Ελληνικό Μεσογειακό Πανεπιστήμιο

Τμήμα Ηλεκτρονικών Μηχανικών

Αρχές Γλωσσών Προγρ/σμού & Μεταγλωττιστές

Κεφ. 6 - Λεκτικοί Αναλυτές

Δρ. Εμμανουήλ Σκουνάκης
Διδάκτωρ του Πανεπιστημίου Brunel του Λονδίνου
M.Sc. στιςΤηλεπικοινωνίες και Δίκτυα Η/Υ
M.Sc. Ηλεκτρονικού Μηχανικού και Μηχανικού Η/Υ

Χανιά, Κρήτη

Λεκτικοί Αναλυτές

Βασικές Γνώσεις

Οι Λεκτικοί Αναλυτές λειτουργούν ως σαρωτές (προγράμματα που αναγνωρίζουν λεξικογραφικά μοτίβα (patterns) σε ένα κείμενο και παράγουν λεξικογραφικούς αναλυτές σε γλώσσες C/C++, που αποτελούν την πρώτη φάση της ανάλυσης της μεταγλώττισης).

Η ιστορία στους Λεκτικούς Αναλυτές ξεκινάει με το Lex (μια κλασική γεννήτρια παραγωγής λεκτικών αναλυτών που συνεργάζεται με την γεννήτρια συντακτικών αναλυτών yacc). Η εξέλιξή του έφερε τον λεκτικό αναλυτή Flex.

Ο Flex γράφτηκε σε γλώσσα προγραμματισμού C από τον Vern Paxson το 1987.
Σήμερα υποστηρίζει την δημιουργία κώδικα σε C και C++ και συνεργάζεται με τη γεννήτρια συντακτικών αναλυτών **Bison**, για τη δημιουργία μεταγλωττιστών.

Ο Flex διαβάζει αρχεία εισόδου που έχουν δοθεί από το χρήστη ώστε να γίνει η περιγραφή του λεξικογραφικού αναλυτή που θα παραχθεί.
Η περιγραφή αποτελείται από ***Κανόνες*** και περιλαμβάνει συνήθως κώδικα σε εκδόσεις της C.
Το παραγόμενο από τον Flex είναι το αρχείο, ***lex.yy.c*** που καθορίζει τη ρουτίνα **yylex()**, τη βασική συνάρτηση της λεξικής ανάλυσης.

Το αρχείο αυτό μεταγλωττίζεται και δημιουργεί ένα εκτελέσιμο αρχείο, το οποίο όταν εκτελείται ελέγχει αν υπάρχουν κανονικές εκφράσεις.

Κανονικές Εκφράσεις ονομάζουμε τους κανόνες που θα χρησιμοποιηθούν για την αναγνώριση προτύπων χαρακτήρων.
Έτσι μπορούμε να ορίσουμε τις ακολουθίες χαρακτήρων που ανήκουν στη γλώσσα, δηλαδή τις **Λεκτικές Μονάδες** τις οποίες θέλουμε να αναγνωρίζει ο λεκτικός αναλυτής.

Λεκτικοί Αναλυτές

Γραμματική Κανονικών Εκφράσεων και Σάρωση

Οι σαρωτές αναγνωρίζουν κυρίως πρότυπα χαρακτήρων.

Για παράδειγμα σε ένα πρόγραμμα της C

- ένας ακέραιος αποτελείται από ένα ή το πολύ 5 αριθμητικά ψηφία (έως 32767).
- το όνομα μίας μεταβλητής αποτελείται από ένα γράμμα ακολουθούμενο από περισσότερα γράμματα ή ψηφία.
- οι διάφοροι τελεστές είναι μεμονωμένοι χαρακτήρες ή ζεύγη χαρακτήρων.

Έτσι, για να ορίσουμε τα πρότυπα χρησιμοποιούμε τις ***Κανονικές Εκφράσεις***.

Το πρόγραμμα Flex αποτελείται από μια λίστα *Κανονικών Εκφράσεων* μαζί με οδηγίες που καθορίζουν τι πρέπει να γίνει, όταν βρεθεί στην είσοδο μια ή περισσότερες από τις κανονικές εκφράσεις.

Μία **Κανονική Έκφραση** χρησιμοποιείται για να περιγράψει μία κανονική γλώσσα. Έτσι, αναπαριστά πρότυπα (συμβολοσειρές) που χρησιμοποιούνται σε μία ***Μεταγλώσσα*** (μια γλώσσα που χρησιμοποιεί κανονικούς χαρακτήρες κειμένου για να περιγράψει ένα πρότυπο που θέλουμε να ταιριαστεί).

Οι χαρακτήρες με ειδική σημασία στις Κανονικές Εκφράσεις σε μία **Μεταγλώσσα** είναι για παράδειγμα:

[0-9]	Όλοι οι αριθμητικοί χαρακτήρες [0123456789]
[a-z]	Όλοι οι μικροί χαρακτήρες της αγγλικής αλφαβήτου
[A-z]	Όλοι οι πεζοί και κεφαλαίοι χαρακτήρες
.	Όλοι οι χαρακτήρες, εκτός από το τέλος γραμμής (\n).
" "	Καθορίζουν την αρχή ή το τέλος μιας ακολουθίας χαρακτήρων. Για παράδειγμα το πρότυπο "abcd" αντιστοιχεί στην ακολουθία χαρακτήρων abcd. Επίσης τα χρησιμοποιούμε όταν δηλώνουμε τον κενό χαρακτήρα \t, η αλλαγή γραμμής \n, κα
^	Ταιριάζει τον πρώτο χαρακτήρα της κανονικής έκφρασης στην αρχή της γραμμής. Για παράδειγμα το πρότυπο ^a ταιριάζει την

Λεκτικοί Αναλυτές

	γραμμή που αρχίζει με τον χαρακτήρα a. Όταν όμως ο χαρακτήρας ^ περικλείεται σε αγκύλες η σημασία του αλλάζει, δηλώνει άρνηση. Το πρότυπο [^abc] ταιριάζει όλους τους υπόλοιπους χαρακτήρες, συμπεριλαμβανομένων και των ειδικών χαρακτήρων, εκτός από τους a,b,c.
$	Ταιριάζει τον τελευταίο χαρακτήρα της κανονικής έκφρασης στο τέλος της γραμμής. Για παράδειγμα η έκφραση ab$ μπορεί να ανιχνεύσει την ακολουθία ab που βρίσκεται στο τέλος της γραμμής (δηλαδή που ακολουθείται αλλαγή γραμμής \n).
\	Όταν θέλουμε να ταιριάξουμε ειδικούς χαρακτήρες όπως οι τελεστές σαν απλούς χαρακτήρες, χρησιμοποιούμε τον χαρακτήρα διαφυγής(\). Για παράδειγμα το πρότυπο abc\+ αντιστοιχεί στην ακολουθία abc+
+	Όταν θέλουμε να ορίσουμε πρότυπα που να ταυτίζονται με τουλάχιστον μία ή περισσότερες εμφανίσεις των αντίστοιχων εκφράσεων. Για παράδειγμα το a+ σημαίνει ένα ή περισσότερα a. Αντίστοιχα το πρότυπο [a-z]+ ταιριάζει όλες τις συμβολοσειρές που αποτελούνται από μικρά γράμματα.
?	Όταν θέλουμε να ορίσουμε πρότυπα που να ταυτίζονται με εκφράσεις ο τελεστής ? σημαίνει ότι στη θέση αυτή μπορεί να μπει οτιδήποτε ή τίποτα. Για παράδειγμα το πρότυπο ab?c μπορεί ταιριαστεί με τις συμβολοσειρές abc, ab2c, κλπ
I	Ορίζει την έννοια της εναλλαγής. Για παράδειγμα το πρότυπο ab I cd μπορεί να ταυτιστεί με το ab ή με το cd.
()	Οι παρενθέσεις τις χρησιμοποιούμε όταν θέλουμε να εκφράσουμε ομαδικότητα. Για παράδειγμα το πρότυπο (abc) θα ταυτιστεί με την συμβολοσειρά abc.
/	Η κάθετη μπάρα / ορίζει ακολουθούμενα συμφραζόμενα. Για παράδειγμα η έκφραση ab/cd μπορεί να ταυτιστεί με την συμβολοσειρά ab εφόσον ακολουθείται από την συμβολοσειρά cd.

Λεκτικοί Αναλυτές

r{i}	Αναγνωρίζει i ή περισσότερες εμφανίσεις της έκφρασης r. Για παράδειγμα το πρότυπο r{2} σημαίνει ότι η έκφραση r πρέπει να εμφανίζεται δύο φορές. Αντίστοιχα το πρότυπο r{2,3} αναγνωρίζει συμβολοσειρές που περιέχουν δύο ή τρεις φορές την έκφραση r.
EOF	Δηλώνει το τέλος του αρχείου (End of File)

Ταίριασμα και Επίλυση Ασαφών Προτύπων

Όταν ο παραγόμενος σαρωτής τρέχει, ανιχνεύει τα πρότυπα της βάσης του, που ταιριάζουν με την είσοδο.

Στην περίπτωση που αναγνωριστεί ένα πρότυπο, η ενέργεια που αντιστοιχεί σε αυτό εκτελείται.

Στη συνέχεια το υπόλοιπο μέρος της εισόδου ελέγχεται για άλλο ταίριασμα προτύπου.

Όταν βρεθεί ένα η περισσότερα πρότυπα που φαίνεται να ταιριάζουν στην ίδια είσοδο, τα πράγματα περιπλέκονται και δημιουργούνται ασάφειες.

Ο Flex επιλύει την ασάφεια με δύο απλούς κανόνες:

1. επιλέγεται το πρότυπο που έχει οριστεί πρώτο.
2. ταιριάζει το μακρύτερο πρόθεμα χαρακτήρων κάθε φορά.

Παράδειγμα, έστω ότι έχουμε τους παρακάτω κανόνες:

```
case          {return}
KEYWORD;
[a-z]+   {return IDENTIFIER;}
```

και ως είσοδο έχουμε τη λέξη cases.

Λεκτικοί Αναλυτές

Για το ταίριασμα θα προτιμηθεί ο δεύτερος κανόνας, γιατί ταυτίζεται με είσοδο μήκους 5, σε αντίθεση με τον πρώτο που ταυτίζεται με είσοδο μήκους 4.

Αντίθετα, αν η είσοδος ήταν case, θα ταυτιζόταν με τον πρώτο κανόνα, γιατί αυτός έχει οριστεί πρώτος.

Παράδειγμα Λεκτικής Ανάλυσης

Μετατροπή μιας έκφρασης από χαρακτήρες σε μια σειρά από tokens.

k = 1 – 4 * 8

Lex	Token
k	μεταβλητή
=	Εντολή ανάθεσης τιμής
1	Αριθμός
-	Πράξη αφαίρεσης
4	Αριθμός
*	Πράξη πολλαπλασιασμού
8	αριθμός

Λεκτικοί Αναλυτές

Η Λεκτική Ανάλυση

Ο μεταγλωττιστής δέχεται ως είσοδο μια σειρά από χαρακτήρες (αρχικό πρόγραμμα).

Κατά τη λεκτική ανάλυση γίνεται διαχωρισμός των χαρακτήρων σε ομάδες με συγκεκριμένη σημασία, σύμφωνα με τη γλώσσα προγραμματισμού που χρησιμοποιούμε.

Έτσι προκύπτουν οι **λεκτικές μονάδες (lexemes)** και σε κάθε μία από αυτές αντιστοιχεί ένα **αναγνωριστικό (token)**.

Έτσι, για παράδειγμα η παρακάτω εντολή σε γλώσσα C:

ust17 >= k + 33 * ust16

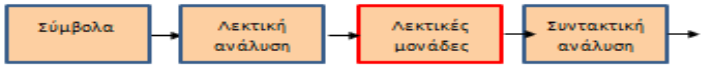
δίνει ως αποτέλεσμα τις παρακάτω λεκτικές μονάδες και τα αντίστοιχα αναγνωριστικά:

Λεκτική Μονάδα (Lexeme)	Αναγνωριστικό (Token)
ust17	Όνομα
>=	Τελεστής >=
k	Όνομα
+	Τελεστής +
33	Αριθμός
*	Τελεστής *
ust16	Όνομα

Βλέπουμε λοιπόν ότι κάθε λεκτική μονάδα αποτελείται από έναν ή περισσότερους χαρακτήρες.

Λεκτικοί Αναλυτές

Στάδια Μεταγλώττισης



Κατά τη Μεταγλώττιση,

με βάση τα **αποδεκτά σύμβολα** και τα **πρότυπα αναγνώρισης** της γλώσσας προγραμματισμού που χρησιμοποιούμε,

καταρχήν **η αναγνώριση των λεκτικών μονάδων (lexemes) και των αναγνωριστικών τους (tokens) σε μία ροή συμβόλων (συμβολοσειρά).**

Τα Πρότυπα Αναγνώρισης της γλώσσας

Τα Πρότυπα Αναγνώρισης (Paterns) συγκροτούν το λεξικό αναγνώρισης εκφράσεων της γλώσσας. π.χ.

1	Καταρχήν ένας λατινικός χαρακτήρας που μπορεί προαιρετικά να ακολουθείται από άλλους λατινικούς χαρακτήρες ή και αριθμούς	Όνομα
2	Μία σειρά από ψηφία μεταξύ 0 και 9	Αριθμός

Λεξήματα (Λεκτικές Μονάδες)

Ένας κώδικας εντολών αποτελείται από λεξήματα (lexemes), δηλαδή συμβολοσειρές που αντιστοιχούν σε πρότυπα αναγνώρισης της γλώσσας.

Η αναγνώριση των λεξημάτων και η αντιστοίχιση τους σε αναγνωριστικά είναι στην ουσία μία διαδικασία αναγνώρισης προτύπων.

Λεκτικοί Αναλυτές

π.χ.

Λέξημα	Πρότυπο αναγνώρισης	Αναγνωριστικό
if	Οι χαρακτήρες i και f	IF
-	Ο χαρακτήρας -	MINUS_OP
Metr44	Συμβολοσειρά που αρχίζει από λατινικό χαρακτήρα και ακολουθούν προαιρετικά περισσότερα γράμματα ή αριθμοί	ID
748	Σειρά Αριθμητικών συμβόλων (0-9)	NUM
<	Ο χαρακτήρας <	LT_OP
<=	Οι χαρακτήρες < και =	LE_OP
=	Ο χαρακτήρας =	EQ_OP
<>	Οι χαρακτήρες < και >	NE_OP
>	Ο χαρακτήρας >	GT_OP

Αντιστοίχιση Αναγνωριστικών (Tokens)

Έχοντας την εντολή **Metr44 = Metr44 + 1**, ο λεκτικός αναλυτής θα προσδιορίσει τα παρακάτω αναγνωριστικά:

Metr44	Όνομα	ID
=	Τελεστής =	EQ_OP
Metr44	Όνομα	ID
+	Τελεστής +	PLUS_OP
1	Αριθμός	NUM

Λεκτικοί Αναλυτές

Διαχωριστές

Διαχωριστές είναι οι χαρακτήρες που δεν μπορούν να αποτελούν μέρος μια λεκτικής μονάδας.

Στο παραπάνω παράδειγμα **Metr44 = Metr44 + 1** διαχωρίζεται το = ως άλλο αναγνωριστικό γιατί δεν μπορεί από τα πρότυπα να είναι μέρος του ονόματος Metr44.

Το ίδιο συμβαίνει και με το + μετά από το Metr44 όπως και με το 1 μετά από το +.

Οι χαρακτήρες αυτοί δεν μπορούν να αποτελούν (λόγω προτύπων) μέρος των λεκτικών μονάδων που προηγούνται.

FLEX – Γεννήτρια Λεκτικών Αναλυτών
Δομή Προγράμματος

```
%{
/* 1ο Μέρος - Τμήμα Ορισμών */
Μπορεί να περιλαμβάνει σε γλώσσα C μέσα σε %{ και %})
    - Σχόλια
    - Κώδικα (συνήθως δηλώσεις βιβλιοθηκών)
    - Αρχικές τιμές μεταβλητών
Και επιπλέον
   - Ονόματα εκφράσεων (π.χ. void Error (const char msg[]);)
%}
/* 2ο Μέρος – Τμήμα Κανόνων */
 - εντολές σε γλώσσα C
   (είναι συνήθως της μορφής → έκφραση ενέργεια)
- Διαβάζεται το μεγαλύτερο δυνατό μήκος της συμβολοσειράς
εισόδου και εκτελείται η ενέργεια που ορίζεται.
```

Λεκτικοί Αναλυτές

Εκφράσεις παραδείγματα:

a	Ο χαρακτήρας a
.	Οποιοσδήποτε χαρακτήρας εκτός της αλλαγής γραμμής
"abc"	Η συμβολοσειρά abc
[abc]	Ένας από τους χαρακτήρες a,b ή c
[a-z]	Ένας από τους χαρακτήρες από a έως z
[^a-f]	Ένας από τους χαρακτήρες εκτός περιοχής από a έως f

```
%%

/* 3ο Μέρος – Τμήμα Υπορουτίνων*/

π.χ.
void ERROR (const char msg [])
{
fprintf(stderr, "ERROR: %s\n", msg);
exit(1);
}
%%

Κώδικας Χρήστη

π.χ.
int main ()
{
....
return 0;
}
```

Λεκτικοί Αναλυτές

Παραδείγματα

Κατασκευή Λεκτικού Αναλυτή με τη Χρήση του FLEX

Πραγματοποιείται συνήθως με τη χρήση του μεταεργαλείου Flex (Γεννήτρια Λεκτικών Αναλυτών).

Είσοδος	• Μεταπρόγραμμα που περιέχει λεκτικές μονάδες.
Έξοδος	• Πρόγραμμα σε C • Επιστρέφει τον κωδικό της λεκτικής μονάδας που αναγνωρίστηκε ή 0 (*/n) στο τέλος της συμβολοσειράς εισόδου.

Η συνάρτηση **yylex** καλεί τον λεκτικό αναλυτή.

Η μεταβλητή **yytext** αποθηκεύει την τρέχουσα ακολουθία των χαρακτήρων.

1ο Παράδειγμα

```
%{
        /* πρόγραμμα αρίθμησης των γραμμών ενός κειμένου */
        #include <stdio.h>
        int linenum = 1;
%}
//.....
line .*\n
-------------------------------------------------------------------------------
%%
{line} {printf("%d %s", linenum++, yytext);}
%%
-------------------------------------------------------------------------------
void main()
{
//.....
yylex();
}
```

Λεκτικοί Αναλυτές

Παραδείγματα

2ο Παράδειγμα

```
%{
/* πρόγραμμα εκτύπωσης γραμμών που ξεκινούν ή καταλήγουν στο a */
#include <stdio.h>
%}
begins_with_a      a.*\n
ends_with_a        .a*\n
%%
{begins_with_a}    {printf(" %s", yytext);};
{ends_with_a}      {printf(" %s", yytext);};
.*\n;
%%
void main()
{ yylex();}
```