Μεταγλωττιστές 2024-25

Εκτέλεση ενεργειών εφαρμογής
κατά τη διάρκεια της
Συντακτικής Ανάλυσης

Πού βρισκόμαστε τώρα

- Έχουμε κατασκευάσει έναν συντακτικό αναλυτή LL(1) με τη μέθοδο της «αναδρομικής κατάβασης»
- Για την ανάλυση της γραμματικής μια απλής «γλώσσας υπολογισμού αριθμητικών εκφράσεων»

```
a = 2 + 7.55*44
print a
b = 3*(a-99.01)
print b*0.23
c = 5-3-2
print c
```

Η γλώσσα υπολογισμού αριθμητικών εκφράσεων

- 2 εντολές (statements)

  `=` (assignment)

  `print` (output)

- 1 τύπος δεδομένων

  floats

- Χρήση απλών (scalar) μεταβλητών

  Δυναμικά, χωρίς δήλωση

  Σφαιρική εμβέλεια (global scope)

- Οι βασικοί αριθμητικοί τελεστές

  `+ - * /`

Εκτέλεση εντολών κατά τη διάρκεια της συντακτικής ανάλυσης

- Για την πολύ απλή γλώσσα των αριθμητικών εκφράσεων μπορούμε να εκτελούμε τις εντολές και τις πράξεις ταυτόχρονα με τη συντακτική ανάλυση
  - Διερμηνευτής εντολών (interpreter)
  - Η διαδικασία περιγράφεται από τον (γενικότερο) όρο Syntax Directed Translation
- Προσοχή: Η εκτέλεση εντολών κατά τη διάρκεια της συντακτικής ανάλυσης δεν είναι η ενδεδειγμένη όταν η γλώσσα είναι στοιχειωδώς πιο σύνθετη
  - Πώς εκτελούμε υπό συνθήκη;
    - Θα εκτελούμε τον κλάδο που είναι ανενεργός;
  - Πώς χειριζόμαστε τις δομές επανάληψης;
    - Θα αναλύουμε συντακτικά σε κάθε επανάληψη;
  - Τι γίνεται με τις συναρτήσεις;
    - Η συντακτική ανάλυση δεν πρέπει είναι κλήση της συνάρτησης
- Για τον λόγο αυτόν, μετά το απλό σημερινό παράδειγμα θα περάσουμε σε μορφές ενδιάμεσης αναπαράστασης και ξεχωριστής διερμηνείας

Η χρήση των μεταβλητών

- Ονόματα μεταβλητών μπορούν να εμφανιστούν
  - Στο αριστερό μέρος (lhs) μιας ανάθεσης (=)
  - Μέσα σε κάποια έκφραση (δεξιό μέρος, rhs)

    ```
    b = 3*(a-99.01)
    ```

    Τυπικά αυτό σημαίνει

    ```
    *(&b) = 3*(*&(a)-99.01)
    ```

- rhs: ανάγνωση της τιμής από τη διεύθυνση του δεσμευμένου χώρου της μεταβλητής (μια μορφή “dereferencing”)
- lhs: εγγραφή νέας τιμής στον δεσμευμένο χώρο της μεταβλητής

Οι μεταβλητές στην υλοποίησή μας

- Δυναμική χρήση μεταβλητών (χωρίς δήλωση, με έναν μόνο τύπο)
  - Θα τις τοποθετήσουμε σε ένα λεξικό (dict) της Python
    - Το ονομάζουμε symbol table (πίνακα συμβόλων)
    - Κλειδιά τα ονόματα των μεταβλητών, τιμές οι τρέχουσες τιμές των μεταβλητών

    ```
    b = 3*(a-99.01)
    ```

    Στην υλοποίησή μας σημαίνει

    ```
    symbol_table['b'] = 3*(symbol_table['a']-99.01)
    ```

- rhs: ανάγνωση της τιμής της μεταβλητής από το λεξικό
  - θα πρέπει να προηγηθεί έλεγχος ύπαρξης τιμής για τη μεταβλητή

    ```
    if 'b' in symbol_table:
        current_value = symbol_table['b']
    ```

- lhs: εγγραφή νέας τιμής της μεταβλητής στο λεξικό
  - είτε υπάρχει προηγούμενη τιμή είτε όχι

Μέθοδος υλοποίησης

- Η μέθοδος Expr() επιστρέφει την τελική τιμή κάθε αριθμητικής έκφρασης
  - Η εντολή ανάθεσης (=) τοποθετεί την τελική αυτή τιμή σε κάποια μεταβλητή
  - Η εντολή εξόδου (print) τυπώνει την τελική τιμή
- Οι υπόλοιπες μέθοδοι των μη τερματικών συμβόλων κάτω από την Expr() επιστρέφουν μέρος της αριθμητικής έκφρασης ή πληροφορία για τον υπολογισμό της
  - Factor()
  - Addop(), Multop()
  - Factor_tail(), Term_tail()
  - Term(), Expr()

Factor()

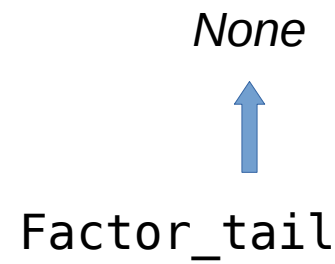
- Η μέθοδος Factor() επιστρέφει τις πρώτες αριθμητικές τιμές
  - Σε περίπτωση number επιστρέφει float(lexeme)
  - Σε περίπτωση id επιστρέφει την τρέχουσα τιμή της μεταβλητής από το symbol table
    - Εάν δεν υπάρχει η μεταβλητή στο symbol table θα πρέπει να παράγεται σφάλμα (runtime error, uninitialized variable)
  - Στην περίπτωση του (Expr) επιστρέφει ό,τι δίνει η κλήση της Expr()

Addop() και Multop()

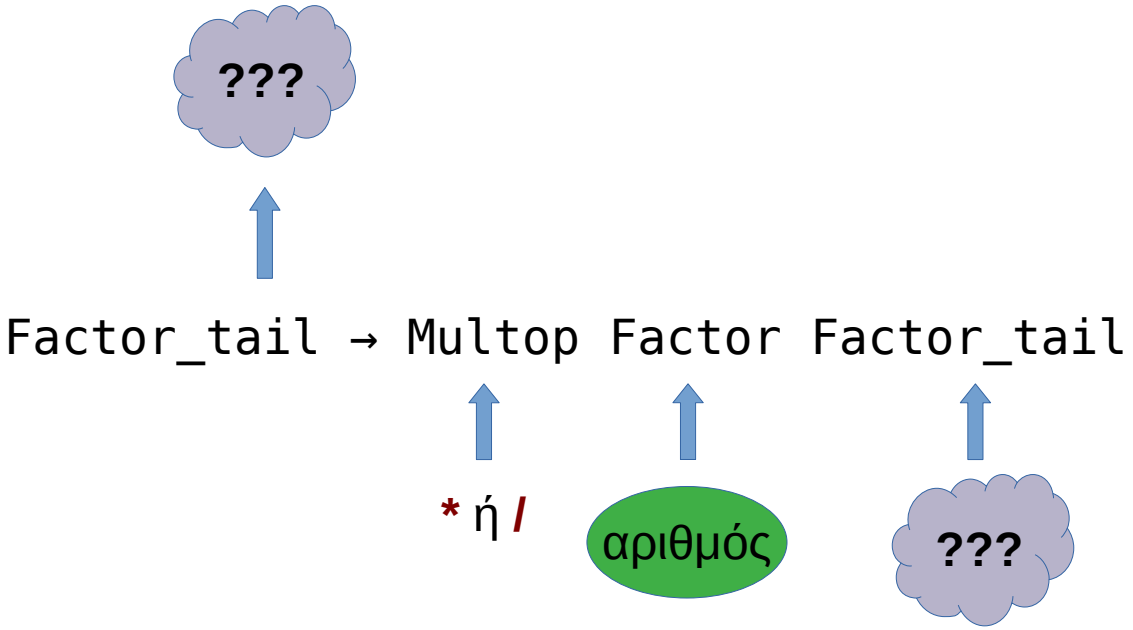
- Επιστρέφουν τον τελεστή που αναγνωρίστηκε
  - '+' ή '-' για την Addop()
  - '*' ή '/' για την Multop()
- Οι Addop() και Multop() δεν επιστρέφουν αριθμητική τιμή αλλά πληροφορία που θα χρειαστεί στις μεθόδους που τις έχουν καλέσει

Factor_tail()

- Η μέθοδος `Factor_tail()` επιστρέφει `None` στην περίπτωση της κενής παραγωγής

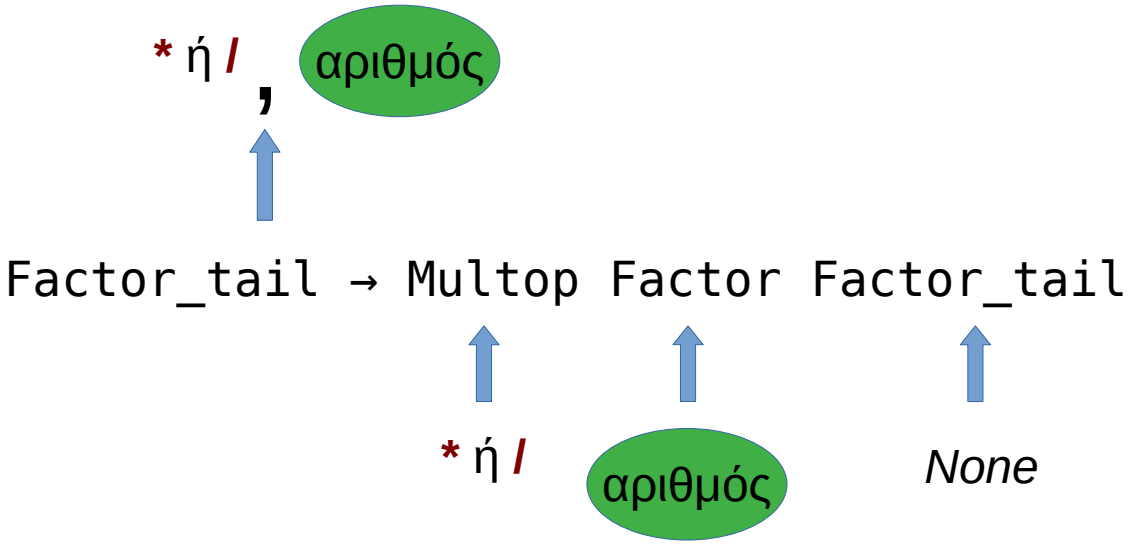


- Τι επιστρέφει όμως στην αντίθετη περίπτωση;



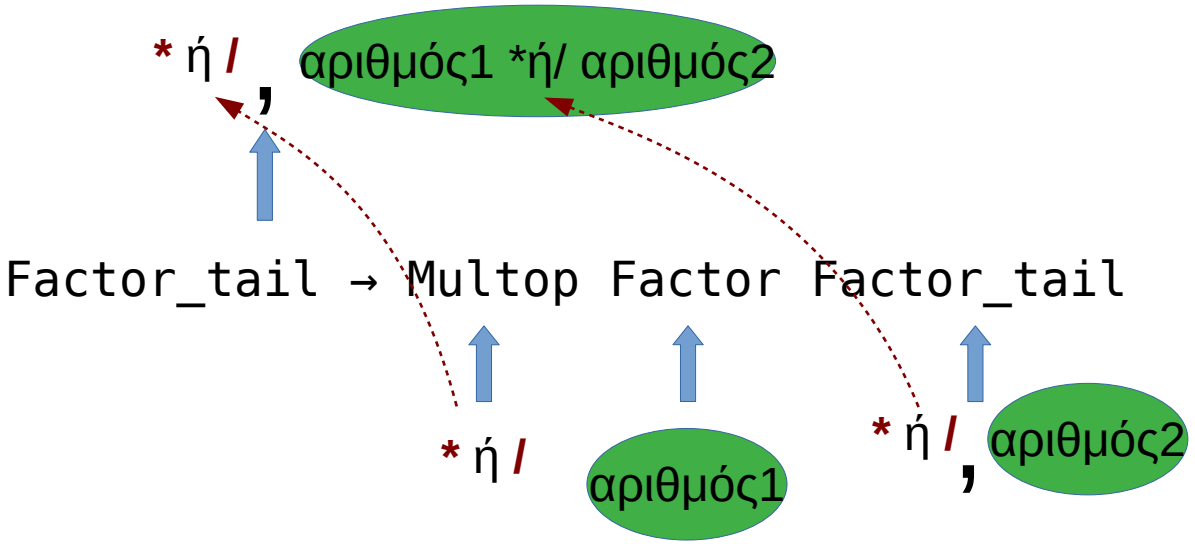
Factor_tail(): περίπτωση 1

- Περίπτωση 1: Η αναδρομική κλήση της `Factor_tail()` επιστρέφει `None`
- Η διαθέσιμη πληροφορία είναι ο τελεστής (* ή /) και το δεξιό μέρος της έκφρασης που εφαρμόζεται στον τελεστή
  - Λείπει το αριστερό μέρος
- Δεν μπορεί να γίνει κάποιος υπολογισμός, το ζευγάρι (τελεστής,δεξιό μέρος) επιστρέφεται ως έχει



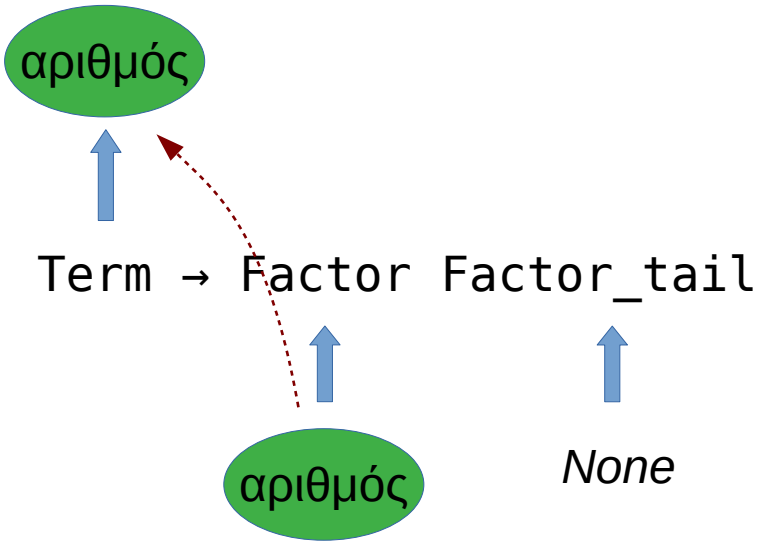
Factor_tail(): περίπτωση 2

- Περίπτωση 2: Η αναδρομική κλήση της `Factor_tail()` επιστρέφει ένα ζευγάρι (τελεστής2,αριθμός2)
- Υπολογίζεται η πράξη μεταξύ των αριθμών 1 και 2 με τον τελεστή2 και επιστρέφεται το ζευγάρι (τελεστής1, αριθμός1 *ή/ αριθμός2)
- Η `Term_tail()` υλοποιείται ακριβώς όπως η `Factor_tail()`, χρησιμοποιώντας όμως το + ή -



Term(): περίπτωση 1

- Περίπτωση 1: Η αναδρομική κλήση της `Factor_tail()` επιστρέφει `None`
- Η `Term()` επιστρέφει τον αριθμό που έλαβε μέσω της κλήσης της `Factor()`



Term(): περίπτωση 2

- Περίπτωση 2: Η αναδρομική κλήση της `Factor_tail()` επιστρέφει ένα ζευγάρι (τελεστής,αριθμός2)
- Υπολογίζεται και επιστρέφεται το αποτέλεσμα της πράξης μεταξύ των αριθμών 1 και 2 με τον τελεστή
- Η `Expr()` υλοποιείται ακριβώς όπως η `Term()`, χρησιμοποιώντας αυτή τη φορά το `+` ή `-`