

# ΠΛΗ31

## PROLOG

Μάθημα 1:  
Εισαγωγή

Δημήτρης Ψούνης



[www.psounis.gr](http://www.psounis.gr)

## ΠΕΡΙΕΧΟΜΕΝΑ

### A. Σκοπός του Μαθήματος

### B. Θεωρία

#### 1. Σταθερές και Μεταβλητές

#### 2. Γεγονότα

#### 3. Ερωτήσεις

1. Ερωτήσεις σε Γεγονότα
2. Ερωτήσεις με μεταβλητές
3. Ερωτήσεις με ανώνυμες μεταβλητές
4. Σύνθετες ερωτήσεις

#### 4. Κανόνες

1. Ορισμός Κανόνα
2. Ενσωμάτωση στο Πρόγραμμα
3. Αναδρομικοί Κανόνες

### Γ. Ασκήσεις

## B. Θεωρία

### Η Γλώσσα Προγραμματισμού Prolog

- Η Prolog είναι μια γλώσσα προγραμματισμού που βασίζεται στην κατηγορηματική λογική.
  - Δεν έχει καμία σχέση με τον διαδικαστικό προγραμματισμό (δεν υπάρχει if και δεν υπάρχει for – οι δύο ουσιώδεις δομές κάθε διαδικαστικής γλώσσας προγραμματισμού)
- Ο στόχος της Prolog είναι:
  - Να ενσωματώσει γνώση του πραγματικού κόσμου στο πρόγραμμα.
  - Να ορίσει ένα σύνολο κανόνων εξαγωγής νέας γνώσης.
- Με τον τρόπο αυτό:
  - Ο προγραμματιστής δεν θα καθοδηγεί το πρόγραμμα για το πώς να κάνει μια ενέργεια.
  - Ο προγραμματιστής θα ρωτάει το πρόγραμμα για μια πληροφορία και αυτό θα εξάγει μόνο του την απάντηση.

## B. Θεωρία

### 1. Σταθερές και Μεταβλητές

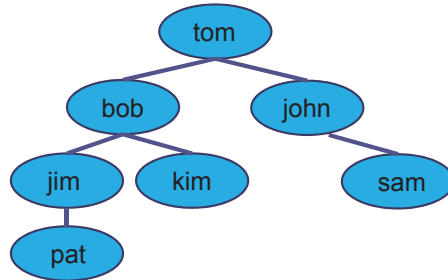
- Στην Prolog μία σταθερά αναπαρίσταται με μικρούς λατινικούς χαρακτήρες.
- Π.χ.
  - socrates
  - tom
  - man
- Αντίθετα μια μεταβλητή ξεκινάει πάντα με κεφαλαίο γράμμα
  - X
  - Person
  - Father

## Β. Θεωρία

### 2. Γεγονότα

- Η αναπαράσταση υφιστάμενης γνώσης σε ένα πρόγραμμα Prolog γίνεται μέσω κανόνων που ονομάζονται γεγονότα.
- Το όνομα ενός κανόνα είναι δικής μας επιλογής και απεικονίζει μια πληροφορία που έχουμε για τον πραγματικό κόσμο
- Π.χ. αν `parent/2` ένα κατηγορήμα που εκφράζει ότι το 1<sup>ο</sup> όρισμα είναι γονέας του 2<sup>ου</sup> ορίσματος τότε το σύνολο γεγονότων που φαίνεται αριστερά, αναπαριστά το οικογενειακό δένδρο που φαίνεται δεξιά.

```
parent(tom,bob).
parent(tom,john).
parent(bob,jim).
parent(bob,kim).
parent(john,sam).
parent(jim,pat).
```



## Β. Θεωρία

### 3. Ερωτήσεις

- Στην SWI Prolog αποθηκεύουμε τα γεγονότα σε ένα ξεχωριστό αρχείο κειμένου (κατά προτίμηση γραμμένο με τον κειμενογράφο) με προέκταση `.pl`
  - Η συγγραφή των αρχείων γεγονότων γίνεται και με την SWI Prolog επιλέγοντας `File->New` οπότε θα ανοίξει ο κειμενογράφος στον οποίο μπορούμε να καταγράψουμε τα γεγονότα.
- Αφού έχουμε γράψει τα γεγονότα, επιλέγουμε `File->Consult` και επιλέγουμε το αρχείο. Αυτομάτως τα γεγονότα φορτώνονται στον πυρήνα της Prolog, άρα θεωρούνται γνώση του συστήματος.
- Πλέον είμαστε σε θέση να κάνουμε ερωτήσεις που αφορούν τα γεγονότα που έχουμε καταγράψει

## Β. Θεωρία

### 3. Ερωτήσεις (1. Ερωτήσεις σε Γεγονότα)

- Οι ερωτήσεις γίνονται στην κονσόλα της Prolog αμέσως μετά το σήμα `?-`
- Παραδείγματα ερωτήσεων γεγονότων. Προσοχή, ότι το ερώτημα είναι το όνομα του κατηγορήματος ακολουθούμενο από την τελεία:

```
?- parent(tom,bob).
true.

?- parent(tom,kim).
false.

?- parent(tom,michael).
false.
```

## Β. Θεωρία

### 3. Ερωτήσεις (2. Ερωτήσεις με μεταβλητές)

- Παραδείγματα ερωτήσεων με μεταβλητές.

```
?- parent(tom,X).
X = bob ;
X = john.

?- parent(X,john).
X = tom.

?- parent(X,Y).
X = tom,
Y = bob ;
X = tom,
Y = john ;
.
.
.
```

## B. Θεωρία

### 3. Ερωτήσεις (2. Ερωτήσεις με μεταβλητές)

- Η ερώτηση `parent(tom,X)` στην Prolog μεταφράζεται ως «βρες όλες τις τιμές για την μεταβλητή `X`, ώστε το `parent(tom,X)` να είναι αληθές.
- Πατώντας ερωτηματικό μετά από κάθε τιμή που μας επιστρέφει η κονσόλα, μας εμφανίζει επόμενες τιμές που επαληθεύουν την ερώτηση έως ότου να μην υπάρχουν άλλες τέτοιες τιμές.
- Η σειρά με την οποία εμφανίζονται τα αποτελέσματα στην κονσόλα, έχει να κάνει με την σειρά που καταγράψαμε τα γεγονότα στο πρόγραμμά μας.

## B. Θεωρία

### 3. Ερωτήσεις (3. Ανώνυμες Μεταβλητές)

- Η ανώνυμη μεταβλητή συμβολίζεται με underscore `'_'` και παίζει ακριβώς τον ίδιο ρόλο με μια μεταβλητή.
- Την χρησιμοποιούμε όταν δεν θέλουμε να μας επιστραφούν οι συγκεκριμένες τιμές που αντιστοιχούν στην μεταβλητή αλλά θέλουμε να μας επιστραφεί αν υπάρχουν τιμές που επαληθεύουν την ερώτηση.
- Π.χ. η ερώτηση:
 

```
?- parent(tom, _).
```

  - Θα απαντήσει απλά `true`, διότι υπάρχουν τιμές που μπορούν να ανατεθούν στην μεταβλητή, ώστε να επαληθευτεί η σχέση.
  - Έτσι η παραπάνω ερώτηση δεν είναι πλέον «ποιοι είναι τα παιδιά του `tom`», αλλά είναι «έχει ο `tom` παιδιά;»
- Ενώ η ερώτηση:
 

```
?- parent(_, pat).
```

  - είναι «έχει η `pat` γονέα;» και θα απαντηθεί `true`.

## B. Θεωρία

### 3. Ερωτήσεις (4. Σύνθετες Ερωτήσεις)

- Για να κάνουμε πιο σύνθετες ερωτήσεις μπορούμε να χρησιμοποιήσουμε τους συνήθεις λογικούς τελεστές AND, OR και NOT

| ΤΕΛΕΣΤΗΣ | Συμβολισμός     |
|----------|-----------------|
| AND      | , (κόμμα)       |
| OR       | ; (ερωτηματικό) |
| NOT      | \+              |

- Παραδείγματα:

```
?- \+parent(tom, _).
false.
```

```
?- parent(X,tom);parent(tom,X).
X = bob ;
X = john.
```

```
?- parent(X,bob),parent(X,john).
X = tom ;
false.
```

## B. Θεωρία

### 4. Κανόνες (1. ορισμός κανόνα)

- Ένας κανόνας είναι ένας τρόπος για να ορίσουμε μία σχέση μέσω άλλων σχέσεων.
- Ένας κανόνας γράφεται με το σύμβολο `:-` (που διαβάζεται «αληθεύει αν»)
 

```
Όνομα-Σχέσης (Ορίσματα) :- Υποθέσεις
```
- Για παράδειγμα μπορούμε να ορίσουμε τη σχέση `grandparent` να αληθεύει αν το 1<sup>ο</sup> όρισμα είναι παππούς (ή γιαγιά) του 2<sup>ου</sup> ως εξής:
 

```
grandparent(X,Y) :-
    parent(X,Z),
    parent(Z,Y).
```
- Με βάση το συντακτικό που έχουμε ορίσει ο παραπάνω κανόνας διαβάζεται:
  - Η σχέση `grandparent(X,Y)` αληθεύει αν το `X` είναι γονέας του `Z` ΚΑΙ το `Z` είναι γονέας του `Y`.



## B. Θεωρία

### 4. Κανόνες (2. Ενσωμάτωση στο πρόγραμμα)

- Οι κανόνες γράφονται αμέσως μετά τα γεγονότα στο αρχείο προγράμματος
- Ενσωματώνονται στην SWI-Prolog αφού επιλέξουμε File->Consult στο αρχείο πηγαίου κώδικα (Προσοχή αν επαναφορτώνουμε αρχείο: επιλέγουμε File->Reload modified files)
- Θεωρείται καλή προγραμματιστική πρακτική να αναφέρουμε τους κανόνες μετά τα γεγονότα στο αρχείο προγράμματος.
- Έπειτα η κονσόλα μπορεί να μας απαντήσει και στα ερωτήματα που θέτουμε με χρήση του κατηγορήματος που έχουμε ορίσει.



## B. Θεωρία

### 4. Κανόνες (2. Ενσωμάτωση στο πρόγραμμα)

- Από τη στιγμή που έχουμε ορίσει ένα κατηγορημα στο «πρόγραμμά μας» μπορούμε να το χρησιμοποιήσουμε ως ένα οποιοδήποτε άλλο κατηγορημα.
- Παραδείγματα:

- Η ερώτηση «έχει ο tom εγγόνια διατυπώνεται ως εξής:»

```
?- grandparent(tom, _).
true
```

- Ενώ η ερώτηση: «ποια είναι τα εγγόνια του tom;»

```
?- grandparent(tom, X).
X=jim ;
X=kim ;
X=sam.
```



## B. Θεωρία

### 4. Κανόνες (3. Αναδρομικοί Κανόνες)

- Πως μπορούμε να ορίσουμε έναν κανόνα που να αληθεύει αν το 1<sup>ο</sup> όρισμα είναι πρόγονος του 2<sup>ου</sup> ορίσματος;

```
ancestor(X, Y) :-
    parent(X, Y).
ancestor(X, Y) :-
    parent(X, Z), parent(Z, Y).
ancestor(X, Y) :-
    parent(X, Z), parent(Z, W), parent(W, Y).
```

- Προφανώς δεν μπορούμε να κάνουμε αυτήν την καταγραφή για οσαδήποτε επίπεδα.
- Η λύση έρχεται με την **αναδρομή**! Μπορούμε να ορίσουμε αναδρομικά την σχέση πρόγονος ως εξής:
  - Αληθεύει αν ο X είναι ο γονέας του Y ή
  - Αληθεύει αν ο X είναι ο γονέας ενός Z, ο οποίος είναι πρόγονος του Y.



## B. Θεωρία

### 4. Κανόνες (3. Αναδρομικοί Κανόνες)

- Έτσι ορίζουμε την σχέση πρόγονος ως εξής:

```
ancestor(X, Y) :-
    parent(X, Y).
ancestor(X, Y) :-
    parent(X, Z), ancestor(Z, Y).
```

- Είναι συνήθης πρακτική όταν γράφουμε την αναδρομή να γράφουμε πρώτα τον κανόνα τερματισμού της αναδρομής και έπειτα τον αναδρομικό κανόνα.
- Ας δούμε πως θα εκτελέσει η prolog το ερώτημα ancestor(tom, pat).



## B. Θεωρία

### 4. Κανόνες (3. Εκτέλεση Προγράμματος)

Ασκηση: Δώστε το πλήρες δένδρο εκτέλεσης του ερωτήματος ? - **ancestor(tom,pat)**.