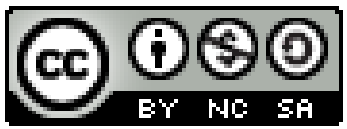
ΑΡΙΣΤΟΤΕΛΕΙΟ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΘΕΣΣΑΛΟΝΙΚΗΣ

ΑΝΟΙΚΤΑ
ΑΚΑΔΗΜΑΙΚΑ
ΜΑΘΗΜΑΤΑ

# Συστήματα Γνώσης

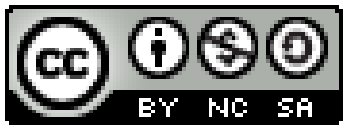
Πρακτικό Κομμάτι Μαθήματος
Πρότυπα Γεγονότων

Νίκος Βασιλειάδης, Αναπλ. Καθηγητής
Τμήμα Πληροφορικής




# Άδειες Χρήσης

- Το παρόν εκπαιδευτικό υλικό υπόκειται σε άδειες χρήσης Creative Commons.
- Για εκπαιδευτικό υλικό, όπως εικόνες, που υπόκειται σε άλλου τύπου άδειας χρήσης, η άδεια χρήσης αναφέρεται ρητώς.



# Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «Ανοικτά Ακαδημαϊκά Μαθήματα στο Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

# Πρότυπα Γεγονότων

# Μεγάλα/Σύνθετα Γεγονότα

- Σε μεγάλα προγράμματα χρειάζεται να αναπαρασταθεί η πληροφορία με μεγάλα ή σύνθετα γεγονότα
- Π.χ. βάση δεδομένων μαθητών:

```
(student name <name> surname <surname>
         sex <sex> age <age>
         classes <classes>)

(student name john surname ref sex
  male
         age 28
         classes math physics chem)
```

# Χρήση Μεγάλων Γεγονότων

- Όταν κάποιο μεγάλο γεγονός αλλάζει μορφή, τότε πρέπει να γίνουν σημαντικές αλλαγές σε **κάθε κανόνα** που αναφέρει το γεγονός στη συνθήκη του, καθώς το γεγονός πρέπει να αναφέρεται **κάθε φορά** με **όλες τις παραμέτρους** του

```
(defrule print-students
  (student name ?n surname ?sn
           sex ?s age ?a classes $?cl)
  =>
  (printout t "Student: " ?n crlf)
)
```

# Πρότυπα Γεγονότων

- Το CLIPS προσφέρει τα ***πρότυπα γεγονότων*** (***templates***) ως εναλλακτικό τρόπο δημιουργίας και διαχείρισης μεγάλων γεγονότων
- Τα templates είναι μια δομή με την οποία μπορεί να οριστεί η **μορφή** που θα έχουν τα γεγονότα σε ένα πρόγραμμα
- Κάθε πρότυπο έχει ένα σύνολο από ***ιδιότητες*** (***slots***), στις οποίες μπορούν να ανατεθούν τιμές αυτόνομα
  - Μπορεί να ορισθούν και οι τύποι των τιμών για να γίνονται οι απαραίτητοι έλεγχοι.

# Συνάρτηση deftemplate

```
(deftemplate <template name>
  (slot <slotname1>
        (type <type1>))
  (multislot <slotname2>
        (type <type2>))
  ...
  (slot <slotnameN>
        (type <typeN>))
)
```

# Συνάρτηση deftemplate

- **slotnameN** είναι το όνομα της ιδιότητας
- **(type <typeN>)** καθορίζει τον τύπο της τιμής της συγκεκριμένης ιδιότητας
  - Ο καθορισμός τύπου είναι προαιρετικός
  - Αν δεν ορισθεί για κάποια ιδιότητα αυτή μπορεί να δεχθεί τιμή οποιουδήποτε τύπου
- Υπάρχουν δύο είδη ιδιοτήτων:
  - **slot** που μπορούν να πάρουν σαν τιμή μόνο ένα σύμβολο ή αριθμό (μονότιμες μεταβλητές)
  - **multislot** οι οποίες δέχονται πολλαπλές τιμές (μεταβλητές πολλαπλών τιμών)

# Παράδειγμα προτύπου μαθητή

```
(deftemplate student
     (slot name)
     (slot surname)
     (slot sex)
     (slot age)
     (multislot classes)
)

CLIPS>(assert
        (student
          (name john)
          (surname smith)
          (sex male)
          (age 30)
          (classes physics computer)))
```

# Παράδειγμα Κανόνα με template

```
(defrule print-students
  (student (name ?name) (sex male))
  =>
  (printout t "Student: " ?name crlf))
```

- Στις συνθήκες των κανόνων δε χρειάζεται να γραφεί ολόκληρο το γεγονός, αλλά μόνο το **όνομα του προτύπου** και τα **ονόματα των ιδιοτήτων** που ενδιαφέρουν

# Διαθέσιμοι Τύποι Ιδιοτήτων

| Τύπος | Η ιδιότητα μπορεί να περιέχει |
|---|---|
| SYMBOL | σύμβολα |
| STRING | Αλφαριθμητικά |
| LEXEME | σύμβολα ή αλφαριθμητικά |
| INTEGER | ακέραιες τιμές |
| FLOAT | πραγματικές τιμές |
| NUMBER | ακέραιες ή πραγματικές τιμές |
| ?VARIABLE | τιμές οποιουδήποτε τύπου |

# Παράδειγμα προτύπου μαθητή με τύπους

```
(deftemplate student
  (slot name (type SYMBOL))
  (slot surname (type SYMBOL))
  (slot sex (type SYMBOL))
  (slot age (type INTEGER))
  (multislot classes (type SYMBOL))
)
```

# Απαρίθμηση επιτρεπτών τιμών

- Για να απαριθμήσουμε τις επιτρεπτές τιμές που μπορεί να πάρει μια ιδιότητα, χρησιμοποιούμε τη δήλωση:
  **(allowed-*<prefix>* *<values>*)**

```
(allowed-symbols <symbols>)
(allowed-strings <strings>)
(allowed-lexemes <symbols or strings>)
(allowed-integers <integers>)
(allowed-floats <floats>)
(allowed-numbers <numbers or floats>)
(allowed-values <values>)
```

# Παράδειγμα προτύπου μαθητή με απαρίθμηση τιμών

```
(deftemplate student
  ...
  (slot sex
     (type SYMBOL)
     (allowed-symbols male
  female))
  ...
)
```

# Περιορισμός εύρους αριθμητικών τιμών

- Για να καθορίσουμε το επιτρεπτό εύρος τιμών που επιδέχεται μια ιδιότητα με αριθ-μητικές τιμές χρησιμοποιούμε τη δήλωση:
  **(range *<min value>* *<max value>*)**

```
(deftemplate student
   ....
   (slot age (type INTEGER)
                (range 18 60))
   ...
)
```

# Περιορισμός εύρους αριθμητικών τιμών

- Εάν η ιδιότητα δεν πρέπει να έχει άνω ή κάτω όριο τότε εισάγεται το σύμβολο **?VARIABLE** στην αντίστοιχη θέση

```
(deftemplate student
   ....
   (slot age (type INTEGER)
             (range 18 ?VARIABLE))
   ...
)
```

# Περιορισμός πλήθους συμβόλων

- Στις ιδιότητες πολλαπλών τιμών (*multislots*) περιορίζουμε το πλήθος των συμβόλων που μπορούν να δοθούν σαν τιμή, με τη δήλωση:
  **(cardinality *<min>* *<max>*)**

```
(deftemplate student
   ...
   (multislot classes (type SYMBOL)
                      (cardinality 1 4))
   ...
)
```

# Δήλωση προκαθορισμένων τιμών (defaults)

- Σε όλες τις ιδιότητες μπορούμε να προκαθορίσουμε τιμές (default) με τη δήλωση:
  **(default *<value>*)**

```
(deftemplate student
   ...
   (slot age (type INTEGER)
             (default 18))
   ...
)
```

# Σύμβολα με ειδική σημασία στη δήλωση `default`

**`(default ?DERIVE)`**

- Δίνεται σαν τιμή στην ιδιότητα μια από τις επιτρεπόμενες τιμές, όπως αυτές προκύπτουν από τους περιορισμούς που υπάρχουν για την ιδιότητα
- Αν δεν υπάρχει δήλωση **`default`**, τότε υπονοείται η δήλωση
  **`(default ?DERIVE)`**

# Παράδειγμα προτύπου μαθητή με δήλωση `default`

```
(deftemplate student
   ...
   (slot age (type INTEGER)
             (range 18 ?VARIABLE)
             (default ?DERIVE))
   ...
)
```

- Αν εισαχθεί γεγονός μαθητή, χωρίς τιμή για το **age**, θα πάρει την τιμή 18

# Παράδειγμα προτύπου μαθητή με δήλωση `default`

```
(deftemplate student
  ...
   (slot sex
     (type SYMBOL)
     (allowed-symbols male female)
     (default ?DERIVE))
  ...
)
```

- Αν εισαχθεί γεγονός μαθητή, χωρίς τιμή για το **`sex`**, θα πάρει την τιμή **`male`**

# Παράδειγμα προτύπου μαθητή με δήλωση `default`

```
(deftemplate student
   ...
   (slot name (type SYMBOL)
              (default ?DERIVE))
   ...
)
```

- Αν εισαχθεί γεγονός μαθητή χωρίς όνομα, τότε η ιδιότητα **`name`** θα πάρει την τιμή **`nil`**

# Σύμβολα με ειδική σημασία στη δήλωση `default`

`(default ?NONE)`

- Δηλώνει ότι δεν υπάρχει προκαθορισμένη τιμή για την ιδιότητα
  - Πρέπει οπωσδήποτε να δοθεί τιμή στη συγκεκριμένη ιδιότητα κατά τη δημιουργία του αντίστοιχου γεγονότος
  - Αλλιώς το γεγονός δε θα εισαχθεί στη λίστα γεγονότων

# Παράδειγμα προτύπου μαθητή με δήλωση `default`

```
(deftemplate student
  ...
  (slot name (type SYMBOL)
             (default ?NONE))
  ...
)
```

- Δεν μπορεί να εισαχθεί γεγονός μαθητή, αν δεν υπάρχει τιμή για την ιδιότητα **`name`**

# Έλεγχος τιμών

- Υπάρχουν 2 επίπεδα ελέγχου τιμών που εισάγο-νται στα πρότυπα γεγονότων, το στατικό και το δυναμικό
- Ο **στατικός έλεγχος τιμών** αφορά τα γεγονότα τα οποία εισάγονται στο σύστημα από αρχείο, μέσα από τις δηλώσεις **`deffacts`**, **`defrule`**, κλπ.
  - Ενεργοποιείται από την εντολή:

```
(set-static-constraint-checking TRUE|FALSE
)
```

  - Η τρέχουσα τιμή επιστρέφεται με την εντολή:

```
(get-static-constraint-checking)
```

# Έλεγχος τιμών

- Στο **δυναμικό έλεγχο** κάθε γεγονός που εισάγεται στη λίστα γεγονότων ελέγχεται για την εγκυρότητα των τιμών του, ακόμη και αν η εισαγωγή γίνεται κατά την εκτέλεση του προγράμματος
  - Ενεργοποίηση/απενεργοποίηση :

```
(set-dynamic-constraint-checking
TRUE|FALSE )
```

  - Τρέχουσα τιμή:

```
(get-dynamic-constraint-checking)
```

- Σε περίπτωση ύπαρξης σφάλματος στις τιμές που δίνονται σε κάποια ιδιότητα επιστρέφεται μήνυμα λάθους και σταματά η εκτέλεση των κανόνων

# Αλλαγή τιμής μιας ιδιότητας

```
(modify <fact-index>
        (<slot> <νέα τιμή slot>))
```

- Αλλάζει την τιμή της ιδιότητας *<slot>* σε *<νέα τιμή slot>* στο γεγονός με αριθμό *fact-index*
- Δίνει τη δυνατότητα αλλαγής της τιμής κάθε ιδιότητας ενός γεγονότος που ακολουθεί κάποιο πρότυπο, **ανεξάρτητα** από τις άλλες ιδιότητες
- Η χρήση του χαρακτηριστικού αριθμού για την εφαρμογή της εντολής επιβάλλει τη χρήση του τελεστή **<-** στις συνθήκες του κανόνα

# Παράδειγμα χρήσης `modify`

- Ο κανόνας αλλάζει ταυτόχρονα τις τιμές των ιδιοτήτων **`name`** και **`classes`** στα γεγονότα **`student`** που βρίσκονται στη μνήμη εργασίας

```
(defrule change-information
  ?x <- (student (name ?name))
  =>
   (modify ?x
      (name noname)
      (classes (create$ math physics
                        chem)))
)
```

# Εντολή `modify`

- Η εντολή **`modify`** αφαιρεί το παλαιό γεγονός από τη λίστα και προσθέτει σε αυτή ένα καινούργιο με τις απαραίτητες αλλαγές
- Αυτό σημαίνει ότι ο προηγούμενος κανόνας θα εκτελείται επ' άπειρον, καθώς θα υπάρχει πάντα σε κάθε κύκλο ένα "νέο" γεγονός (με νέο **`fact index`**) το οποίο θα ικανοποιεί τις συνθήκες του

# Ορθότερο παράδειγμα χρήσης modify

```
(defrule change-information
  ?x <- (student
            (name ?name&~noname))
  =>
  (modify ?x
     (name noname)
      (classes (create$ math
  physics
                                chem)))
)
```

# Γεγονότα χωρίς template

- Τα γεγονότα χωρίς την χρήση template που εξετάσαμε έως σήμερα ονομάζονται ***διατεταγμένα γεγονότα*** (***ordered facts***)
- Στην πραγματικότητα έχουν μόνο ένα «***κρυφό***» (***implied***) slot για να αποθηκεύουν όλες τις παραμέτρους του γεγονότος
  - Το slot αυτό έχει όνομα **`implied`** και είναι **`multifield`**

# Παράδειγμα

```
CLIPS> (assert (person John Smith age 27))
CLIPS> (defrule my-rule
             (person $?x)
             =>
             (printout t "All data for person: " $?x
                         crlf))
CLIPS> (run)
All data for person: (John Smith age 27)
```

- Το slot **implied** δεν αναφέρεται ρητά
  - Απόδειξη ότι το slot **implied** υπάρχει!

```
CLIPS> (deftemplate-slot-names person)
(implied)
```

ΑΡΙΣΤΟΤΕΛΕΙΟ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΘΕΣΣΑΛΟΝΙΚΗΣ

ΑΝΟΙΚΤΑ
ΑΚΑΔΗΜΑΙΚΑ
ΜΑΘΗΜΑΤΑ

# Τέλος Ενότητας

Επεξεργασία: Εμμανουήλ Ρήγας

Θεσσαλονίκη, 17/3/2014



Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο

ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ & ΘΡΗΣΚΕΥΜΑΤΩΝ, ΠΟΛΙΤΙΣΜΟΥ & ΑΘΛΗΤΙΣΜΟΥ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ

ΕΣΠΑ 2007-2013
πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης