Μεταγλωττιστές 2022-23

Συντακτική ανάλυση top-down LL(1)

Συντακτική ανάλυση top-down LL(1)

- Για μια σημαντική υποκατηγορία context-free γραμματικών με πολυπλοκότητα O(n)
  - Σάρωση εισόδου από αριστερά προς τα δεξιά (left-to-right)
  - Αριστερότερες ακολουθίες παραγωγής (leftmost derivations)
  - Συμβουλευόμενοι ένα (το επόμενο) token εισόδου σε κάθε βήμα

Προϋποθέσεις για αριστερότερες παραγωγές

- Δεν πρέπει οι κανόνες να έχουν αριστερή αναδρομή

  `Expr → Expr + Term`

  - Ο συντακτικός αναλυτής θα πέσει σε άπειρη αναδρομή
  - Απαλείφεται με αλγοριθμικό τρόπο, π.χ. ο κανόνας

    `A → A α | β`

    (α, β ακολουθίες οποιωνδήποτε συμβόλων) γίνεται

    `A → β A'`

    `A'→ α A' | ε`

  - Προσοχή: πρέπει να γίνει και σε περιπτώσεις έμμεσης αναδρομής (αλυσιδωτά, μετά από περισσότερα από ένα βήματα/κανόνες)

Προϋποθέσεις για αριστερότερες παραγωγές

- Δεν πρέπει να υπάρχει κοινός παράγοντας στην αρχή του δεξιού μέρους ενός κανόνα, π.χ.

```
Factor → id | id [ ExprList ] | id ( ExprList )
```

- Μετασχηματισμός με προσθήκη ενδιάμεσου κανόνα

```
Factor → id Args

Arg    → ε | [ ExprList ] | ( ExprList )
```

Υλοποίηση συντακτικού αναλυτή LL(1)

- Από τη Θεωρία Υπολογισμού έχουμε ένα θεωρητικό εργαλείο
  - Pushdown Automaton (PDA)
    - Αυτόματο ενισχυμένο με μια στοίβα (stack)
  - Για κάθε μετάβαση το αυτόματο συμβουλεύεται το σύμβολο εισόδου και την κορυφή της στοίβας
    - Για να αποφασίσει ποια θα είναι η επόμενη κατάσταση
    - Και αν/τι θα μπει στη στοίβα
  - Υπάρχει πάντα ένα αυτόματο PDA ισοδύναμο με μια γραμματική χωρίς συμφραζόμενα (CFG)
    - Για τις γραμματικές LL(1) το αυτόματο PDA είναι αιτιοκρατικό (ντετερμινιστικό)
      - Είναι δυνατή το πολύ μια μετάβαση

Πώς χρησιμοποιείται το αυτόματο PDA στη συντακτική ανάλυση

- Αρχικά στη στοίβα υπάρχει το αρχικό μη τερματικό σύμβολο της γραμματικής
- Το αυτόματο επαναλαμβάνει:
  - (λειτουργία predict) Αν στην κορυφή της στοίβας βρίσκεται μη τερματικό σύμβολο A , τότε το A αντικαθίσταται στη στοίβα από το δεξιό μέρος ενός κατάλληλου κανόνα της γραμματικής, σύμφωνα και με το επόμενο σύμβολο στην είσοδο
  - (λειτουργία match) Αν στην κορυφή της στοίβας βρίσκεται τερματικό σύμβολο a , τότε, αν το επόμενο σύμβολο εισόδου είναι ίδιο με το a, το a αφαιρείται από τη στοίβα
    - Σε κάθε άλλη περίπτωση παράγεται σφάλμα

Απλή (μη ρεαλιστική) περίπτωση LL(1)

- Έστω η γραμματική

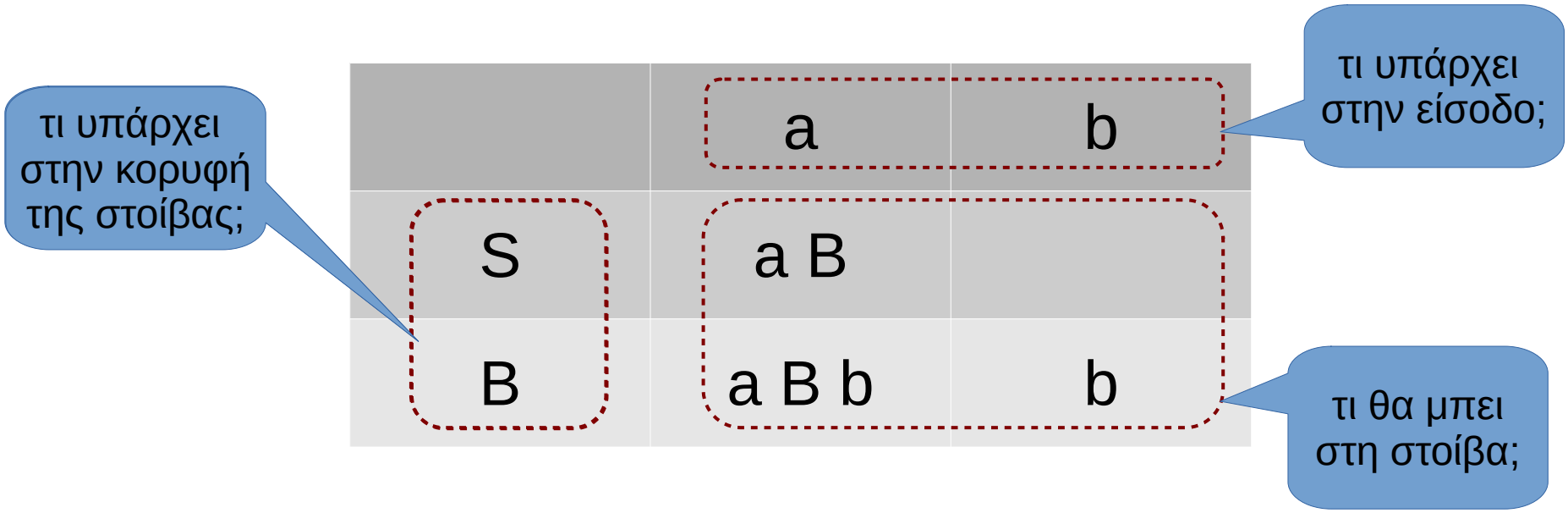
  S → a B

  B → b | a B b

- Όλα τα δεξιά μέρη των κανόνων ξεκινούν με τερματικό σύμβολο
  - διαφορετικό για τους εναλλακτικούς κανόνες (παραγωγές) του ίδιου μη τερματικού συμβόλου
    - π.χ. B → b και B → a B b

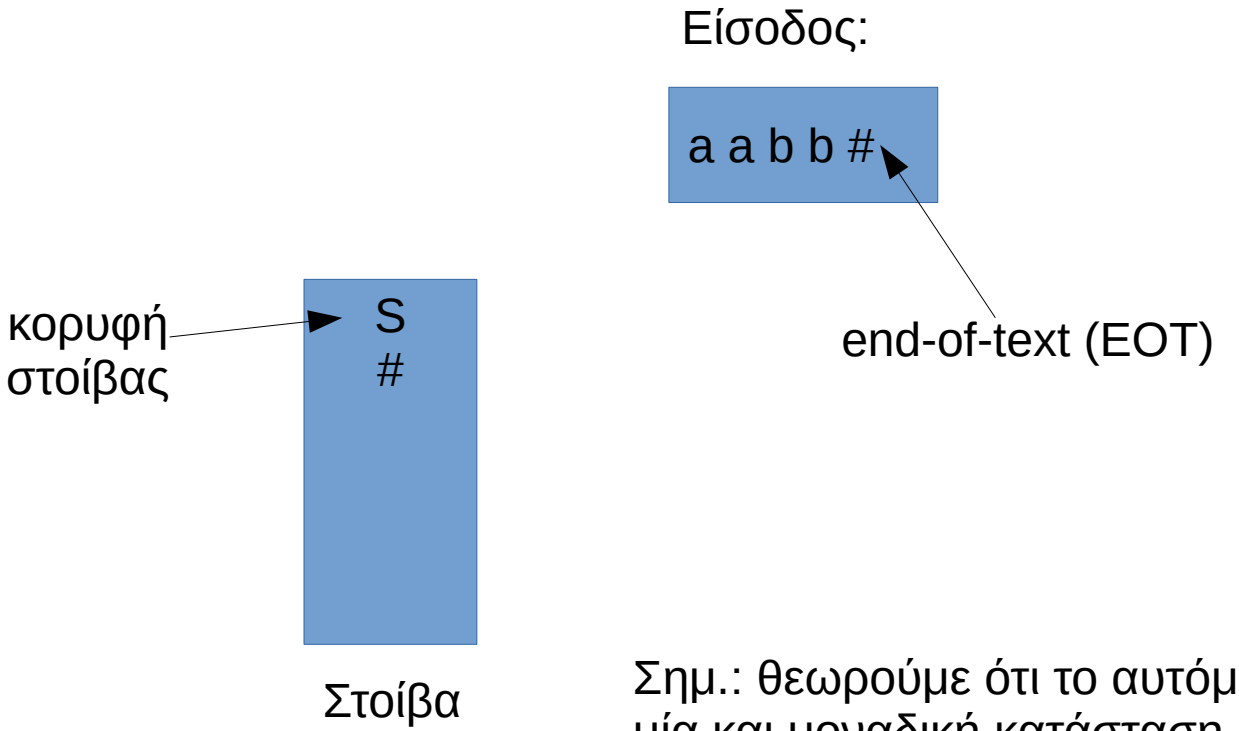
Πίνακας συντακτικής ανάλυσης

- Θεωρητικό εργαλείο, μέρος του αυτομάτου PDA
  - μπορεί επίσης να χρησιμοποιηθεί και σε έναν πραγματικό συντακτικό αναλυτή
  - επιλέγει τι θα μπει στη στοίβα στο βήμα predict



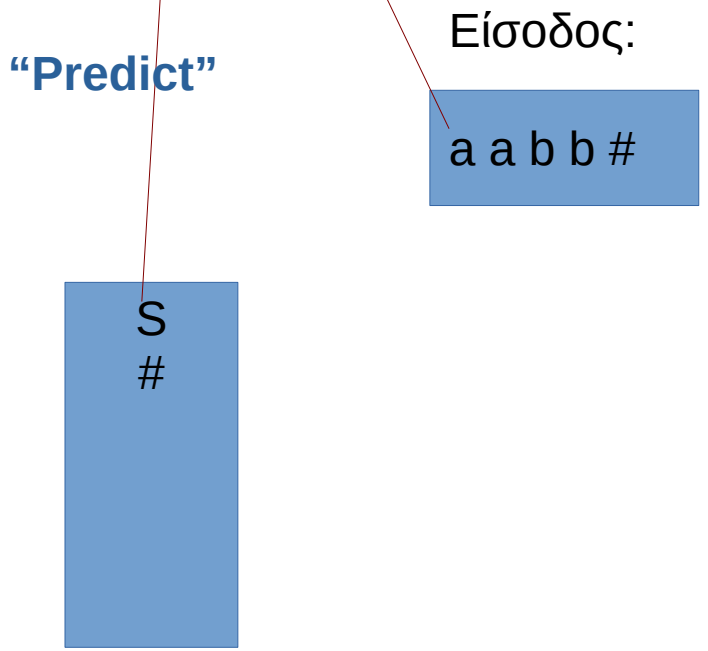
Π.χ. αν στην κορυφή της στοίβας είναι το S και το επόμενο σύμβολο στην είσοδο είναι το a, το S θα αντικατασταθεί στη στοίβα από το a B

	a	b
S	a B	
B	a B b	b



Σημ.: θεωρούμε ότι το αυτόματο έχει μία και μοναδική κατάσταση, στην οποία βρίσκεται πάντα

	a	b
S	a B	
B	a B b	b



	a	b
S	a B	
B	a B b	b

“Predict”

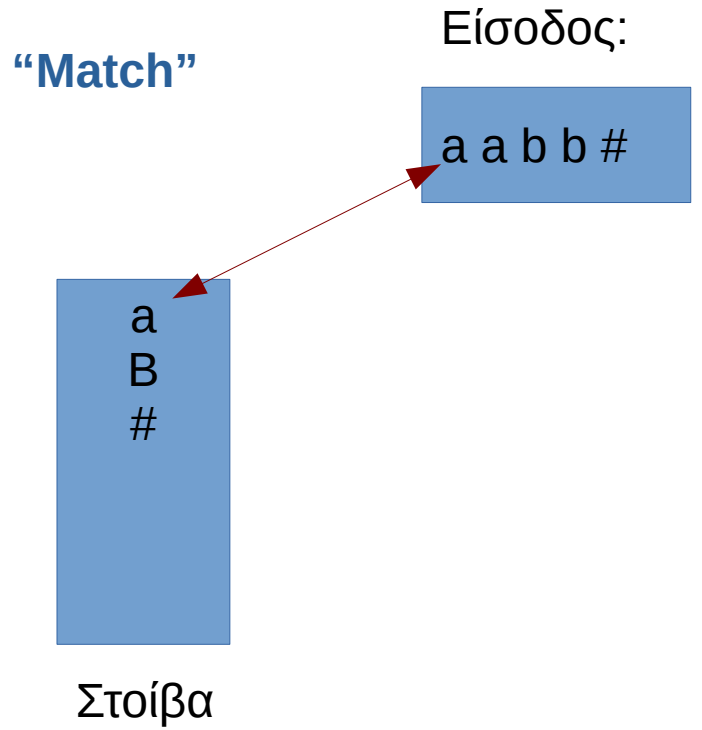
Είσοδος:

a a b b #

a
B
#

Στοίβα

	a	b
S	a B	
B	a B b	b

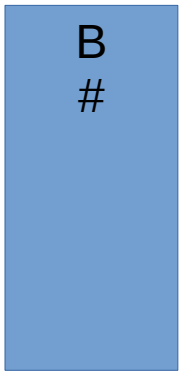


	a	b
S	a B	
B	a B b	b

Είσοδος:

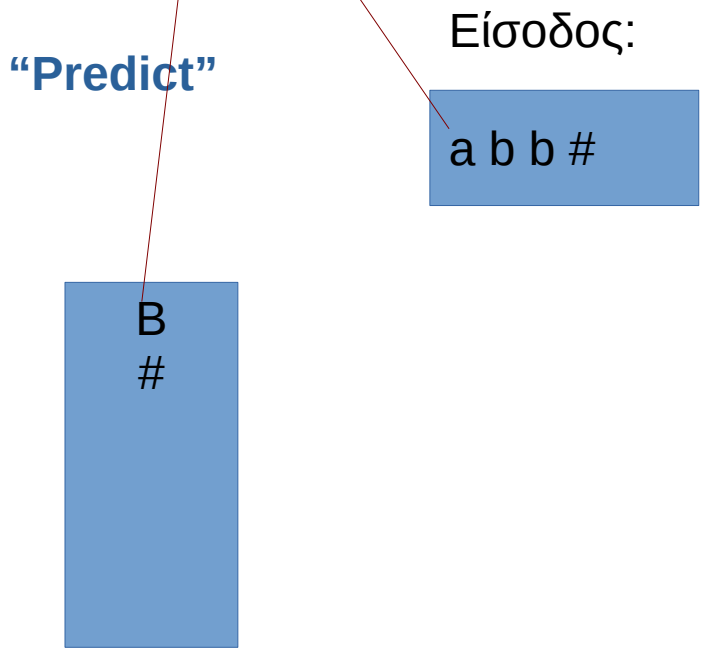
a b b #

“Match”



Στοίβα

	a	b
S	a B	
B	a B b	b



	a	b
S	a B	
B	a B b	b

“Predict”

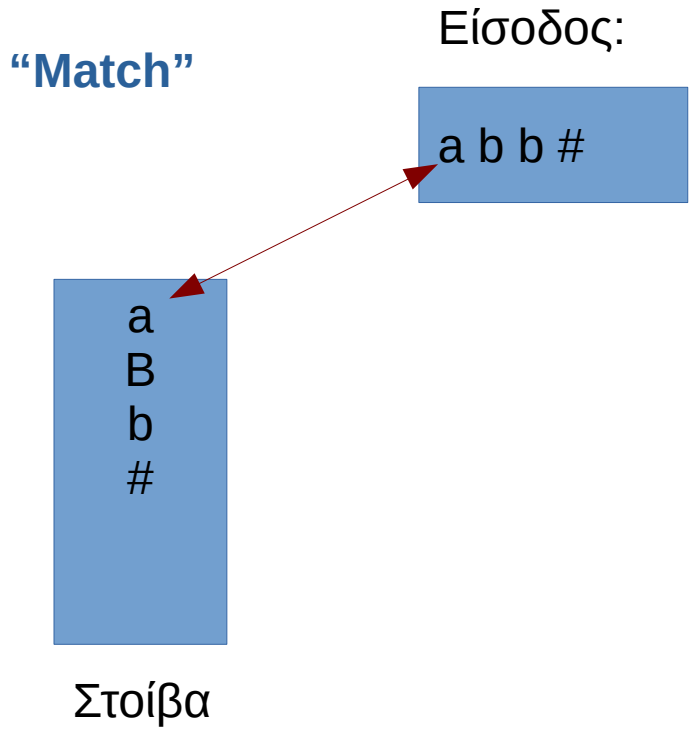
Είσοδος:

a b b #

a
B
b
#

Στοίβα

	a	b
S	a B	
B	a B b	b

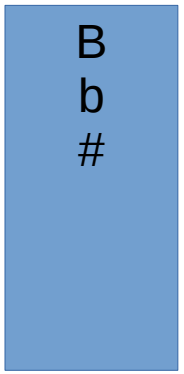


	a	b
S	a B	
B	a B b	b

Είσοδος:

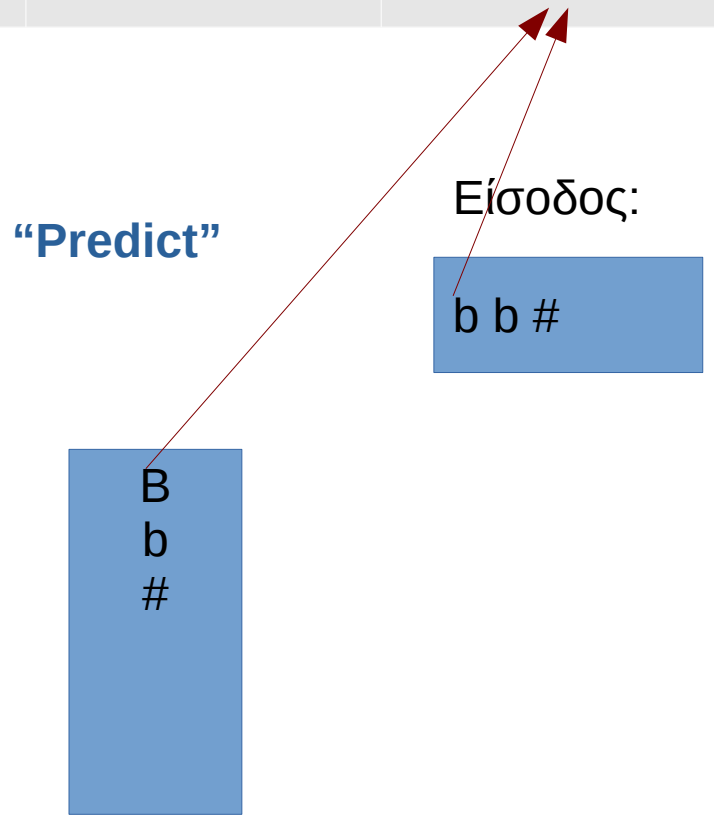
b b #

“Match”



Στοίβα

	a	b
S	a B	
B	a B b	b



	a	b
S	a B	
B	a B b	b

“Predict”

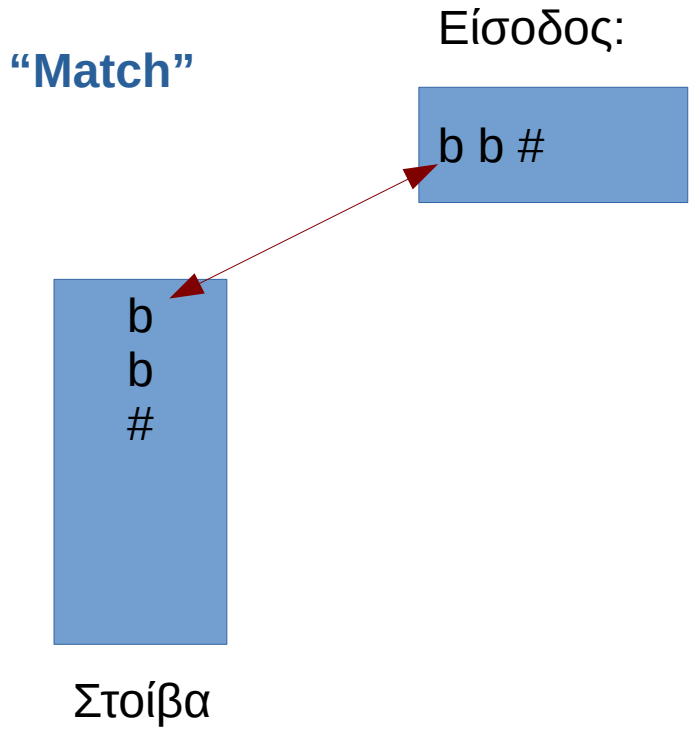
Είσοδος:

b b #

b
b
#

Στοίβα

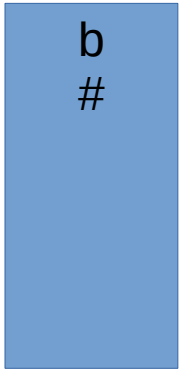
	a	b
S	a B	
B	a B b	b



	a	b
S	a B	
B	a B b	b

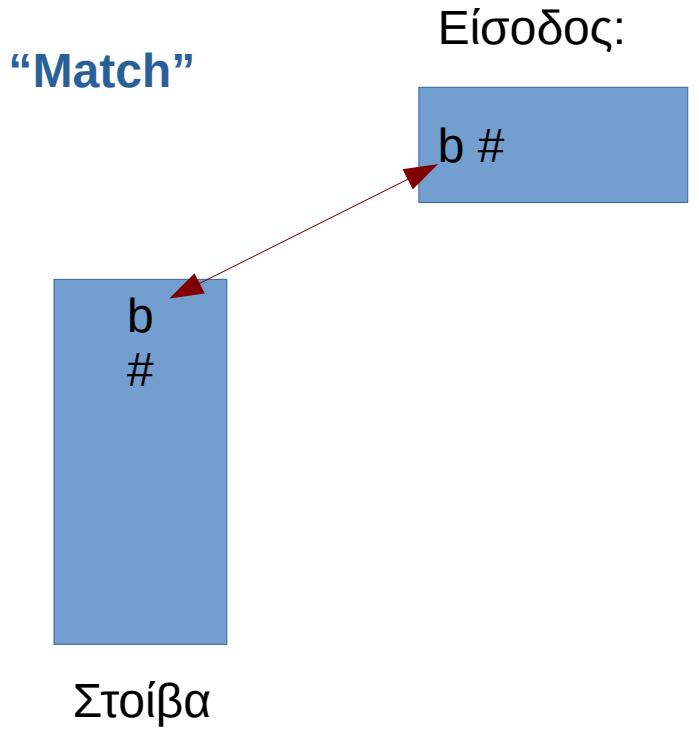
Είσοδος:

“Match”



Στοίβα

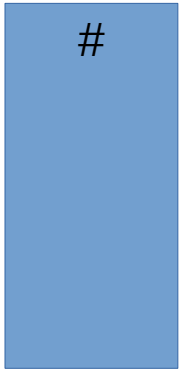
	a	b
S	a B	
B	a B b	b



	a	b
S	a B	
B	a B b	b

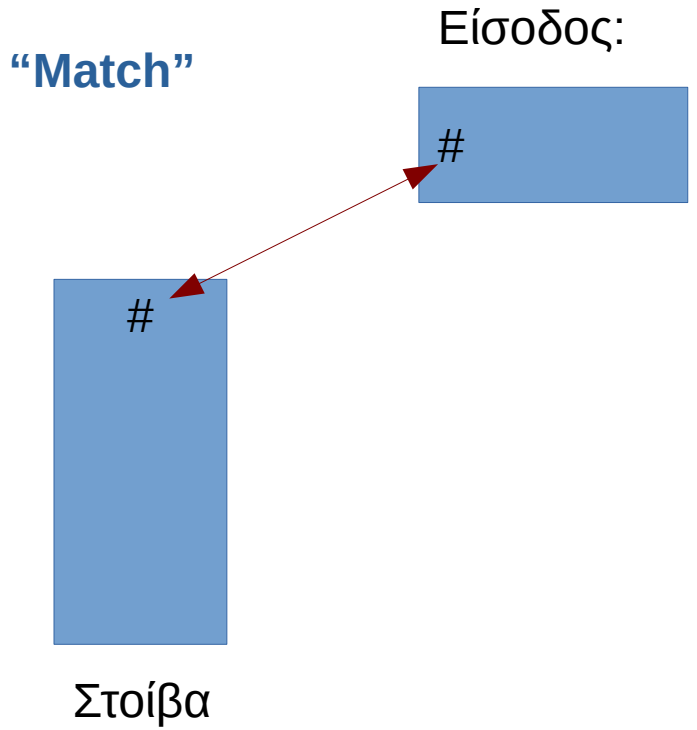
“Match”

Είσοδος:



Στοίβα

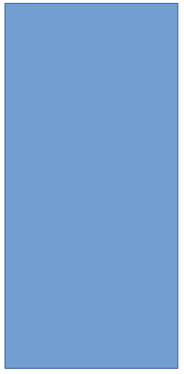
	a	b
S	a B	
B	a B b	b



	a	b
S	a B	
B	a B b	b

Είσοδος:

Τερματισμός



Στοίβα

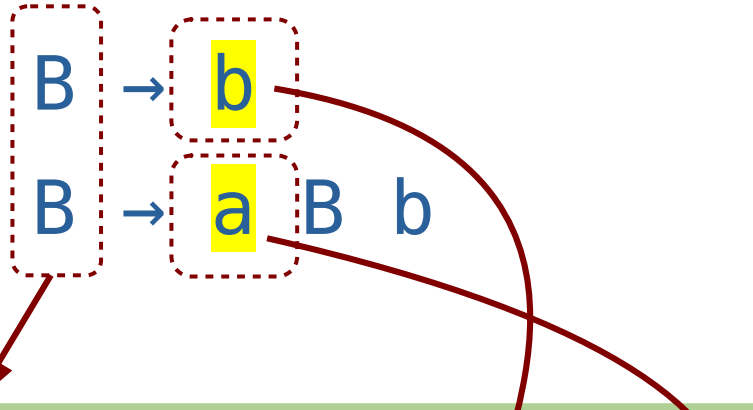
Κατασκευή συντακτικού αναλυτή LL(1)

- Θα μπορούσαμε να φτιάξουμε ένα πρόγραμμα που να μιμείται τη λειτουργία ενός PDA
- Συνήθως όμως υλοποιούμε τον συντακτικό αναλυτή ως ένα σύνολο συναρτήσεων που καλούν η μία την άλλη (ενδεχομένως και αναδρομικά)
  - Stack του PDA: υλοποιείται από τη στοίβα κλήσης των συναρτήσεων
  - Μπορούμε να κάνουμε σημασιολογική ανάλυση και να κατασκευάσουμε το AST μέσα στις συναρτήσεις αυτές
- Ο συντακτικός αναλυτής που κατασκευάζεται με τη μέθοδο αυτή ονομάζεται «αναδρομικής κατάβασης»
  - Recursive descent parser

Μέθοδος αναδρομικής κατάβασης

- Για κάθε μη τερματικό σύμβολο φτιάχνουμε μια συνάρτηση
  - Στο κυρίως πρόγραμμα καλείται πρώτη η συνάρτηση του αρχικού μη τερματικού συμβόλου της γραμματικής
- Κάθε συνάρτηση υλοποιεί όλους τους κανόνες που έχουν στο αριστερό μέρος το αντίστοιχο μη τερματικό σύμβολο
  - Ένας κλάδος if για κάθε κανόνα
  - Πώς διαλέγω; στην απλή γραμματική του προηγούμενου παραδείγματος, αρκεί να ελέγχω το επόμενο σύμβολο εισόδου
    - Δεν αρκεί για πιο σύνθετες γραμματικές…
  - Υλοποίηση του δεξιού μέρος κάθε κανόνα
    - Όπου υπάρχει τερματικό, καλώ μια συνάρτηση match() η οποία, αν η είσοδος ταιριάζει με το αναμενόμενο τερματικό, προχωρά στο επόμενο σύμβολο εισόδου
    - Όπου υπάρχει μη τερματικό, καλώ την αντίστοιχη συνάρτηση

Παράδειγμα για τους κανόνες του B




```
def B():

    if next_token=='B_TOKEN':
        # B -> b
        match('B_TOKEN')

    elif next_token=='A_TOKEN':
        # B -> a B b
        match('A_TOKEN')
        B()
        match('B_TOKEN')

    else:
        raise ParseError("...error msg...")
```

Γραμματική LL(1) αριθμητικών εκφράσεων

```
Stmt_list   → Stmt Stmt_list | ε
Stmt        → id = Expr | print Expr
Expr        → Term Term_tail
Term_tail   → Addop Term Term_tail | ε
Term        → Factor Factor_tail
Factor_tail → Multop Factor Factor_tail | ε
Factor      → (Expr) | id | number
Addop       → + | -
Multop      → * | /
```

- Πιο σύνθετη από το προηγούμενο παράδειγμα
  - Υπάρχουν κανόνες που το δεξιό μέρος ξεκινά με μη τερματικό
  - Υπάρχουν κενές παραγωγές (με ε στο δεξιό μέρος των κανόνων)
- Ποιους κανόνες μπορείτε να υλοποιήσετε με όσα ξέρετε μέχρι τώρα;