



Μεταγλωττιστές 2018 Προγραμματιστική Εργασία #2

Ονοματεπώνυμο : Μαγουνάκη Ουρανία

A.M: Π2015140

Η παρούσα εργασία αφορά την κατασκευή top-down αναλυτή με την μέθοδο της αναδρομικής κατάβασης, ο οποίος αναγνωρίζει εντολές για την ανάθεση σε μεταβλητές και εκτύπωση λογικών εκφράσεων οι οποίες περιέχουν λογικές εκφράσεις, αναγνωριστικά ονόματα μεταβλητών και λογικούς τελεστές.

Η ζητούμενη γραμματική η οποία κατασκευάστηκε αρχικά, ήταν η δημιουργία κανόνων για κάθε τελεστή καθώς έχουν διαφορετική προτεραιότητα. Όμως δεν κατάφερα να δημιουργήσω τον κώδικα ο οποίος αντιστοιχεί στην συγκεκριμένη γραμματική καθώς παρουσιάστηκαν αρκετά προβλήματα στην φάση της υλοποίησης. Συνεπώς, η γραμματική που δημιουργήθηκε είναι η εξής :

```
Stmt_list -> Stmt Stmt_list|.
Stmt -> id assign Expr|print Expr.
Expr -> Term Term_tail.
Term_tail -> Or_And_op Term Term_tail|.
Term -> Factor Factor_tail.
Factor_tail -> Not_op Factor Factor_tail|.
Factor -> (Expr) |Boolconst |id|.
Or_And_op -> or|not.
Not_op -> not.
Boolconst -> true|false|t|f|0|1.
```

Όσον αφορά τον τελευταίο κανόνα (**Boolconst**) δεν δημιουργήθηκε αντίστοιχη συνάρτηση, τα **True** , **False** ορίζονται ως **tokens**. Τονίζεται ότι χρησιμοποιείται **plex.NoCase** προκειμένου να ληφθούν υπ' όψη κεφαλαία και μικρά.

Παρακάτω παρουσιάζεται σχετική εικόνα με τα αποτελέσματα ελέγχου για LL(1) συμβατότητα καθώς και ο πίνακας που περιέχει τα FIRST και FOLLOW sets.

```

Grammar
Stmt_list → Stmt Stmt_list
          | .
Stmt →    id assign Expr
          | print Expr.
Expr →    Term Term_tail.
Term_tail → Or_And_op Term Term_tail
           | .
Term →     Factor Factor_tail.
Factor_tail → Not_op Factor Factor_tail
            | .
Factor →    (Expr)
          | Boolconst
          | id
          | .
Or_And_op → or
          | not.
Not_op →    not.
Boolconst → true
          | false
          | t
          | f
          | 0
          | 1.

```

Some sentences generated by this grammar: (t, print, print f, print t, print 0, print 1, print id, id assign, print true, id assign 0, id assign f, id assign 1, id assign t, print false, print (Expr), id assign id, id assign true, id assign false, id assign (Expr), id assign (Expr) not (Expr))

- All nonterminals are reachable and realizable.
- The nullable nonterminals are: Stmt_list Term_tail Factor_tail Factor Term Expr.
- The endable nonterminals are: Boolconst Not_op Factor_tail Factor Or_And_op Term_tail Term Expr Stmt_list Stmt.
- No cycles.

nonterminal	first set	follow set	nullable	endable
Stmt_list	id print	∅	yes	yes
Stmt	id print	id print	no	yes
Expr	(Expr) id or true false t f 0 1 not	id print	yes	yes
Term_tail	or not	id print	yes	yes
Term	(Expr) id true false t f 0 1 not	or not id print	yes	yes
Factor_tail	not	or not id print	yes	yes
Factor	(Expr) id true false t f 0 1	or not id print	yes	yes
AndOr_op	or not	(Expr) true false t f 0 1 or not id print	no	yes
Not_op	not	(Expr) true false t f 0 1 or not id print	no	yes
Boolconst	true false t f 0 1	or not id print	no	yes

Συνοπτική περιγραφή κώδικα

Επισημαίνεται ότι ο κώδικας στον οποίο βασίστηκε είναι από αντίστοιχο παράδειγμα που υλοποιήθηκε κατά την διάρκεια του εργαστηρίου.

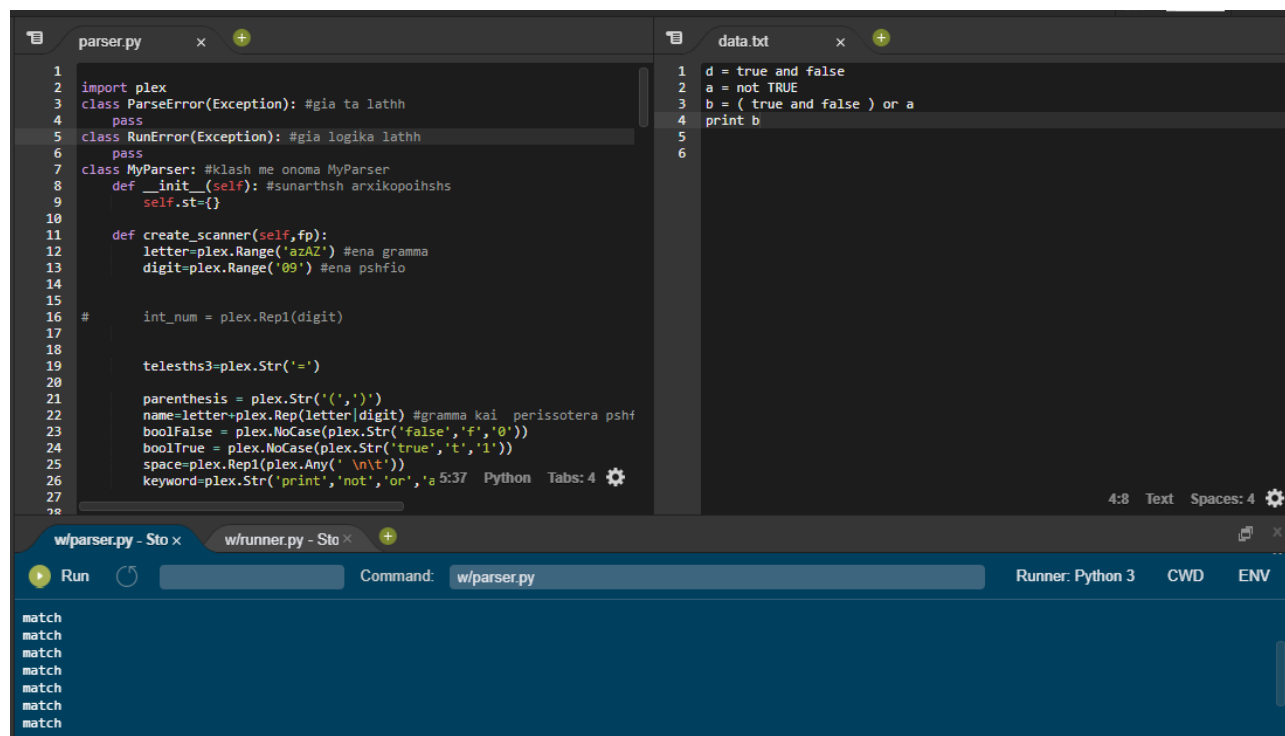
Όσον αφορά τον parser χρησιμοποιεί το *module plex*. Αναλυτικότερα, αρχικά έχει δημιουργηθεί η κλάση *ParseError* η οποία εκτυπώνει μηνύματα λάθους. Η κλάση *MyParser* περιέχει μία συνάρτηση αρχικοποίησης *init*, το *create_scanner* εντός του οποίου ορίζονται τα γράμματα (*letters*), τα ψηφία (*digits*), ο τελεστής ισότητας, οι παρενθέσεις, τα ονόματα μεταβλητών, οι λογικές τιμές (*boolFalse*, *boolTrue*), τα κενά (*spaces*), τα απαραίτητα *keywords* (*print*, *not*, *and*, *or*) και η αφαίρεση των σχολίων.

Έπειτα, ορίζεται το λεξικό με τα απαραίτητα *tokens* ενώ το *scanner* προσπελαύνει το αρχείο και δημιουργούμε τις μεταβλητές *self.la* και *self.val*. Η συνάρτηση *match()* ελέγχει τα *tokens* εάν υπάρχουν, ώστε να γίνει η αντιστοίχιση και στην συνέχεια ορίζονται οι συναρτήσεις που αντιστοιχούν στους κανόνες της γραμματικής βάση των *first* και *follow*.

Εν κατακλείδι, δημιουργείται ένα αντικείμενο της κλάσης *MyParser* προκειμένου να γίνει η προσπέλαση του αρχείου.

Ωστόσο, επισημαίνεται ότι κατά την εκτέλεση του *parser* παρατηρήθηκε ότι δεν αντιστοιχεί όλα τα *tokens* καθώς και ότι δεν υλοποιήθηκε ο *runner*.

Ορισμένα αποτελέσματα έπειτα από την εισαγωγή διάφορων συνδυασμών εισόδου:



```
1 import plex
2 class ParseError(Exception): #για τα lathh
3     pass
4 class RunError(Exception): #για logika lathh
5     pass
6 class MyParser: #klash me onoma MyParser
7     def __init__(self): #sunarthsh arxikopoihshs
8         self.st={}
9
10    def create_scanner(self,fp):
11        letter=plex.Range('azAZ') #ena gramma
12        digit=plex.Range('09') #ena pshfio
13
14        #
15        int_num = plex.Repl(digit)
16
17        telesths3=plex.Str('=')
18
19        parenthesis = plex.Str('(')
20        name=letter+plex.Rep(letter|digit) #gramma kai perissotera pshf
21        boolFalse = plex.NoCase(plex.Str('false','f','0'))
22        boolTrue = plex.NoCase(plex.Str('true','t','1'))
23        space=plex.Repl(plex.Any(' \n\t'))
24        keyword=plex.Str('print','not','or','e 5:37 Python Tabs: 4
25
26
27
28
```

```
1 d = true and false
2 a = not TRUE
3 b = ( true and false ) or a
4 print b
5
6
```

```
match
match
match
match
match
match
match
```

Πηγές

<http://smlweb.cpsc.ucalgary.ca/>

Scanning με το Plex Compiler Lecture Notes 1.pdf, "Ανοικτά Ακαδημαϊκά Μαθήματα στο Ιόνιο Πανεπιστήμιο"

<https://opencourses.ionio.gr/modules/document/file.php/DD1154/Scanning%20με%20το%20Plex%20%20Compiler%20Lecture%20Notes%201.pdf>