

**Ασκήσεις μελέτης της ενότητας B3 (συντακτική και σημασιολογική ανάλυση φυσικής γλώσσας με γραμματικές)**

1. Σχεδιάστε τα δύο συντακτικά δέντρα που προκύπτουν για την πρόταση «είδαμε το γιατρό με την άσπρη μπλούζα», όταν χρησιμοποιείται η ακόλουθη γραμματική με αρχικό σύμβολο S.

|                      |                            |
|----------------------|----------------------------|
| V → είδαμε           | Nominal → N   Adj N   N PP |
| Det → το   την       | NP → Det Nominal           |
| N → γιατρό   μπλούζα | PP → Prep NP               |
| Prep → με            | VP → V NP   V NP PP        |
| Adj → άσπρη          | <u>S</u> → VP              |

2. Δίνεται η ακόλουθη γραμματική:

|                       |                                              |
|-----------------------|----------------------------------------------|
| S → V NP              | Adj → πρωινή   απογευματινή   ακριβή   φτηνή |
| NP → Det Nominal      | V → θέλω   επιθυμώ                           |
| Det → μια             |                                              |
| Nominal → Adj Nominal | Nominal → πτήση                              |

Συμπληρώστε στον παρακάτω πίνακα τους υπολογισμούς που θα κάνει ο **αλγόριθμος CKY** κατά την ανάλυση της πρότασης «**επιθυμώ μια φτηνή απογευματινή πτήση**».

|   | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| 0 |   |   |   |   |   |   |
| 1 |   |   |   |   |   |   |
| 2 |   |   |   |   |   |   |
| 3 |   |   |   |   |   |   |
| 4 |   |   |   |   |   |   |

3. Επιβεβαιώστε ότι η πολυπλοκότητα του αλγορίθμου CKY είναι  $O(n^3)$ , όπου  $n$  το πλήθος των λέξεων της πρότασης.

*Απάντηση:* Έχουμε να γεμίσουμε  $O(n^2)$  κελιά του πίνακα. Σε κάθε κελί του πίνακα, έχουμε να ελέγξουμε  $O(n)$  δυνατούς τεμαχισμούς (σε δύο μέρη  $A, B$  που θα συνδυάζονται με κάποιον κανόνα  $X \rightarrow A B$ ) του τμήματος της πρότασης εισόδου που αντιστοιχεί σε αυτό το κελί. Για κάθε έλεγχο δυνατού τεμαχισμού (σε δύο μέρη  $A, B$ ), πρέπει να σαρώσουμε τους κανόνες της γραμματικής (εξαιρώντας τους κανόνες του λεξικού) για να βρούμε τους αντίστοιχους κανόνες (μορφής  $X \rightarrow A B$ ). Αν η γραμματική έχει  $|G|$  κανόνες (εξαιρώντας τους κανόνες του λεξικού) και ο έλεγχος της καταλληλότητας κάθε κανόνα κοστίζει  $O(1)$ , τότε κάθε σάρωση της γραμματικής κοστίζει  $O(|G|)$  και η συνολική πολυπλοκότητα γίνεται  $O(n^3|G|)$ .<sup>1</sup> Αν αγνοήσουμε το μέγεθος της γραμματικής, τότε η συνολική πολυπλοκότητα είναι  $O(n^3)$ .

<sup>1</sup> Βλ. και [https://en.wikipedia.org/wiki/CYK\\_algorithm](https://en.wikipedia.org/wiki/CYK_algorithm). Η απάντηση βασίζεται σε λεπτομερέστερα σχόλια του φοιτητή Μιχάλη Βαζαίου, τον οποίον ο διδάσκων ευχαριστεί.

4. (α) Εξηγήστε πώς ακριβώς είναι δυνατόν να τροποποιηθεί ο αλγόριθμος CKY, ώστε να παράγει και συντακτικά δέντρα. (β) Δείξτε πώς θα γινόταν η παραγωγή του συντακτικού δέντρου στο παράδειγμα των διαφανειών που δείχνουν τον τρόπο λειτουργίας του CKY. (γ) Υλοποιήστε τον αλγόριθμο CKY στη γλώσσα προγραμματισμού που προτιμάτε, ώστε να επιστρέφονται και τα συντακτικά δέντρα.

5. (α) Μετατρέψτε την ακόλουθη γραμματική σε μορφή DCG, προσθέτοντας επίσης έλεγχο συμφωνίας γένους, ώστε να μην επιτρέπονται προτάσεις όπως «η σκύλος κυνήγησε το γάτα».

|                        |                                               |
|------------------------|-----------------------------------------------|
| $S \rightarrow NP VP$  | $Det \rightarrow o \mid \eta \mid το \mid τη$ |
| $NP \rightarrow Det N$ | $N \rightarrow σκύλος \mid σκύλο \mid γάτα$   |
| $VP \rightarrow V NP$  | $V \rightarrow κυνήγησε \mid δάγκωσε$         |

(β) Τροποποιήστε τη γραμματική DCG του προηγούμενου σκέλους, ώστε αν την δώσουμε στην Prolog μαζί με το στόχο:

phrase(s(Action, Agent, Object), [ο, σκύλος, κυνήγησε, τη, γάτα]).

η Prolog να απαντήσει: Action = chase, Agent = dog, Object = cat,

ενώ στην περίπτωση του στόχου:

phrase(s(Action, Agent, Object), [ο, σκύλος, δάγκωσε, τη, γάτα]).

η Prolog να απαντήσει: Action = bite, Agent = dog, Object = cat.

*Απάντηση:*

s(Action, Agent, Object) --> np(Agent), vp(Action, Object).

np(Entity) --> det(Gender), n(Entity, Gender).

vp(Action, Object) --> v(Action), np(Object).

det(masc) --> [ο].

det(fem) --> [η].

det(masc) --> [το].

det(fem) --> [τη].

n(dog, masc) --> [σκύλος].

n(dog, masc) --> [σκύλο].

n(cat, fem) --> [γάτα].

v(chase) --> [κυνήγησε].

v(bite) --> [δάγκωσε].

6. Τροποποιήστε τη γραμματική DCG της διαφάνειας «Σημασιολογία αριθμητικής γλώσσας», ώστε να επιτρέπει αριθμητικές πράξεις μεταξύ αριθμών από το μηδέν ως και το 9.999. Ως απλούστευση, το δέκα να γράφεται ως [one, ten], το δώδεκα ως [one, ten, two], το 23 ως [two, ten, three], το εκατό ως [one, hundred], το 110 ως [one, hundred, one, ten], το 123 ως [one, hundred, two, ten, three], το 1000 ως [one, thousand], το 1123 ως [one, thousand, one, hundred, two, ten, three] κ.ο.κ. Για παράδειγμα, η ακόλουθη εντολή:

?-phrase(expression(V), [open, two, thousand, three, hundred, five, ten, two, plus, two, ten, one, close]).

θα πρέπει να επιστρέφει:

V = 2373

και η εντολή:

?-phrase(expression(V), [open, open, two, thousand, three, hundred, five, ten, two, plus, two, ten, one, close, minus, one, thousand, seven, ten, close]).

V = 1303

*Απάντηση:*

```
digit(0) --> [zero].      digit(1) --> [one].
digit(2) --> [two].       digit(3) --> [three].
digit(4) --> [four].      digit(5) --> [five].
digit(6) --> [six].       digit(7) --> [seven].
digit(8) --> [eight].     digit(9) --> [nine].
```

```
number(Value) --> digit(Value).
number(NewValue) --> digit(Value1), [ten], number(Value2),
    {Value2 < 10, NewValue is Value1 * 10 + Value2}.
number(NewValue) --> digit(Value1), [ten], {NewValue is Value1 * 10}.
number(NewValue) --> digit(Value1), [hundred], number(Value2),
    {Value2 < 100, NewValue is Value1 * 100 + Value2}.
number(NewValue) --> digit(Value1), [hundred], {NewValue is Value1 * 100}.
number(NewValue) --> digit(Value1), [thousand], number(Value2),
    {Value2 < 1000, NewValue is Value1 * 1000 + Value2}.
number(NewValue) --> digit(Value1), [thousand], {NewValue is Value1 * 1000}.
```

```
expression(Value) --> number(Value).
expression(NewValue) --> [open], expression(Value1), [plus], expression(Value2), [close],
    {NewValue is Value1 + Value2}.
expression(NewValue) --> [open], expression(Value1), [minus], expression(Value2), [close],
    {NewValue is Value1 - Value2}.
```

7. Γράψτε μια γραμματική DCG που να επιτρέπει προτάσεις όπως:

[άνοιξε,τα,παράθυρα,του,σαλονιού], [κλείσε,το,μικρό,παράθυρο,του,σαλονιού],  
[σβήσε,το,μεγάλο,φως,της,μικρής,κουζίνας], [άναψε,τα,φώτα,των,μικρών,μπάνιων]

αλλά να μην επιτρέπει προτάσεις με ασυμφωνία γένους, αριθμού ή πτώσης, όπως οι παρακάτω:

X [άνοιξε,το,παράθυρα,του,σαλονιού], X [κλείσε,το,μικρή,παράθυρο,του,σαλονιού],  
X [σβήσε,το,μεγάλου,φως,της,μικρής,κουζίνας], X [άναψε,τα,φώτα,το,μικρού,μπάνιων]

και να απαιτεί τα ρήματα «ανάβω» και «σβήνω» να χρησιμοποιούνται μόνο με φώτα, ενώ τα «ανοίγω» και «κλείνω» μόνο με παράθυρα. Για παράδειγμα, να μην επιτρέπονται προτάσεις όπως:

X [σβήσε,το,μικρό,παράθυρο,του,σαλονιού], X [άνοιξε,το,μεγάλο,φως,της,μικρής,κουζίνας]

Για τις επιτρεπόμενες προτάσεις, θέλουμε η γραμματική να μπορεί να χρησιμοποιηθεί όπως φαίνεται στα παρακάτω παραδείγματα:

?- phrase(s(Action, Type1, Size1, Number1, Type2, Size2, Number2,  
[άνοιξε,τα,μεγάλα,παράθυρα,του,μικρού,σαλονιού]).  
Action = open, Type1 = window, Size1 = large, Number1 = plur,

Type2 = livingroom, Size2=small, Number2 = sing

?- phrase(s(Action, Type1, Size1, Number1, Type2, Size2, Number2,  
[άναψε,τα,φώτα,των,μικρών,μπάνιων]).

Action = on, Type1 = light, Size1 = unknown, Number1 = plur,  
Type2 = bathroom, Size2=small, Number2 = plur

?- phrase(s(Action, Type1, Size1, Number1, Type2, Size2, Number2,  
[κλείσε,τα,παράθυρα]).

Action = close, Type1 = window, Size1 = unknown, Number1 = plur,  
Type2 = unknown, Size2=unknown, Number2 = unknown

*Απάντηση:*

s(Action, Type1, Size1, Number1, Type2, Size2, Number2) -->

vp(Action, Type1, Size1, Number1, Type2, Size2, Number2).

vp(Action, Type, Size, Number, unknown, unknown, unknown) -->

v(Action, Type), np(Gender, Number, acc, Type, Size).

vp(Action, Type1, Size1, Number1, Type2, Size2, Number2) -->

v(Action, Type1), np(Number1, acc, Type1, Size1), np(Number2, gen, Type2, Size2).

np(Number, Case, Type, Size) -->

det(Gender, Number, Case), nominal(Gender, Number, Case, Type, Size).

nominal(Gender, Number, Case, Type, Size) -->

adj(Gender, Number, Case, Size), n(Gender, Number, Case, Type).

nominal(Gender, Number, Case, Type, unknown) -->

n(Gender, Number, Case, Type).

v(on, light) --> [άναψε].

v(open, window) --> [άνοιξε].

v(off, light) --> [σβήσε].

v(close, window) --> [κλείσε].

det(neut, sing, acc) --> [το].

det(neut, sing, gen) --> [του].

det(Gender, plur, gen) --> [των].

det(neut, plur, acc) --> [τα].

det(fem, sing, gen) --> [της].

n(neut, sing, acc, window) --> [παράθυρο].

n(neut, plur, acc, window) --> [παράθυρα].

n(neut,sing,acc, light) --> [φως].

n(neut, plur, acc, light) --> [φώτα].

n(neut, sing, gen, livingroom) --> [σαλονιού].

n(neut, plur, gen, livingroom) --> [σαλονιών].

n(neut, sing, gen, bathroom) --> [μπάνιου].

n(neut, plur, gen, bathroom) --> [μπάνιων].

n(fem, sing, gen, kitchen) --> [κουζίνας].

n(fem, plur, gen, kitchen) --> [κουζινών].

adj(neut, sing, acc, small) --> [μικρό].  
adj(neut, sing, gen, small) --> [μικρού].  
adj(fem, sing, acc, small) --> [μικρή].  
adj(fem, sing, gen, small) --> [μικρής].  
adj(Gender, plur, gen, small) --> [μικρών].  
adj(neut, sing, acc, large) --> [μεγάλο].  
adj(neut, sing, gen, large) --> [μεγάλου].  
adj(fem, sing, acc, large) --> [μεγάλη].  
adj(fem, sing, gen, large) --> [μεγάλης].  
adj(Gender, plur, gen, large) --> [μεγάλων].