος Μανής

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ Η/Υ & ΠΛΗΡΟΦΟΡΙΚΗΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ
DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
UNIVERSITY OF IOANNINA



Γενικά

- # το μεταεργαλείο yacc (bison) είναι ένας **γεννήτορας συντακτικών αναλυτών**
 - # δέχεται σαν είσοδο ένα **μεταπρόγραμμα** που τη σύνταξη μίας γλώσσας (κωδικοποιημένη με σε μία γραμματική)
 - # η έξοδος είναι ένα **πρόγραμμα σε γλώσσα C** που περιέχει τη συνάρτηση `yyparse` η οποία υλοποιεί το συντακτικό αναλυτή
 - # η συνάρτηση αναγνωρίζει ένα **συντακτικά ορθό πρόγραμμα** ή επιστρέφει μήνυμα λάθους
-

Σχηματικά



Σύνταξη Yacc

- # Μέρος Α: Ορισμοί
 - # Μέρος Β: Κανόνες
 - # Μέρος Γ: Συναρτήσεις
-

Παράδειγμα *calc1 - lex*

```
%token INTEGER
%{
#include <stdlib.h>
void yyerror(char *);
#include "y.tab.h"
}%

%%

[0-9]+      {
              yynval = atoi(yytext);
              return INTEGER;
            }

[-+\\n]     return *yytext;

[ \\t]      ; /* skip whitespace */

.           yyerror("invalid character");

%%
```

Παράδειγμα *calc1* - yacc

```
%{  
    int yylex(void);  
    void yyerror(char *);  
}%  
  
%token INTEGER  
  
%%  
  
program:  
    program expr '\n'          { printf("%d\n", $2); }  
    |  
    ;  
  
expr:  
    INTEGER                    { $$ = $1; }  
    | expr '+' expr            { $$ = $1 + $3; }  
    | expr '-' expr            { $$ = $1 - $3; }  
    ;  
  
%%
```

Παράδειγμα *calc2*

```
program:
    program statement '\n'
    |
    ;

statement:
    expr                                { printf("%d\n", $1); }
    | VARIABLE '=' expr               { sym[$1] = $3; }
    ;

expr:
    INTEGER
    | VARIABLE                        { $$ = sym[$1]; }
    | expr '+' expr                   { $$ = $1 + $3; }
    | expr '-' expr                   { $$ = $1 - $3; }
    | expr '*' expr                   { $$ = $1 * $3; }
    | expr '/' expr                   { $$ = $1 / $3; }
    | '(' expr ')'                    { $$ = $2; }
    ;
```

Επίλυση Συγκρούσεων στο Yacc

Οι **αμφισημίες** στις γραμματικές μπορούν να επιλυθούν

- # είτε κατασκευάζοντας μία **μη διαφορούμενη** γραμματική
 - # ή ευκολότερα χρησιμοποιώντας τις **δυνατότητες** που παρέχει ο yacc για το σκοπό αυτό
-

Προτεραιότητα και Συσχέτιση

left '+' '-'

left '*' '/'

- ⌘ **Μεγαλύτερη** προτεραιότητα έχουν οι τελεστές διαίρεσης και πολλαπλασιασμού και **μικρότερη** αυτοί της πρόσθεσης και αφαίρεσης

Προτεραιότητα και Συσχέτιση

left '+' '-'

left '*' '/'

- οι τελεστές αυτοί έχουν **αριστερή συσχέτιση**, δηλ.

$$\alpha + \beta + \gamma = (\alpha + \beta) + \gamma$$

right '='

- ο τελεστής αυτός έχει **δεξιά συσχέτιση**, δηλ.

$$\alpha = \beta = \gamma \text{ σημαίνει } \alpha = (\beta = \gamma)$$

Επίλυση Συγκρούσεων στο Yacc

Αν οι κανόνες προτεραιότητας και συσχέτισης δεν επιλύουν τις συγκρούσεις τότε οι συγκρούσεις επιλύονται με βάση τους ακόλουθους κανόνες

- # σύγκρουση **ελάτωσης-ελάτωσης** επιλύεται επιλέγοντας τον κανόνα που έχει δηλωθεί πρώτος στη γραμματική
 - # σύγκρουση **ολίσθησης-ελάτωσης** επιλύεται επιλέγοντας ολίσθηση
-

μία λίστα με πληροφορίες για φοιτητές

CIS	W7	Abramson,Paul B	CS3001	CS3071	CS3102	CS3132	CS3311	CS3322	CS3361	CS3900	EM2490
CE	X1	O'Rourke,Daniel M	CS3001	CS3041	CS3052	CS3071	CS3082	CS3111	CS3322	CS3900	EM2490
AI	Y6	Naismith,Gregory S	CS3001	CS3052	CS3071	CS3082	CS3311	CS3322	CS3361	CS3900	EM2490
CIS	X4	Sanders,Alexander P	CS3001	CS3071	CS3102	CS3132	CS3311	CS3322	CS3361	CS3900	EM2490
SI	__	Varney,Samantha	CS3041	CS3052	CS3071	CS3082	CS3251	CS3311	CS3432	CS3900	EM2490
CS	W1	Smith,Mark	CS3001	CS3052	CS3071	CS3082	CS3212	CS3232	CS3241	CS3251	CS3900 EM2490
MI	I3	Cooper,Paul	CS3311	CS3561	CS3572	CS3902	MK3322	HM3111	HM3132	HM3142	HM3152 HM3121
MI	I4	Smythe,Helen Ruth	CS3561	CS3572	CS3902	MK3322	MK3311	DI3111	HM3132	HM3152	HM3121 HS3132
CE	Z5	Grant,Ellie	CS3052	CS3071	CS3082	CS3111	CS3311	CS3322	CS3361	CS3900	EM2490

μία λίστα με πληροφορίες για φοιτητές

CIS	W7	Abramson,Paul B	CS3001	CS3071	CS3102	CS3132	CS3311	CS3322	CS3361	CS3900	EM2490
CE	X1	O'Rourke,Daniel M	CS3001	CS3041	CS3052	CS3071	CS3082	CS3111	CS3322	CS3900	EM2490
AI	Y6	Naismith,Gregory S	CS3001	CS3052	CS3071	CS3082	CS3311	CS3322	CS3361	CS3900	EM2490
CIS	X4	Sanders,Alexander P	CS3001	CS3071	CS3102	CS3132	CS3311	CS3322	CS3361	CS3900	EM2490
SI	__	Varney,Samantha	CS3041	CS3052	CS3071	CS3082	CS3251	CS3311	CS3432	CS3900	EM2490
CS	W1	Smith,Mark	CS3001	CS3052	CS3071	CS3082	CS3212	CS3232	CS3241	CS3251	CS3900 EM2490
MI	I3	Cooper,Paul	CS3311	CS3561	CS3572	CS3902	MK3322	HM3111	HM3132	HM3142	HM3152 HM3121
MI	I4	Smythe,Helen Ruth	CS3561	CS3572	CS3902	MK3322	MK3311	DI3111	HM3132	HM3152	HM3121 HS3132
CE	Z5	Grant,Ellie	CS3052	CS3071	CS3082	CS3111	CS3311	CS3322	CS3361	CS3900	EM2490

```
student : school group name modules  
;
```

μία λίστα με πληροφορίες για φοιτητές

CIS	W7	Abramson,Paul B	CS3001	CS3071	CS3102	CS3132	CS3311	CS3322	CS3361	CS3900	EM2490
CE	X1	O'Rourke,Daniel M	CS3001	CS3041	CS3052	CS3071	CS3082	CS3111	CS3322	CS3900	EM2490
AI	Y6	Naismith,Gregory S	CS3001	CS3052	CS3071	CS3082	CS3311	CS3322	CS3361	CS3900	EM2490
CIS	X4	Sanders,Alexander P	CS3001	CS3071	CS3102	CS3132	CS3311	CS3322	CS3361	CS3900	EM2490
SI	__	Varney,Samantha	CS3041	CS3052	CS3071	CS3082	CS3251	CS3311	CS3432	CS3900	EM2490
CS	W1	Smith,Mark	CS3001	CS3052	CS3071	CS3082	CS3212	CS3232	CS3241	CS3251	CS3900 EM2490
MI	I3	Cooper,Paul	CS3311	CS3561	CS3572	CS3902	MK3322	HM3111	HM3132	HM3142	HM3152 HM3121
MI	I4	Smythe,Helen Ruth	CS3561	CS3572	CS3902	MK3322	MK3311	DI3111	HM3132	HM3152	HM3121 HS3132
CE	Z5	Grant,Ellie	CS3052	CS3071	CS3082	CS3111	CS3311	CS3322	CS3361	CS3900	EM2490

```
file      : student
           | file student
           ;
```

```
student : school group name modules
        ;
```

μία λίστα με πληροφορίες για φοιτητές

CIS	W7	Abramson,Paul B	CS3001	CS3071	CS3102	CS3132	CS3311	CS3322	CS3361	CS3900	EM2490
CE	X1	O'Rourke,Daniel M	CS3001	CS3041	CS3052	CS3071	CS3082	CS3111	CS3322	CS3900	EM2490
AI	Y6	Naismith,Gregory S	CS3001	CS3052	CS3071	CS3082	CS3311	CS3322	CS3361	CS3900	EM2490
CIS	X4	Sanders,Alexander P	CS3001	CS3071	CS3102	CS3132	CS3311	CS3322	CS3361	CS3900	EM2490
SI	__	Varney,Samantha	CS3041	CS3052	CS3071	CS3082	CS3251	CS3311	CS3432	CS3900	EM2490
CS	W1	Smith,Mark	CS3001	CS3052	CS3071	CS3082	CS3212	CS3232	CS3241	CS3251	CS3900 EM2490
MI	I3	Cooper,Paul	CS3311	CS3561	CS3572	CS3902	MK3322	HM3111	HM3132	HM3142	HM3152 HM3121
MI	I4	Smythe,Helen Ruth	CS3561	CS3572	CS3902	MK3322	MK3311	DI3111	HM3132	HM3152	HM3121 HS3132
CE	Z5	Grant,Ellie	CS3052	CS3071	CS3082	CS3111	CS3311	CS3322	CS3361	CS3900	EM2490

```
file      : student          modules :
            | file student      | modules module
            ;                  ;
```

```
student : school group name modules
        ;
```

μία λίστα με πληροφορίες για φοιτητές

CIS	W7	Abramson,Paul B	CS3001	CS3071	CS3102	CS3132	CS3311	CS3322	CS3361	CS3900	EM2490
CE	X1	O'Rourke,Daniel M	CS3001	CS3041	CS3052	CS3071	CS3082	CS3111	CS3322	CS3900	EM2490
AI	Y6	Naismith,Gregory S	CS3001	CS3052	CS3071	CS3082	CS3311	CS3322	CS3361	CS3900	EM2490
CIS	X4	Sanders,Alexander P	CS3001	CS3071	CS3102	CS3132	CS3311	CS3322	CS3361	CS3900	EM2490
SI	__	Varney,Samantha	CS3041	CS3052	CS3071	CS3082	CS3251	CS3311	CS3432	CS3900	EM2490
CS	W1	Smith,Mark	CS3001	CS3052	CS3071	CS3082	CS3212	CS3232	CS3241	CS3251	CS3900 EM2490
MI	I3	Cooper,Paul	CS3311	CS3561	CS3572	CS3902	MK3322	HM3111	HM3132	HM3142	HM3152 HM3121
MI	I4	Smythe,Helen Ruth	CS3561	CS3572	CS3902	MK3322	MK3311	DI3111	HM3132	HM3152	HM3121 HS3132
CE	Z5	Grant,Ellie	CS3052	CS3071	CS3082	CS3111	CS3311	CS3322	CS3361	CS3900	EM2490

Εναλλακτικά, αν ένας μαθητής μπορεί να μην παίρνει ενότητες

```
student : school group name modules
        | school group name
        ;
modules : module
        | modules module
        ;
```


μία λίστα με πληροφορίες για φοιτητές

CIS	W7	Abramson,Paul B	CS3001	CS3071	CS3102	CS3132	CS3311	CS3322	CS3361	CS3900	EM2490
CE	X1	O'Rourke,Daniel M	CS3001	CS3041	CS3052	CS3071	CS3082	CS3111	CS3322	CS3900	EM2490
AI	Y6	Naismith,Gregory S	CS3001	CS3052	CS3071	CS3082	CS3311	CS3322	CS3361	CS3900	EM2490
CIS	X4	Sanders,Alexander P	CS3001	CS3071	CS3102	CS3132	CS3311	CS3322	CS3361	CS3900	EM2490
SI	__	Varney,Samantha	CS3041	CS3052	CS3071	CS3082	CS3251	CS3311	CS3432	CS3900	EM2490
CS	W1	Smith,Mark	CS3001	CS3052	CS3071	CS3082	CS3212	CS3232	CS3241	CS3251	CS3900 EM2490
MI	I3	Cooper,Paul	CS3311	CS3561	CS3572	CS3902	MK3322	HM3111	HM3132	HM3142	HM3152 HM3121
MI	I4	Smythe,Helen Ruth	CS3561	CS3572	CS3902	MK3322	MK3311	DI3111	HM3132	HM3152	HM3121 HS3132
CE	Z5	Grant,Ellie	CS3052	CS3071	CS3082	CS3111	CS3311	CS3322	CS3361	CS3900	EM2490

Κανόνες για το lex:

%token school group module name

μία λίστα με πληροφορίες για φοιτητές

CIS	W7	Abramson,Paul B	CS3001	CS3071	CS3102	CS3132	CS3311	CS3322	CS3361	CS3900	EM2490
CE	X1	O'Rourke,Daniel M	CS3001	CS3041	CS3052	CS3071	CS3082	CS3111	CS3322	CS3900	EM2490
AI	Y6	Naismith,Gregory S	CS3001	CS3052	CS3071	CS3082	CS3311	CS3322	CS3361	CS3900	EM2490
CIS	X4	Sanders,Alexander P	CS3001	CS3071	CS3102	CS3132	CS3311	CS3322	CS3361	CS3900	EM2490
SI	__	Varney,Samantha	CS3041	CS3052	CS3071	CS3082	CS3251	CS3311	CS3432	CS3900	EM2490
CS	W1	Smith,Mark	CS3001	CS3052	CS3071	CS3082	CS3212	CS3232	CS3241	CS3251	CS3900 EM2490
MI	I3	Cooper,Paul	CS3311	CS3561	CS3572	CS3902	MK3322	HM3111	HM3132	HM3142	HM3152 HM3121
MI	I4	Smythe,Helen Ruth	CS3561	CS3572	CS3902	MK3322	MK3311	DI3111	HM3132	HM3152	HM3121 HS3132
CE	Z5	Grant,Ellie	CS3052	CS3071	CS3082	CS3111	CS3311	CS3322	CS3361	CS3900	EM2490

Κανόνες για το lex:

```
%token school group module name
```

```
([A-Z][0-9]|"__") {if(trace)ECHO; return group;}
```

μία λίστα με πληροφορίες για φοιτητές

CIS	W7	Abramson,Paul B	CS3001	CS3071	CS3102	CS3132	CS3311	CS3322	CS3361	CS3900	EM2490
CE	X1	O'Rourke,Daniel M	CS3001	CS3041	CS3052	CS3071	CS3082	CS3111	CS3322	CS3900	EM2490
AI	Y6	Naismith,Gregory S	CS3001	CS3052	CS3071	CS3082	CS3311	CS3322	CS3361	CS3900	EM2490
CIS	X4	Sanders,Alexander P	CS3001	CS3071	CS3102	CS3132	CS3311	CS3322	CS3361	CS3900	EM2490
SI	__	Varney,Samantha	CS3041	CS3052	CS3071	CS3082	CS3251	CS3311	CS3432	CS3900	EM2490
CS	W1	Smith,Mark	CS3001	CS3052	CS3071	CS3082	CS3212	CS3232	CS3241	CS3251	CS3900 EM2490
MI	I3	Cooper,Paul	CS3311	CS3561	CS3572	CS3902	MK3322	HM3111	HM3132	HM3142	HM3152 HM3121
MI	I4	Smythe,Helen Ruth	CS3561	CS3572	CS3902	MK3322	MK3311	DI3111	HM3132	HM3152	HM3121 HS3132
CE	Z5	Grant,Ellie	CS3052	CS3071	CS3082	CS3111	CS3311	CS3322	CS3361	CS3900	EM2490

Κανόνες για το lex:

```
[A-Z][A-Z][0-9][0-9][0-9][0-9] {if(trace)ECHO; return module;}
[A-Z][A-Z][0-9][0-9][0-9]      {if(trace)ECHO; return module;}
```

μία λίστα με πληροφορίες για φοιτητές

CIS	W7	Abramson,Paul B	CS3001	CS3071	CS3102	CS3132	CS3311	CS3322	CS3361	CS3900	EM2490
CE	X1	O'Rourke,Daniel M	CS3001	CS3041	CS3052	CS3071	CS3082	CS3111	CS3322	CS3900	EM2490
AI	Y6	Naismith,Gregory S	CS3001	CS3052	CS3071	CS3082	CS3311	CS3322	CS3361	CS3900	EM2490
CIS	X4	Sanders,Alexander P	CS3001	CS3071	CS3102	CS3132	CS3311	CS3322	CS3361	CS3900	EM2490
SI	__	Varney,Samantha	CS3041	CS3052	CS3071	CS3082	CS3251	CS3311	CS3432	CS3900	EM2490
CS	W1	Smith,Mark	CS3001	CS3052	CS3071	CS3082	CS3212	CS3232	CS3241	CS3251	CS3900 EM2490
MI	I3	Cooper,Paul	CS3311	CS3561	CS3572	CS3902	MK3322	HM3111	HM3132	HM3142	HM3152 HM3121
MI	I4	Smythe,Helen Ruth	CS3561	CS3572	CS3902	MK3322	MK3311	DI3111	HM3132	HM3152	HM3121 HS3132
CE	Z5	Grant,Ellie	CS3052	CS3071	CS3082	CS3111	CS3311	CS3322	CS3361	CS3900	EM2490

Κανόνες για το lex:

`[A-Z][-A-Za-z',]*`

`{if(trace)ECHO; return name;}`

μία λίστα με πληροφορίες για φοιτητές

CIS	W7	Abramson,Paul B	CS3001	CS3071	CS3102	CS3132	CS3311	CS3322	CS3361	CS3900	EM2490
CE	X1	O'Rourke,Daniel M	CS3001	CS3041	CS3052	CS3071	CS3082	CS3111	CS3322	CS3900	EM2490
AI	Y6	Naismith,Gregory S	CS3001	CS3052	CS3071	CS3082	CS3311	CS3322	CS3361	CS3900	EM2490
CIS	X4	Sanders,Alexander P	CS3001	CS3071	CS3102	CS3132	CS3311	CS3322	CS3361	CS3900	EM2490
SI	__	Varney,Samantha	CS3041	CS3052	CS3071	CS3082	CS3251	CS3311	CS3432	CS3900	EM2490
CS	W1	Smith,Mark	CS3001	CS3052	CS3071	CS3082	CS3212	CS3232	CS3241	CS3251	CS3900 EM2490
MI	I3	Cooper,Paul	CS3311	CS3561	CS3572	CS3902	MK3322	HM3111	HM3132	HM3142	HM3152 HM3121
MI	I4	Smythe,Helen Ruth	CS3561	CS3572	CS3902	MK3322	MK3311	DI3111	HM3132	HM3152	HM3121 HS3132
CE	Z5	Grant,Ellie	CS3052	CS3071	CS3082	CS3111	CS3311	CS3322	CS3361	CS3900	EM2490

ένα όνομα φοιτητή μπορεί επίσης να αναγνωριστεί σαν school

```
[A-Z][-A-Za-z', ]*  
[A-Z]+
```

```
{if(trace)ECHO; return name;}  
{if(trace)ECHO; return school;}
```

μία λίστα με πληροφορίες για φοιτητές

CIS	W7	Abramson,Paul B	CS3001	CS3071	CS3102	CS3132	CS3311	CS3322	CS3361	CS3900	EM2490
CE	X1	O'Rourke,Daniel M	CS3001	CS3041	CS3052	CS3071	CS3082	CS3111	CS3322	CS3900	EM2490
AI	Y6	Naismith,Gregory S	CS3001	CS3052	CS3071	CS3082	CS3311	CS3322	CS3361	CS3900	EM2490
CIS	X4	Sanders,Alexander P	CS3001	CS3071	CS3102	CS3132	CS3311	CS3322	CS3361	CS3900	EM2490
SI	__	Varney,Samantha	CS3041	CS3052	CS3071	CS3082	CS3251	CS3311	CS3432	CS3900	EM2490
CS	W1	Smith,Mark	CS3001	CS3052	CS3071	CS3082	CS3212	CS3232	CS3241	CS3251	CS3900 EM2490
MI	I3	Cooper,Paul	CS3311	CS3561	CS3572	CS3902	MK3322	HM3111	HM3132	HM3142	HM3152 HM3121
MI	I4	Smythe,Helen Ruth	CS3561	CS3572	CS3902	MK3322	MK3311	DI3111	HM3132	HM3152	HM3121 HS3132
CE	Z5	Grant,Ellie	CS3052	CS3071	CS3082	CS3111	CS3311	CS3322	CS3361	CS3900	EM2490

η λύση είναι να μπει ο κανόνας για το school πρώτα ώστε να αναγνωριστεί πρώτα:

[A-Z]+	{if(trace)ECHO; return school;}
[A-Z][-A-Za-z',]*	{if(trace)ECHO; return name;}

μία λίστα με πληροφορίες για φοιτητές

CIS	W7	Abramson,Paul B	CS3001	CS3071	CS3102	CS3132	CS3311	CS3322	CS3361	CS3900	EM2490
CE	X1	O'Rourke,Daniel M	CS3001	CS3041	CS3052	CS3071	CS3082	CS3111	CS3322	CS3900	EM2490
AI	Y6	Naismith,Gregory S	CS3001	CS3052	CS3071	CS3082	CS3311	CS3322	CS3361	CS3900	EM2490
CIS	X4	Sanders,Alexander P	CS3001	CS3071	CS3102	CS3132	CS3311	CS3322	CS3361	CS3900	EM2490
SI	__	Varney,Samantha	CS3041	CS3052	CS3071	CS3082	CS3251	CS3311	CS3432	CS3900	EM2490
CS	W1	Smith,Mark	CS3001	CS3052	CS3071	CS3082	CS3212	CS3232	CS3241	CS3251	CS3900 EM2490
MI	I3	Cooper,Paul	CS3311	CS3561	CS3572	CS3902	MK3322	HM3111	HM3132	HM3142	HM3152 HM3121
MI	I4	Smythe,Helen Ruth	CS3561	CS3572	CS3902	MK3322	MK3311	DI3111	HM3132	HM3152	HM3121 HS3132
CE	Z5	Grant,Ellie	CS3052	CS3071	CS3082	CS3111	CS3311	CS3322	CS3361	CS3900	EM2490

εναλλακτικά φτιάχνουμε κανόνα έτσι ώστε το δεύτερο γράμμα είναι απόστροφος ή μικρό:

[A-Z]+	{if(trace)ECHO; return school;}
[A-Z][-A-Za-z',]*	{if(trace)ECHO; return name;}

μία λίστα με πληροφορίες για φοιτητές

CIS	W7	Abramson,Paul B	CS3001	CS3071	CS3102	CS3132	CS3311	CS3322	CS3361	CS3900	EM2490
CE	X1	O'Rourke,Daniel M	CS3001	CS3041	CS3052	CS3071	CS3082	CS3111	CS3322	CS3900	EM2490
AI	Y6	Naismith,Gregory S	CS3001	CS3052	CS3071	CS3082	CS3311	CS3322	CS3361	CS3900	EM2490
CIS	X4	Sanders,Alexander P	CS3001	CS3071	CS3102	CS3132	CS3311	CS3322	CS3361	CS3900	EM2490
SI	__	Varney,Samantha	CS3041	CS3052	CS3071	CS3082	CS3251	CS3311	CS3432	CS3900	EM2490
CS	W1	Smith,Mark	CS3001	CS3052	CS3071	CS3082	CS3212	CS3232	CS3241	CS3251	CS3900 EM2490
MI	I3	Cooper,Paul	CS3311	CS3561	CS3572	CS3902	MK3322	HM3111	HM3132	HM3142	HM3152 HM3121
MI	I4	Smythe,Helen Ruth	CS3561	CS3572	CS3902	MK3322	MK3311	DI3111	HM3132	HM3152	HM3121 HS3132
CE	Z5	Grant,Ellie	CS3052	CS3071	CS3082	CS3111	CS3311	CS3322	CS3361	CS3900	EM2490

τέλος χρειαζόμαστε και κάτι για λευκούς και μη νόμιμους χαρακτήρες:

```
[ \t\n]                {if(trace)ECHO;}  
.  
                        {ECHO; yyerror("unexpected character");}
```


Παράδειγμα (κι άλλο) *calculator – yacc*

```
%{  
#include <stdio.h>  
  
int regs[26];  
int base;  
  
%}  
  
%start list  
  
%token DIGIT LETTER  
  
%left '|'   
%left '&'   
%left '+' '-'   
%left '*' '/' '%'   
%left UMINUS /*supplies precedence for unary minus */  
  
%%          /* beginning of rules section */
```

```
list:                                /*empty */
    |
    list stat '\n'
    |
    list error '\n'
    {
        yyerrok;
    }
    ;

stat:  expr
    {
        printf("%d\n", $1);
    }
    |
    LETTER '=' expr
    {
        regs[$1] = $3;
    }
    ;
```

<pre> expr: '(' expr ')' { \$\$ = \$2; } expr '*' expr { \$\$ = \$1 * \$3; } expr '/' expr { \$\$ = \$1 / \$3; } expr '%' expr { \$\$ = \$1 % \$3; } expr '+' expr { \$\$ = \$1 + \$3; } expr '-' expr { \$\$ = \$1 - \$3; } </pre>	<hr style="border: 2px solid blue;"/>	<pre> expr '&' expr { \$\$ = \$1 & \$3; } expr ' ' expr { \$\$ = \$1 \$3; } '-' expr %prec UMINUS { \$\$ = -\$2; } LETTER { \$\$ = regs[\$1]; } number ; </pre>
--	---------------------------------------	---

```
number: DIGIT
{
    $$ = $1;
    base = 10;
}
|
number DIGIT
{
    $$ = base * $1 + $2;
}
;
```

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

yyerror(s)
char *s;
{
    fprintf(stderr, "%s\n", s);
}

yywrap()
{
    return(1);
}
```

Λέξεις

```
%{  
#include "words2.tab.h"  
extern int yynval;  
extern int at_end;  
%}  
  
%%  
  
<<EOF>>      {  
                at_end = 1;  
                return END;  
                }  
  
" "  
\\n            |  
"\\t"          ;
```

Λέξεις

[Tt]he
[Aa]n? |
 |
 | return ARTICLE;

go(es)? |
jumps? |
runs? |
likes? |
eats? |
 | return VERB;

dogs? |
cats? |
fish |
fox(es)? |
moose |
 | return NOUN;

quick
slow
lazy
clever
smart
stupid
brown
black
blue
red
orange
white
big
small |
 | return ADJECTIVE;

quickly
easily
openly
slowly |
 | return ADVERB;

over
under
around
through
between |
 | return PREPOSITION;

.
%%

return yytext[0]; /* default action allows yacc to see literal characters */

Λέξεις

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

```
  
int at_end = 0;  
extern int yychar;  
%}
```

```
  
%token ARTICLE VERB NOUN ADJECTIVE ADVERB PREPOSITION END  
%start sentence
```

Λέξεις

```
%%
sentence:      nphrase VERB termunct      { printf ("Simple noun-verb sentence.\n");
                                              YYACCEPT;}
      |      nphrase VERB vmodifier termunct { printf ("Sentence with modified verb\n");
                                              YYACCEPT;}
      |      nphrase VERB nphrase termunct { printf ("Sentence with object\n");
                                              YYACCEPT;}
      |      END { printf ("Got EOF from lex.\n");
                  YYACCEPT;}

/* All these YYACCEPTS are needed so yyparse will return immediately,
   rather than waiting for the first token of the next sentence.  They
   wouldn't be necessary if the main program were only calling yyparse() once.  */

termunct:      ' ' ;
      |      '!' ;
      ;
```


Λέξεις

```
nphrase:      modifiednoun      ;
              | ARTICLE modifiednoun { printf ("\tGot an article\n"); }
              ;
modifiednoun: NOUN
              | nmodifier modifiednoun { printf ("\tmodified noun\n"); }
              ;
nmodifier:    ADJECTIVE          ;
              | ADVERB nmodifier { printf ("\tadded an adverb to a noun modifier\n"); }
              ;
vmodifier:    ADVERB             ;
              | ADVERB vmodifier { printf ("\tadded an adverb to a verb modifier\n"); }
              | PREPOSITION nphrase { printf ("\tprepositional phrase\n"); }
              ;
```

Λέξεις

```
%%  
int main (void) {  
    while (! at_end) {  
        yyparse();  
    }  
    printf ("Wasn't that fun?\n");  
}  
  
/* Added because panther doesn't have liby.a installed. */  
int yyerror (char *msg) {  
    return fprintf (stderr, "YACC: %s; yychar=%d\n", msg, yychar);  
}
```

union σε μικρό παράδειγμα - lex

```
%{
#include "y.tab.h"
%}

%%

program      { return PROGRAMTK; }
begin        { return BEGINTK; }
end           { return ENDTK; }
", "         { return COMMATK; }
declare      { return DECLARETK; }
enddeclare   { return ENDDECLARETK; }
[a-z]+       { strcpy(yylval.strV, yytext);
               return ID;
             }
"="          { return ASSIGNTK; }
"+"          { return PLUSTK; }
";"          { return SEMICOLUMNTK; }
\n           ;
.            ;
%%
```

union σε μικρό παράδειγμα - yacc

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

```
%union {  
    int intV;  
    char strV[30];  
};
```

```
%token ID PROGRAMTK DECLARETK ENDDECLARETK COMMATK BEGINTK ENDTK ASSIGNTK PLUSTK SEMICOLUMNTK  
%%
```

union σε μικρό παράδειγμα - yacc

```
program
  : PROGRAMTK ID programblock
  ;
programblock
  : DECLARETK vars ENDDECLARETK block      { printf("%d variables have been declared\n",counter); }
  ;
block
  : BEGINTK assignments ENDTK
  ;
vars
  : ID                                     { printf("addSymbolTable(%s)\n",$1.strV); counter++; }
  | vars COMMATK ID                       { printf("addSymbolTable(%s)\n",$3.strV); counter++; }
  ;
assignments
  : assignments assignment
  | assignment
  ;
assignment
  : ID ASSIGNTK expression SEMICOLUMNTK    { printf("%s = %s ;\n",$1.strV,$3.strV); }
  ;
expression
  : ID                                     { strcpy($$.strV,$1.strV); }
  | expression PLUSTK ID                  { strcat($$.strV," + ");strcat($$.strV,$3.strV); }
  ;
```

union σε μικρό παράδειγμα - yacc

```
%%  
int counter = 0;  
main()  
{  
    yyparse();  
    return 0;  
}
```

Ένα λίγο μεγαλύτερο - yacc

```
program
    : PROGRAMTK STRING programBlock
    ;

programBlock
    : declarations block
    ;

block
    : BEGINTK statements ENDTK;

declarations
    : DECLARETK vars ENDDECLARETK      { if (counter==1) printf ("%d variable has been
declared\n",counter);
    |                                     else printf ("%d variables have been declared\n",counter); }
    ;
```

Ένα λίγο μεγαλύτερο - yacc

```
vars
: STRING          { printf ("add %s to the symbol table\n",$1.strV); counter++;}
| vars COMMA TK STRING      { printf ("add %s to the symbol table\n",$3.strV); counter++;}
;

statements
: statement
| statements statement
;

statement
: assignment
| do_statement
;
```


Ένα λίγο μεγαλύτερο - yacc

```
assignment
: STRING ASSIGNMENTTK expression SEMICOLUMNTK { printf ("assign expression to variable %s, i.e. %s =
%s \n",$1.strV,$1.strV,$3.strV);
|
|
|
|
|
|
|
;

doTk      :
DOTTK      { printf("p1=nextquad()=...=200\n");
|           strcpy($$.strV,"200");
|           }
;

do_statement
: doTk statements WHILETK expression EQUALTK expression { printf("if exp1==expr2 jump at
%s\n",$1.strV); }
;
```

Ένα λίγο μεγαλύτερο - yacc

```
expression
: STRING                                { printf("create a new temporary variable e.g. T_e\n");
    printf("assign the variable %s to T_e, i.e. T_e=%s\n",$1.strV,$1.strV);
    strcpy($$.strV,"T_e");
    }
| expression PLUSTK STRING              { printf("accumulate the variable %s to T_e, i.e.
T_e=T_e+%s\n",$3.strV,$3.strV);
    strcpy($$.strV,$1.strV);
    }
;
```

Ένα λίγο μεγαλύτερο - yacc

```
%%  
  
int counter = 0 ;  
  
main()  
{  
    yyparse();  
    return 0;  
}
```

type checking - lex

```
%{
#include "y.tab.h"
%}

%%

[Ia-z]+ { strcpy(yylval.strV,yytext);
          printf("%s\n",yytext);
          return INTEGER_VAL;
        }

[Ra-z]+ { strcpy(yylval.strV,yytext);
          printf("%s\n",yytext);
          return REAL_VAL;
        }

[\\+\\-\\*\\/=] {
                printf("%s\n",yytext);
                return *yytext;
              }

.          ;

%%
```

type checking - yacc

```
%{  
#include <stdio.h>  
%}  
  
%union {  
    char strV[30];  
};  
  
%token INTEGER_VAL REAL_VAL  
  
%%
```

type checking - yacc

```
program : assign

number : INTEGER_VAL | REAL_VAL

assign : INTEGER_VAL '=' INTEGER_VAL '+' INTEGER_VAL
assign : REAL_VAL '=' number '+' number

assign : INTEGER_VAL '=' INTEGER_VAL '-' INTEGER_VAL
assign : REAL_VAL '=' number '-' number

assign : INTEGER_VAL '=' INTEGER_VAL '*' INTEGER_VAL
assign : REAL_VAL '=' number '*' number

assign : REAL_VAL '=' number '/' number
      {
      // gengquad ("=", "$5.strV", "0", runtime_error)
      }
```

type checking - yacc

```
%%  
  
main()  
{  
    yyparse();  
    return 0;  
}
```

type checking - 2 - yacc

```
%{  
#include <stdio.h>  
#include "defs.h"  
%}
```

```
%union {  
    char strV[30];  
    int type;  
};
```

```
%token ID
```

```
%%
```

type checking - 2 - yacc

```
program : assign

op      : '+'      { strcpy($$.strV, "+"); }
        | '-'      { strcpy($$.strV, "-"); }
        | '*'      { strcpy($$.strV, "*"); }
        | '/'      { strcpy($$.strV, "/"); }

assign  : ID '=' ID op ID
        { if (strcmp($4.strV, "/"))
          {
            if ($1.type==INTEGER_VAL &&
                ($3.type==REAL_VAL || $5.type==REAL_VAL))
                printf("error in typechecking %s\n", $4.strV);
          }
          else
            if ($1.type==INTEGER_VAL)
                printf("error in typechecking %s\n", $4.strV);
        }
```

two lists – def.h

```
typedef struct
{
    int data;
    struct list * next;
}list;

typedef struct
{
    list * a;
    list * b;
} twolists;

typedef twolists * twolistsP;
```

two lists – lex

```
%{
#include "defs.h"
#include "y.tab.h"
%}

%%

[A-Z|a-z]+ { return VAR;
            }

[\\+\\-\\*\\/=/ (/)] {
                    return *yytext;
                }

.           ;
\\n         ;
```

two lists – yacc

```
%{  
#include <stdio.h>  
#include "defs.h"  
  
twolistsP d;  
  
%}  
  
%union {  
    twolistsP q;  
};  
  
%token VAR
```

two lists – yacc

```
S:      expression
      {
          printf("%d\n", ($1.q) ->a->data);
          printf("%d\n", ($1.q) ->b->data);
      }
;

expression :    VAR
      {
          d = (twolistsP) malloc(sizeof(twolists));
          d->a = (list*) malloc(sizeof(list));
          d->b = (list*) malloc(sizeof(list));
          d -> a -> data = 1;
          d -> a -> next = NULL;
          d -> b -> data = 2;
          d -> b -> next = NULL;
          $$ .q = d;
      }
.
```

structs – lex

```
%{  
#include "yacc3.tab.h"  
%}  
%%  
BEGIN      {return BEGIN_T;}  
NORTH      {return NORTH_T;}  
SOUTH      {return SOUTH_T;}  
EAST       {return EAST_T;}  
WEST       {return WEST_T;}  
[ \n\t]    ;  
%%
```

structs – yacc

```
%union{
    struct {
        int x;
        int y;
    } loc;
    struct {
        int dx;
        int dy;
    } offset;
}
%token BEGIN_T EAST_T WEST_T NORTH_T SOUTH_T
%type <offset> instr
%type <loc> seq
%%
```


structs – yacc

```
robot    :      seq          {printf("Robot is at (%d,%d)\n",$1.x,$1.y);}
          ;

seq      :      seq instr    {$$.$x=$1.x+$2.dx; $$.$y=$1.y+$2.dy;}
          |      BEGIN_T     {$$.$x=0;$$.$y=0;}
          ;

instr    :      EAST_T       {$$.$dx=1; $$.$dy=0;}
          |      WEST_T      {$$.$dx=-1; $$.$dy=0;}
          |      SOUTH_T     {$$.$dx=0; $$.$dy=-1;}
          |      NORTH_T     {$$.$dx=0; $$.$dy=1;}
          ;

%%
%%
```

C grammar – lex and yacc

<https://www.lysator.liu.se/c/ANSI-C-grammar-l.html>

<https://www.lysator.liu.se/c/ANSI-C-grammar-y.html>

Ευχαριστώ !!!
