Γραμματική λογικών εκφράσεων

(εξάσκηση στη συντακτική ανάλυση, δημιουργία/εκτέλεση AST)

Στόχος

- Κατασκευή γραμματικής λογικών εκφράσεων
  - Για χρήση σε συνθήκες (conditions)
- Ζητούμενο
  - Δυνατότητα συνδυασμών με λογικούς τελεστές
    - and, or, not
    - Με τη συνήθη προτεραιότητα
    - Με αποφυγή αχρείαστων υπολογισμών (short-circuiting)

Μέρη λογικής έκφρασης

- Συγκρίσεις
  - number relop number (relop = όλοι οι γνωστοί τελεστές σύγκρισης: ==, !=, >, >=, <, <=)
  - Αργότερα το number θα αντικατασταθεί από αριθμητικές εκφράσεις
- Boolean Σταθερές
  - true, false
- Άλλες λογικές εκφράσεις σε παρένθεση

Αρχική γραμματική

Κανόνες	
Lexpr	→ Lterm Ltermtail
Ltermtail	→ or Lterm Ltermtail | ε
Lterm	→ Lfactor Lfactortail
Lfactortail	→ and Lfactor Lfactortail | ε
Lfactor	→ Latom | not Lfactor
Latom	→ true | false | number Relop number | lparen Lexpr rparen
Relop	→ eq | ne | gt | ge | lt | le

Βήμα 1ο

- Βεβαιωθείτε ότι η γραμματική είναι LL(1)
- Βρείτε τα FIRST και FOLLOW sets
- Μπορείτε να συμβουλευτείτε κάποιο on-line εργαλείο, όπως το:

  http://smlweb.cpsc.ucalgary.ca/start.html

Βήμα 2ο

- Χρησιμοποιήστε το notebook-υπόδειγμα

  https://mixstef.github.io/courses/compilers/parser-ast-builder-interpreter-template.ipynb

- Βρείτε τα τερματικά σύμβολα που απαιτούνται και κατασκευάστε το λεξικό του λεκτικού αναλυτή (plex)

Βήμα 3ο

- Κατασκευάστε τον συντακτικό αναλυτή
  - Συναρτήσεις μη τερματικών συμβόλων, σύμφωνα με τη γραμματική και τα FIRST/FOLLOW sets
  - Απλή συντακτική ανάλυση, όχι παραγωγή AST στο βήμα αυτό
- Δοκιμάστε την ορθή λειτουργία με την είσοδο:

  (3>2.45 or not (4!=8.2)) and true or (false) or 7<2.0

Βήμα 4ο

- Προσθέστε στον κώδικα του συντακτικού αναλυτή την παραγωγή του AST
  - Αποφασίστε για τα είδη κόμβων στο AST
    - Θα πρέπει να διευκολύνεται η ζητούμενη λειτουργία (υπολογισμός λογικής έκφρασης)
    - Λάβετε υπ’ όψη ότι αργότερα θα ενοποιηθεί με το AST των αριθμητικών εκφράσεων που έχετε ήδη
    - Μπορείτε να σκεφτείτε λογικές βελτιστοποιήσεις;
      - not not X → X, not true → false, not false → true
      - true and X → X, false and X → false
      - true or X → true, false or X → X

Βήμα 5ο

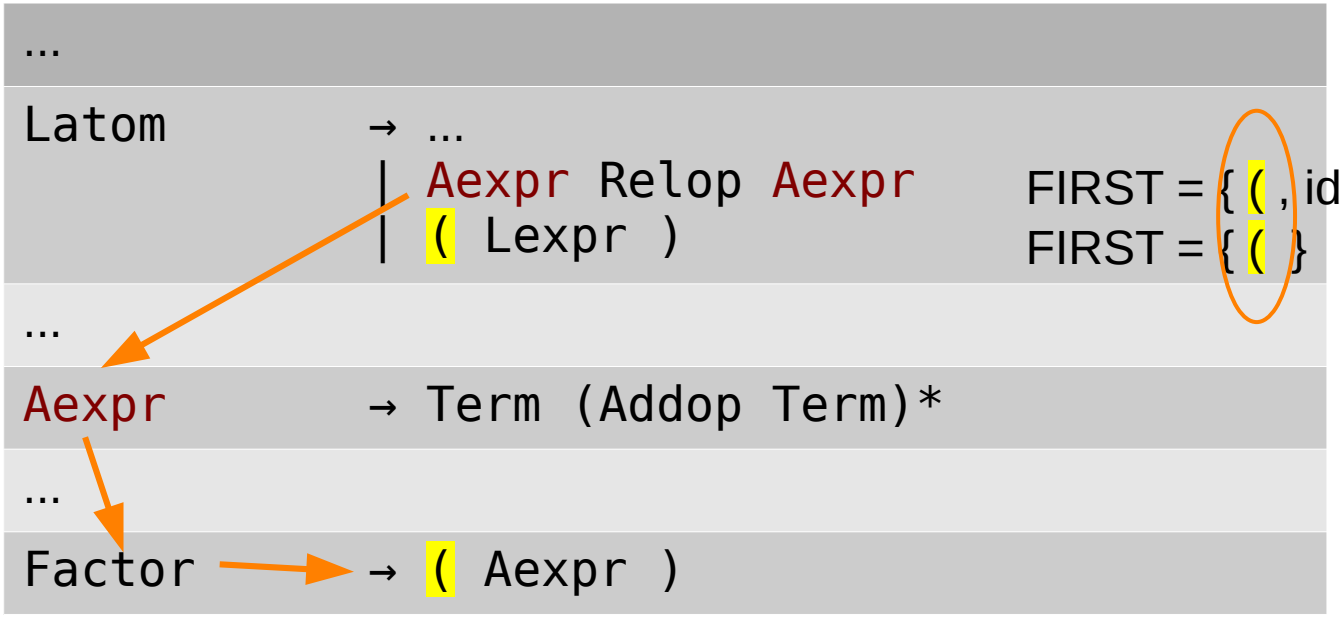
- Προσθέστε τον κώδικα στον interpreter για τον υπολογισμό της boolean τιμής (True, False) του AST
  - Τυπώστε ό,τι επιστρέφει η μέθοδος run()
  - Στον υπολογισμό των and, or μην εκτελείτε αχρείαστους υπολογισμούς
    - Π.χ. στο A and B, εάν το A είναι False δεν χρειάζεται να υπολογίσετε το B (short-circuiting)

Ενσωμάτωση αριθμητικών εκφράσεων

Latom	→ true | false | Aexpr Relop Aexpr | lparen Lexpr rparen
Relop	→ eq | ne | gt | ge | lt | le
Aexpr	→ Term (Addop Term)*
…	

- Η αρχική ιδέα: αντικατάσταση των αριθμών (number) με αριθμητικές εκφράσεις (Aexpr) στους τελεστές σύγκρισης + προσθήκη υπόλοιπων κανόνων γραμματικής αριθμητικών εκφράσεων
  - Θα δουλέψει;

Σύγκρουση FIRST sets



- Πρακτικά: όταν είμαστε στο Latom και το επόμενο token είναι το (τί διαλέγουμε;
  - Ανοίγει παρένθεση λογικής ή αριθμητικής έκφρασης;
  - Αν αποφασίσουμε λάθος δεν μπορούμε να αλλάξουμε επιλογή...

Πότε θα ξέρουμε ποιο είναι το σωστό;

```
(((…..(((5 + …
```

Μπορούμε με 1 lookahead token (ή γενικότερα με k tokens) να ξέρουμε αν η παρένθεση ανήκει σε αριθμητική ή λογική έκφραση;

- Εναλλακτικές λύσεις
  - Χρησιμοποιούμε διαφορετική μέθοδο συντακτικής ανάλυσης
    - Όχι top-down LL(1), όχι αναδρομικής κατάβασης → άλλο εργαλείο
    - Επίτρεψη ελεγχόμενου backtracking → π.χ. Parsing Expression Grammars
  - Απλοποιούμε τη γραμματική: δεν διαχωρίζουμε λογικές από αριθμητικές εκφράσεις
    - Αυτό το επιτυγχάνουμε μέσω της σημασιολογικής ανάλυσης (χρήση πρόσθετης πληροφορίας κατά τη συντακτική ανάλυση)

Απλοποιημένη γραμματική

```
Latom       → true | false
            | Aexpr Aexpr_rest
Aexpr_rest  → Relop Aexpr | ε
…
Factor      → ( Lexpr )
            | id
            | number
```

- Το αρχικό μη τερματικό σύμβολο Lexpr καλύπτει και τις λογικές και τις αριθμητικές εκφράσεις

Διαδικασία

- Προσθέστε στον κώδικα των λογικών εκφράσεων τις αριθμητικές λειτουργίες από τα προηγούμενα εργαστήρια
  - Δοκιμάστε τη συντακτική ανάλυση, τη δημιουργία και τον υπολογισμό του AST με την έκφραση

    (3>(2.45*1.5-128) or not ((8+0.4/2)!=8.2)) and true or (false) or 49/7<2.0
  - Τροποποιήστε τον κώδικα για να ελέγχετε μη αποδεκτές εκφράσεις όπως (3+1) and true
    - Προσθέστε ένα attribute σε κάθε ASTNode με το «είδος» του (εάν επιστρέφει αριθμητική ή boolean τιμή όταν υπολογίζεται)
    - Στους τελεστές ελέγξτε εάν τα subnodes τους έχουν την αναμενόμενη τιμή
      - boolean για τα and, or, not
      - αριθμητική για τα +, -, *, / και τελεστές σύγκρισης