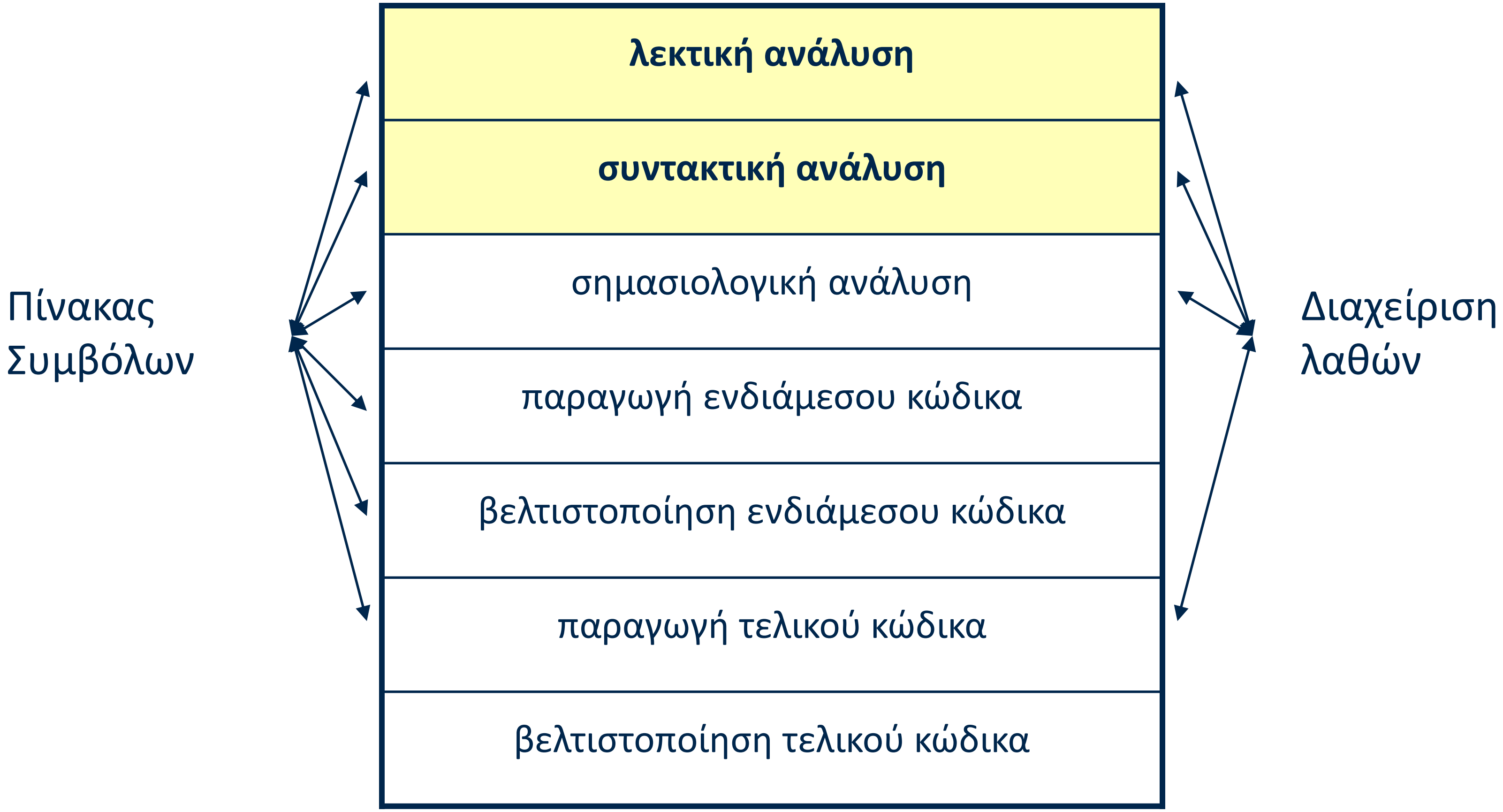
Γραμματικές και Μεταφραστές

Διαλέξεις στο μάθημα: Μεταφραστές
Γεώργιος Μανής

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ Η/Υ & ΠΛΗΡΟΦΟΡΙΚΗΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ
DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
UNIVERSITY OF IOANNINA

Στις φάσεις της μεταγλώττισης



Γραμματικές

- T: αλφάβητο **τερματικών** συμβόλων
- N: αλφάβητο **μη τερματικών** συμβόλων
  - $T \cap N = \varnothing$
- P: πεπερασμένο σύνολο **κανόνων** παραγωγής
  - $P \subseteq (T \cup N)^* \times (T \cup N)^*$
  - συμβολισμός: α -> β
- S: **αρχικό σύμβολο**

Παραγωγή τυπικής γλώσσας

- Κάθε γραμματική G=(T,N,P,S) ορίζει μία **γλώσσα** $L(G) \subseteq T^*$
- Λέμε ότι η **γραμματική παράγει τη γλώσσα** και ορίζεται ως
  - $L(G) = \{ \alpha \in T^* \mid S \Rightarrow^+ \alpha \}$
  - το σύμβολο $\Rightarrow^+$ σημαίνει ότι η συμβολοσειρά παράγεται σε ένα ή περισσότερα βήματα
- Μία γλώσσα λέγεται **τυπική** όταν **υπάρχει γραμματική που την παράγει**

Ιεραρχία Chomsky

<u>Γλώσσες</u>	**<u>Γραμματικές</u>**	**<u>Μηχανές</u>**
Κανονικές Γλώσσες	Κανονικές Γραμματικές	Πεπερασμένα αυτόματα
Γλώσσες Χωρίς Συμφραζόμενα	Γραμματικές Χωρίς Συμφραζόμενα	Μη-Ντετερμινιστικά Αυτόματα Με Στοίβα
Γλώσσες Με Συμφραζόμενα	Γραμματικές Με Συμφραζόμενα	Γραμμικά Πεπερασμένα Αυτόματα
Υπολογίσιμες γλώσσες	Γραμματικές χωρίς περιορισμούς	Μηχανή Turing

Γραμματικές χωρίς περιορισμούς

- Γραμματικές **χωρίς περιορισμούς** – τύπου 0 – προτασιακής σύνταξης
  - κανόνες της μορφής σ -> τ
  - $\sigma \in V^+$, $V = T \cup N$
  - $\tau \in V^*$
  - οι γραμματικές αυτές είναι δύσκολο να συνταχθούν και ο τρόπος αυτός παράστασης δεν είναι φιλικός προς τον άνθρωπο. Οι πρακτικές τους εφαρμογές είναι ελάχιστες έως ανύπαρκτες

Γραμματικές με συμφραζόμενα

- Γραμματικές **με συμφραζόμενα** – τύπου 1
  - κανόνες της μορφής αΑβ -> αxβ
  - $A \in N$
  - $x \in V^+$
  - $\alpha, \beta \in V^*$
  - κατ' εξαίρεση:
    - επιτρέπεται το S -> ε ώστε να παράγεται η κενή συμβολοσειρά
    - αρκεί το S να μην υπάρχει σε δεξί μέλος κανόνα

Γραμματικές με συμφραζόμενα

- Ισοδύναμη μορφή γραμματικών με συμφραζόμενα, καλούνται ***μονότονες*** γραμματικές
  - κανόνες της μορφής α -> β
  - α,β $\in V^+$
  - η α περιέχει τουλάχιστον ένα μη τερματικό σύμβολο
  - $| \alpha | \leq | \beta |$
  - κατ’ εξαίρεση:
    - επιτρέπεται το S -> ε ώστε να παράγεται η κενή συμβολοσειρά
    - αρκεί το S να υπάρχει μόνο σε δεξί μέλος κανόνα

Γραμματικές με συμφραζόμενα

- Παράδειγμα κατασκευής γραμματικής τύπου 1
  - θα κατασκευάσουμε γραμματική για τη γλώσσα: $\mathbf{\alpha^n b^n c^n}$
  - αποδεικνύεται ότι δε μπορεί να κατασκευαστεί γραμματική τύπου 2 για τη γλώσσα αυτή

Γραμματικές με συμφραζόμενα

- αρχίζοντας από την απλή περίπτωση
  - S -> abc
- αν προσθέσουμε ένα a στην αρχή θα πρέπει να προσθέσουμε κάτι και στο τέλος
  - S -> aSQ
- στο τέλος πρέπει να υπάρχει κάτι με b και c αλλά όχι απλά να προσθέσουμε bc αφού όλα τα b πρέπει να βρίσκονται πριν τα c. Κάτι τέτοιο επιτρέπει το b να είναι πάντα πριν το c
  - bQc -> bbcc
- πάλι δεν τελειώσαμε αφού τα Q βρίσκονται δεξιά από τα c, αυτό μπορεί να διορθωθεί με τον κανόνα
  - cQ -> Qc
- η γραμματική ολοκληρώθηκε

Γραμματικές χωρίς συμφραζόμενα

- Γραμματικές **χωρίς συμφραζόμενα** - τύπου 2
  - κανόνες της μορφής A -> α
  - $A \in N$
  - $\alpha \in V^*$

- Παράδειγμα
  - <name> ::= tom | john | harry
  - <sentence> ::= name | <list> and <name>
  - <list> ::= <name>, <list> | name

Γραμματικές χωρίς συμφραζόμενα

- Γραμματική **τύπου LL(1)**
  - L: **Left to right** - αναφέρεται στον τρόπο που σαρώνεται η συμβολοσειρά εισόδου, από τα αριστερά στα δεξιά
  - L: **Left most derivation** – η διαδικασία κατασκευής του συντακτικού δέντρου αντιστοιχεί στην αριστερότερη παραγωγή
  - (**1**): one look ahead symbol – για να επιλέξουμε ανάμεσα σε πιθανούς κανόνες ποιος είναι ο επόμενος κανόνας που πρέπει να εφαρμοσθεί αρκεί να γνωρίζουμε το επόμενο σύμβολο της συμβολοσειράς εισόδου

Γραμματικές χωρίς συμφραζόμενα

- Παράδειγμα γραμματικής τύπου LL(1)
  - E -> TE’
  - E’ -> +TE’ | ε
  - T -> FT’
  - T’ -> x FT’ | ε
  - F -> (E) | id
  - id -> a | b | c | d

- Παραγωγή:
  - (a+bxc+bx(a+c))xd

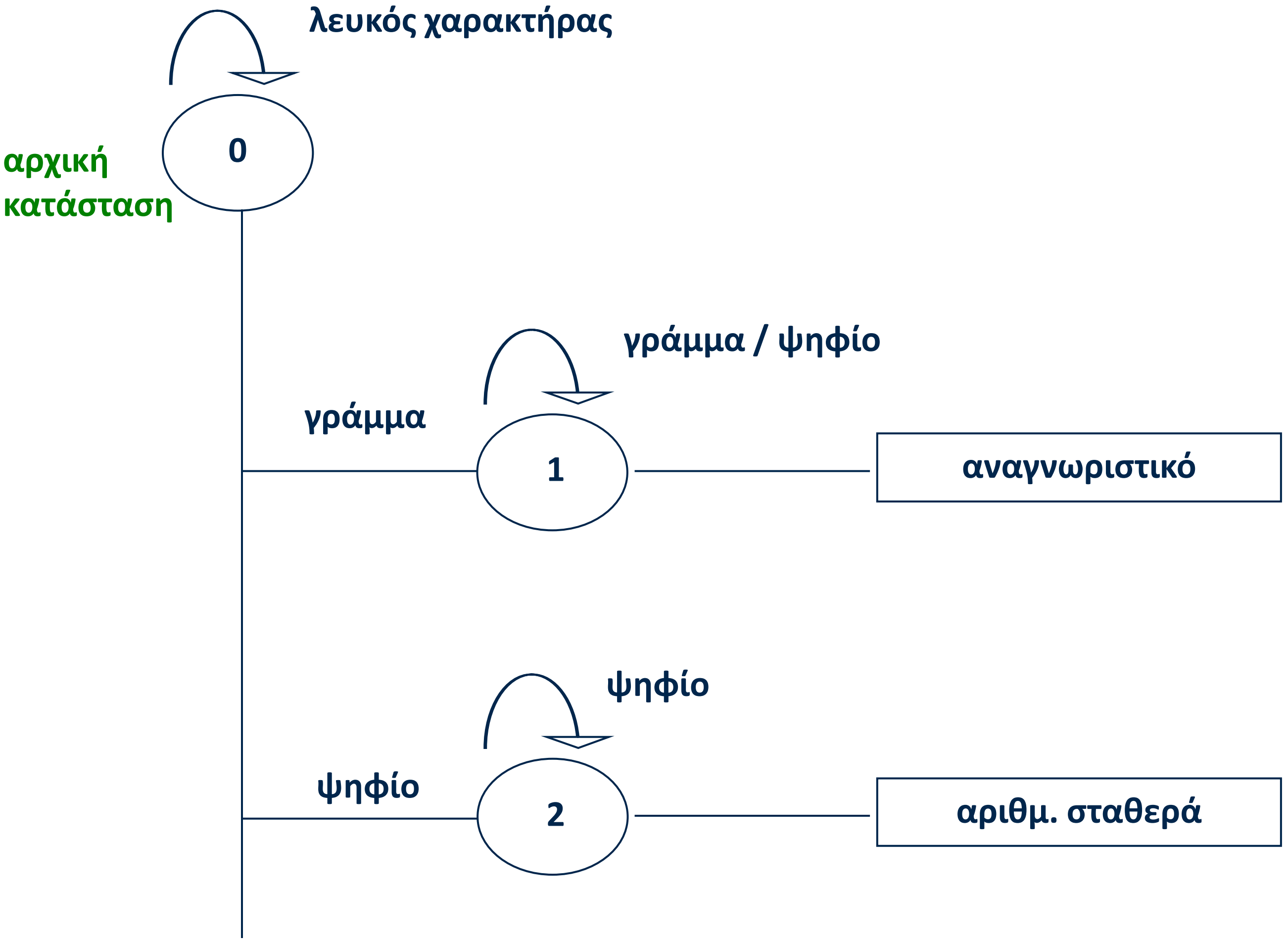
Κανονικές γραμματικές

- Κανονικές γραμματικές – τύπου 3 – γραμματικές **πεπερασμένων καταστάσεων**
  - κανόνες της μορφής
    - A -> α
    - A -> βB
  - A,B $\in$ N
  - α $\in$T $\cup$ {ε}
  - β $\in$T

Κανονικές γραμματικές

- Παράδειγμα
  - S -> a A
  - A -> a A
  - A -> d A
  - A -> ε
- Παραγωγή:
  - σειρές από α και d που ξεκινούν υποχρεωτικά από α

Πεπερασμένα αυτόματα



Γραμματικές πεπερασμένων επιλογών

- Γραμματικές **πεπερασμένων επιλογών** ή τύπου 4
  - κανόνες της μορφής A -> α
  - $A \in N$
  - $\alpha \in T \cup \{\varepsilon\}$

Παραγωγή συμβολοσειρών από γραμματική

- **Επιθυμούμε να παραγάγουμε όλες τις συμβολοσειρές που παράγει μία γραμματική**
  - ξεκινούμε από ένα **σύνολο προς εξέταση συμβολοσειρών** που περιέχει το **αρχικό σύμβολο** της γραμματικής
  - διαπερνούμε τη συμβολοσειρά εισόδου από **αριστερά προς τα δεξιά** ψάχνοντας για υποσυμβολοσειρές που **ταιριάζουν με το αριστερό μέλος** κάποιου γραμματικού κανόνα
  - για κάθε τέτοια υποσυμβολοσειρά που βρίσκουμε
    - δημιουργούμε ένα **αντίγραφο** της συμβολοσειράς εισόδου
    - **αντικαθιστούμε** την υποσυμβολοσειρά που βρίσκουμε στο αριστερό μέλος του κανόνα με αυτό που βρίσκεται στο δεξιό μέλος
    - και την **τοποθετούμε** στο σύνολο των **προς εξέταση συμβολοσειρών**

Παραγωγή συμβολοσειρών από γραμματική

- αν η συμβολοσειρά που παράχθηκε **δεν περιέχει μη τερματικά** σύμβολα τότε την **τοποθετούμε σε ένα σύνολο που περιέχει τις συμβολοσειρές που παράχθηκαν**
- στη συνέχεια **αντλούμε μία νέα προς εξέταση συμβολοσειρά από το σύνολο των προς εξέταση συμβολοσειρών** και **επαναλαμβάνουμε τα τρία προηγούμενα βήματα**

Παραγωγή συμβολοσειρών από γραμματική

- Παράδειγμα της γενικής περίπτωσης για τη γραμματική
    - S -> abc
    - S -> aSQ
    - bQc -> bbcc
    - cQ -> Qc
  - η οποία περιγράφει τη γλώσσα $\alpha^n b^n c^n$

Παραγωγή συμβολοσειρών από γραμματική

- Εκτέλεση του αλγορίθμου
  - S
  - **abc** aSQ
  - aSQ
  - aabcQ aaSQQ
  - aaSQQ aabQc
  - aabQc aaabcQQ aaaSQQQ
  - aaabcQQ aaaSQQQ aabbcc
  - aaaSQQQ aabbcc aaabQcQ
  - **aabbcc** aaabQcQ aaaabcQQQ aaaaSQQQQ
  - aaabQcQ aaaabcQQQ aaaaSQQQQ
  - κλπ.

- **S -> abc**
- **S -> aSQ**
- **bQc -> bbcc**
- **cQ -> Qc**

Παραγωγή συμβολοσειρών από γραμματική

- Παρατήρηση
  - **στην περίπτωση που επιχειρήσουμε να μη φτιάξουμε όλα τα αντίγραφα** που προκύπτουν από όλους τους κανόνες αλλά να επιλέξουμε έναν και να κάνουμε εκεί την αντικατάσταση **υπάρχει κίνδυνος να χαλάσουμε τα συμφραζόμενα της γραμματικής**.
  - Έστω η γραμματική

S -> AC	
A -> b	αντικαθιστώντας με βάση το A->b η γραμματική δε
AC -> ab	θα παράγει τίποτα, ενώ θα μπορούσε να παράγει το ab

  - Εάν η γραμματική είναι χωρίς συμφραζόμενα δεν υπάρχουν κάποια συμφραζόμενα τα οποία μπορεί να καταστραφούν

Κατηγορικές γραμματικές

- κάθε μη τερματικό σύμβολο συνοδεύεται από **ένα ή περισσότερα κατηγορήματα**
- **κάθε κατηγόρημα** είναι μία απλή τιμή π.χ. αριθμός, συμβολοσειρά κλπ.
- σε ένα κανόνα (π.χ μαθηματικό τύπο) **συνδυάζουμε** τα κατηγορήματα κάποιων από τα μη τερματικά σύμβολα του κανόνα για να παράγουμε κάποια άλλα κατηγορήματα για μη τερματικά σύμβολα
- αν το κατηγόρημα που υπολογίζεται ανήκει σε μη τερματικό σύμβολο που βρίσκεται σε αριστερό μέλος ενός κανόνα τότε το κατηγόρημα θεωρείται ότι **συντίθεται** (derived, synthesized)
- αν το κατηγόρημα που υπολογίζεται ανήκει σε μη τερματικό σύμβολο που βρίσκεται σε δεξιό μέλος ενός κανόνα τότε το κατηγόρημα θεωρείται ότι **κληρονομείται** (inherited)

Κατηγορικές γραμματικές

- Παράδειγμα σύνθεσης
  - στο παράδειγμα για λόγους απλότητας θεωρούμε ένα κατηγόρημα για κάθε μη τερματικό σύμβολο το οποίο συμβολίζεται με A_x, όπου x η σειρά με την οποία συναντάται το μη τερματικό σύμβολο στον κανόνα
    - Sum -> Digit $\{ A_o = A_1 \}$
    - Sum -> Sum + Digit $\{ A_o = A_1 + A_3 \}$
    - Digit -> 0 $\{ A_o = 0 \}$
    - Digit -> 1 $\{ A_o = 1 \}$
    - …
    - Digit -> 9 $\{ A_o = 9 \}$

Κατηγορικές γραμματικές

- Παράδειγμα κατηγορήματος που κληρονομείται
  - το ID αναγνωρίζεται στον κανόνα `<PROGRAM>`
  - στον κανόνα `<PROGRAMBLOCK>` πρέπει να χρησιμοποιηθεί

```
<PROGRAM>          ::= program ID
                       <PROGRAMBLOCK>

<PROGRAMBLOCK>     ::= <DECLARATIONS>
                       <SUBPROGRAMS>
                       <BLOCK>
```

Από τη γραμματική στον κώδικα

```
void program()
{
      if (token==programtk) {
                token=lex();
                if (token==idtk) {
                          token=lex();
                          programBlock(ID);   }
                else error("program name exprected");      }
      else error ("the keyword 'program' was expected");
      }
}
                          void programBlock(ID)
                          {
                              declarations();
                              subprograms();
                              print "begin block",ID
                              block();
                              print "end block",ID
                          }
```

Μεταλλακτικές γραμματικές

- Κάθε κανόνας **εμφανίζει ως έξοδο κάποια συμβολοσειρά**
- Παράδειγμα

Sum -> Digit	{ make it the result }
Sum -> Sum + Digit	{ add it to the previous result}
Digit -> 0	{ take a 0 }
Digit -> 1	{ take a 1 }
…	
Digit -> 9	{take a 9 }

Με είσοδο 3+5 η γραμματική θα παραγάγει:

take a 3
make it result
take a 5
add it to the previous result

Δυναμικότητα γραμματικών και μεταφραστές

- Το σύνολο όλων των **λεκτικά ορθών προγραμμάτων** μπορεί να περιγραφεί από μία κανονική γραμματική.
- Μπορούμε να αναγνωρίσουμε ότι:
  - δεσμευμένες λέξεις
  - δεν συναντάμε end-of-file ενώ έχουν ανοίξει σχόλια και δεν έχουν κλείσει
  - τα αναγνωριστικά ξεκινούν από γράμμα και ακολουθεί μία σειρά γραμμάτων και αριθμών
  - κλπ

Δυναμικότητα γραμματικών και μεταφραστές

- Το σύνολο των **συντακτικά ορθών προγραμμάτων** περιγράφεται από μία γραμματική χωρίς συμφραζόμενα. Μπορούμε να αναγμωρίσουμε σφάλματα όπως:
  - βρέθηκε else χωρίς if
  - εκχώρηση σε αριθμητική σταθερά
  - κλπ.
- Το σύνολο των **σημασιολογικά ορθών προγραμμάτων** περιγράφεται από μία γραμματική με συμφραζόμενα. Μπορούμε να απαντήσουμε σε ερωτήματα όπως:
  - η μεταβλητή αυτή έχει δηλωθεί;
  - πρόκειται για απλή μεταβλητή ή πίνακα;
  - γράφουμε μήπως έξω από τα όρια ενός πίνακα;

Δυναμικότητα γραμματικών και μεταφραστές

- Το σύνολο των **προγραμμάτων που τερματίζουν σε πεπερασμένο χρόνο** με συγκεκριμένη είσοδο μπορεί να περιγραφεί από μία (πολύ πολύπλοκη) γραμματική χωρίς περιορισμούς

- Το σύνολο των **προγραμμάτων που λύνουν ένα συγκεκριμένο πρόβλημα δε μπορεί να περιγραφεί** από καμία γραμματική

Διφορούμενες γραμματικές – αποδιφοροποίηση

- **Διφορούμενη** είναι μία γραμματική όταν αυτή μπορεί να **παραγάγει περισσότερα από ένα συντακτικά δέντρα**
- Για να **αποδείξουμε** ότι μία γραμματική είναι διφορούμενη αρκεί **για την ίδια είσοδο** και ακολουθώντας είτε δεξιότερες είτε αριστερότερες παραγωγές να μπορούν να **προκύψουν παραπάνω από ένα συντακτικά δέντρα**
- Μια μέθοδος η οποία μας βοηθά **να μετατρέψουμε μια γραμματική σε μη διφορούμενη** είναι η εισαγωγή μη τερματικών συμβόλων σε κανόνες δυαδικών τελεστών, ορίζοντας νέα **προσεταιριστικότητα και προτεραιότητα**

Διφορούμενες γραμματικές – αποδιφοροποίηση

Έστω ότι έχουμε την παρακάτω γραμματική χωρίς συμφραζόμενα

```
S  ->  S # S  |
       S @ S  |
       a
```

Αποδείξτε με χρήση της συμβολοσειράς a@a#a ότι η γραμματική είναι διφορούμενη.

Εισάγετε κανόνες ώστε να άρετε την ασάφεια ενεργοποίησης των συντακτικών κανόνων.

Διφορούμενες γραμματικές – αποδιφοροποίηση

```
S  ->  S # S  |
       S @ S  |
       a
```

Παρακάτω εμφανίζονται δύο δέντρα που παράγονται από την ίδια γραμματική:

Κανόνας	Παραγωγή
	S
S→S#S	S#S
S→S@S	S@S#S
S→a	a@S#S
S→a	a@a#S
S→a	a@a#a

Κανόνας	Παραγωγή
	S
S→S@S	S@S
S→a	a@S
S→S#S	a@S#S
S→a	a@a#S
S→a	a@a#a

Διφορούμενες γραμματικές – αποδιφοροποίηση

Για την **λύση** του συγκεκριμένου προβλήματος εισάγουμε τα μη τερματικά σύμβολα P και R και εισάγουμε **προτεραιότητα** ανάμεσα στους τελεστές @ και #. Με βάση τα παραπάνω προκύπτει η γραμματική:

```
S  ->  S @ P  |  P
P  ->  P # R  |  R
R  ->  a
```

ευχαριστώ !!!