Εργαστήριο Λογικού Προγραμματισμού

Μανόλης Μαρακάκης, Καθηγητής
mmarak@cs.hmu.gr

Τμήμα Ηλεκτρολόγων Μηχανικών & Μηχανικών Υπολογιστών
Σχολή Μηχανικών
Ελληνικό Μεσογειακό Πανεπιστήμιο

Κατασκευή Μεγάλων Προγραμμάτων σε Prolog

Ενότητα 8.5:

8. Προγραμματιστικές Τεχνικές
8.5 Δομές Δεδομένων σε Prolog

8. Προγρ/κές Τεχνικές: Δομές Δεδομένων σε Prolog.

- 8.5 Δομές Δεδομένων σε Prolog
  - 8.5.1 Εισαγωγή
  - 8.5.2 Ακολουθίες/Sequences
  - 8.5.3 Σύνολα/Sets
  - 8.5.4 Στοίβες/Stacks
  - 8.5.5 Ουρές/Queues

8.5. Δομές Δεδομένων σε Prolog

8.5.1. Εισαγωγή

- Οι **δομές δεδομένων** που επιλέγονται καθορίζουν τη **σαφήνεια** και την **αποτελεσματικότητα** ενός προγράμματος ή αλγορίθμου.
  - Η επιλογή της κατάλληλης δομής δεδομένων θα οδηγήσει στη δημιουργία **αποτελεσματικού** και **ευκρινούς** προγράμματος ή αλγορίθμου.
  - Το αντίθετο θα οδηγήσει σε πρόγραμμα/αλγόριθμο ο οποίος θα είναι **δυσνόητος** και **μη αποτελεσματικός**. Σ' αυτήν την περίπτωση θα είναι δύσκολο να διορθωθούν λάθη στο τελικό πρόγραμμα καθώς και ν' αποδειχθεί η ορθότητα του.
  - Η **οργάνωση των δεδομένων** πάνω στα οποία λειτουργεί ένα πρόγραμμα επηρεάζει σημαντικά την κατασκευή του προγράμματος.
- Συνεπώς, η ***σαφήνεια*** και η ***αποτελεσματικότητα*** του προγράμματος εξαρτάται από την οργάνωση, τη δομή των δεδομένων. Κάθε εφαρμογή με βάση τις ιδιαιτερότητές της χρειάζεται και άλλη δομή δεδομένων.
  - Για παράδειγμα, μια εφαρμογή αναζήτησης σε χώρο καταστάσεων χρειάζεται δένδρο ή γράφο και στοίβα για οπισθοδρόμηση. Η λύση σ' αυτό το πρόβλημα δίνεται από τις δομές δεδομένων.

8.5. Δομές Δεδομένων σε Prolog

8.5.1. Εισαγωγή

- **Ορισμός:** ***Τύπος δεδομένων*** *(data type)* είναι ***ένα σύνολο από τιμές*** **(αντικείμενα)** και ***ένα σύνολο πράξεων*** οι οποίες εφαρμόζονται στις τιμές του τύπου.
- Κάθε ***πράξη*** ενός τύπου δεδομένων μπορούμε να τη δούμε ως μια συνάρτηση η οποία έχει κάποια ορίσματα και σε κάθε κλήσης της επιστρέφει μια τιμή από το πεδίο τιμών της.
- Ένας **τύπος δεδομένων**, δηλαδή **οι τιμές** και **οι πράξεις του**, ορίζεται μ' ένα ***σύνολο αξιωμάτων***. Τα **αξιώματα** ορίζουν **τις ιδιότητες** και **τα χαρακτηριστικά των τιμών και των πράξεων** του τύπου δεδομένων. Αυτός ο τρόπος ορισμού ενός τύπου είναι ανεξάρτητος από τον τρόπο υλοποίησής του.

8.5. Δομές Δεδομένων σε Prolog

8.5.1. Εισαγωγή

- **Ορισμός:** ***Δομές δεδομένων*** ονομάζεται το επιστημονικό πεδίο το οποίο μελετά τη ***χρήση (τις προδιαγραφές)*** και την ***υλοποίηση*** των τύπων δεδομένων.
- Η ***αναπαράσταση ενός τύπου δεδομένων*** καθορίζει πώς γίνεται η κωδικοποίησή του στις δομές δεδομένων μιας γλώσσας προγραμματισμού στην οποία πρόκειται να υλοποιηθεί.
- Η ***υλοποίηση ενός τύπου δεδομένων*** περιλαμβάνει ένα σύνολο από αλγόριθμους οι οποίοι υλοποιούν τις πράξεις του για κάποιο τρόπο αναπαράστασής του [Horowitz, Sahni, 1976].
- Κάποια ***δεδομένα,*** αν τα μελετήσουμε από την άποψη του **χρήστη**, θα τα δούμε ως ***τύπο δεδομένων*** ενώ, εάν τα μελετήσουμε από την άποψη του **προγραμματιστή**, θα τα δούμε ως ***δομή δεδομένων***.

8.5. Δομές Δεδομένων σε Prolog

8.5.1. Εισαγωγή

- Κατά τον Horowitz και Sahni μια δομή δεδομένων πρέπει αρχικά **να σχεδιαστεί** ώστε να γνωρίζουμε «***τί»*** κάνει, αλλά όχι κατ' ανάγκη «***πώς»*** το κάνει.
- Χρειάζεται να διαιρέσουμε το έργο των δομών δεδομένων, στις ***προδιαγραφές (specifications)*** τους, δηλαδή το «***τί***» και στην ***υλοποίηση (implementation)*** τους, το «***πώς***» [Horowitz, Sahni, 1976].
- Σύμφωνα με τον ορισμό που δίνουμε στις δομές δεδομένων η ***χρήση*** μιας δομής δεδομένων αντιστοιχεί στις ***προδιαγραφές*** της και αυτό ενδιαφέρει τον ***χρήστη προγραμματιστή*** (π.χ. χρήστης προγραμματιστής Prolog). Ενώ, η ***υλοποίηση*** ενός τύπου δεδομένων στη γλώσσα υλοποίησης αφορά τον ***προγραμματιστή υλοποίησης*** του τύπου δεδομένων (όχι κατ' ανάγκη προγραμματιστής Prolog).

8.5. Δομές Δεδομένων σε Prolog

8.5.1. Εισαγωγή

- Σε αυτή την ενότητα θα ορίσουμε **περιγραφικά**, όχι τυπικά, τύπους δεδομένων και θα τους υλοποιήσουμε σε Prolog για τις ανάγκες του ***χρήστη προγραμματιστή*** Prolog. Ουσιαστικά από τις **προδιαγραφές** ενός τύπου δεδομένων μας ενδιαφέρουν **οι προδιαγραφές των πράξεων του** και από την **υλοποίηση, η υλοποίηση των πράξεων του**.
- *Η **υλοποίηση ενός τύπου** δεδομένων σε Prolog περιλαμβάνει ένα σύνολο από **κατηγορήματα τα οποία υλοποιούν τις πράξεις του τύπου δεδομένων** για κάποιο τρόπο αναπαράστασής τους.*
- Ο **όρος (term**) είναι ο κύριος τρόπος αναπαράστασης δεδομένων σε Prolog και ο **όρος της λίστας** είναι **η κύρια δομή δεδομένων που διαθέτει η Prolog**.
- Παραδείγματα, τύπων δεδομένων είναι τα εξής: ***οι φυσικοί αριθμοί**, **οι ακέραιοι, οι πραγματικοί,** **οι συμβολοσειρές, η στοίβα, η λίστα**, **η ουρά, η ακολουθία, τα σύνολα,** **τα πολυσύνολα (multisets), τα δένδρα, τα δυαδικά δένδρα,** **οι σχέσεις, τα καρτεσιανά γινόμενα, οι πλειάδες (tuples),** **οι κατευθυνόμενοι γράφοι, οι μη-κατευθυνόμενοι γράφοι** και άλλα.*

8.5. Δομές Δεδομένων σε Prolog

8.5.1. Εισαγωγή

- Οι **ενσωματωμένοι τύποι δεδομένων** στη Prolog είναι οι ***αριθμοί*** και οι ***όροι***. Για τους υπόλοιπους τύπους δεδομένων οι περισσότερες υλοποιήσεις της Prolog διαθέτουν modules (τμήματα) στα οποία περιλαμβάνουν υλοποιήσεις επιπλέον τύπων δεδομένων. Αυτά δεν είναι ενσωματωμένα στη γλώσσα αλλά βρίσκονται στη βιβλιοθήκη της. Θα πρέπει ο χρήστης να φορτώσει το αντίστοιχο module για να μπορεί να το χρησιμοποιήσει.
  - Δηλαδή, τα module έχουν κατηγορήματα τα οποία ορίζουν τις πράξεις της συγκεκριμένης δομής που υλοποιούν.
- Πρέπει να σημειώσουμε ότι οι υλοποιήσεις των τύπων δεδομένων που διαθέτουν οι Prolog στις βιβλιοθήκες τους μπορεί να μην έχουν υλοποιηθεί σε Prolog, όπως θα κάνουμε εμείς σε αυτή την ενότητα, αλλά στην γλώσσα υλοποίησης της Prolog που μπορεί να είναι η C ή κάποια άλλη γλώσσα προγραμματισμού.
- Ένας προγραμματιστής σε Prolog, θα μπορούσε να φτιάξει τη δική του βιβλιοθήκη με υλοποιήσεις των τύπων δεδομένων τις οποίες χρησιμοποιεί στις εφαρμογές του ώστε να κάνει επαναχρησιμοποίηση του κώδικά του και σε άλλες εφαρμογές που θα αναπτύξει.

8.5. Δομές Δεδομένων σε Prolog

8.5.1. Εισαγωγή

- Σε αυτήν την ενότητα θα παρουσιάσουμε τις υλοποιήσεις σε Prolog κάποιων τύπων δεδομένων όπως ***των ακολουθιών***, ***των συνόλων***, ***της στοίβας*** *και* ***της ουράς***. Με αυτόν τον τρόπο μπορούν να υλοποιηθούν και άλλοι τύποι δεδομένων.
- Για κάθε πράξη δίνουμε τον ***τρόπο κλήσης της (mode)*** με τους συμβολισμούς που έχουν οριστεί στην Ενότητα 3.3 του βιβλίου του εργαστηρίου. Για παράδειγμα, **mode(g,g,a)** σημαίνει ότι το **1ο** και το **2ο** όρισμα πρέπει να είναι **πλήρως δεσμευμένα** ενώ το **3ο** όρισμα **μπορεί να έχει οποιαδήποτε δέσμευση**. Επίσης για κάθε όρισμα θα έχουμε τον τύπο των δεδομένων του.

8.5. Δομές Δεδομένων σε Prolog

8.5.1. Εισαγωγή

- Στους τύπους δεδομένων όπου υπάρχει η μεταβλητή «Τ» με ή χωρίς δείκτες λαμβάνεται ως ***μεταβλητή τύπου***.
  - Επιπλέον, εάν δύο ή περισσότερες μεταβλητές τύπων έχουν **διαφορετικό όνομα**, π.χ. T_1, T_2, T_3,κτλ, στον ορισμό του τύπου δεδομένων ενός κατηγορήματος, αυτό σημαίνει ότι **μπορεί να παριστούν τον ίδιο** ή **διαφορετικούς τύπους δεδομένων**.
  - Εάν δυο μεταβλητές τύπων έχουν **ίδιο όνομα** στον ορισμό του τύπου δεδομένων ενός κατηγορήματος, **σημαίνει ότι** αναφέρονται στον **ίδιο τύπο δεδομένων**.
- Έστω τα παρακάτω παραδείγματα.
  - 1) Η δήλωση του κατηγορήματος «**pred1(X:integer, S:set(integer))**» σημαίνει ότι το **1ο όρισμα** του κατηγορήματος «**pred1/2**» είναι το **X** και ο τύπος των δεδομένων του είναι **ακέραιοι αριθμοί**. Δηλαδή το **X** μπορεί να δεσμευτεί με έναν ακέραιο αριθμό. Το **2ο όρισμά** του είναι το **S** και ο τύπος των δεδομένων του είναι **σύνολα από ακεραίους αριθμούς**.
  - 2) Η δήλωση του κατηγορήματος «**pred2(X:T, S1:set(T),S2:set(T))**» σημαίνει ότι το **1ο όρισμα** του κατηγορήματος «**pred2/3**» είναι το **X** και είναι τύπου «**T**» δηλαδή ***μεταβλητή τύπου***. Το **2ο και το 3ο όρισμα** του κατηγορήματος είναι τύπου **«set(T)»**. Σε αυτό το παράδειγμα η ***μεταβλητή τύπου*** **«T»** παριστά **ίδιο τύπο δεδομένων**.

8.5. Δομές Δεδομένων σε Prolog

8.5.1. Εισαγωγή

- Για να τρέξετε τα παραδείγματα με τις πράξεις που παρουσιάζονται στη συνέχεια σε αυτήν την ενότητα θα πρέπει να ενσωματώσετε στο πρόγραμμα άλλες πράξεις από τύπους δεδομένων οι οποίοι αναφέρονται στην περιγραφή της πράξης. Για παράδειγμα, για την υλοποίηση της πράξης «**cart_prod(S1:set(T_1), S2 set(T_2), S1xS2:set(tuple(T_1,T_2))**» θα πρέπει να ενσωματώσετε στο πρόγραμμα που υλοποιεί αυτήν την πράξη άλλες πράξεις που χρησιμοποιούνται από τις **πλειάδες (tuples)** και από τα **σύνολα (sets)**.

8.5. Δομές Δεδομένων σε Prolog

8.5.2. Ακολουθίες/Sequences

- Μια ***ακολουθία (sequence)*** είναι μια συλλογή από στοιχεία στα οποία η σειρά και η πολλαπλότητα είναι σημαντικές [Morgan, 1998].
- **Ορισμός:** Έστω S μια συλλογή από στοιχεία. Μια ***ακολουθία (sequence)*** είναι μια συνάρτηση f της οποίας το **πεδίο ορισμού** είναι οι **φυσικοί αριθμοί**, ℵ={0,1,2,3,..}, ή {κ,κ+1,κ+2,..} όπου κ=1. Το **πεδίο τιμών** της συνάρτησης είναι **τα στοιχεία της συλλογής S**. Μια ακολουθία μπορεί να είναι πεπερασμένη οπότε το πεδίο ορισμού της είναι το **{κ,κ+1,κ+2,..,μ}** όπου το **κ** είναι **0** ή **1** και το **μ** είναι ένας φυσικός αριθμός [Grassmann, Tremblay, 1996], [Munro, 1992], [Ross, Wright, 1992], Δηλαδή, **η ακολουθία απεικονίζει τους φυσικούς αριθμούς στα στοιχεία της συλλογής S = $S_1,S_2,S_3,\ldots$** ως εξής: **$f(1)=S_1$, $f(2)=S_2$, $f(3)=S_3$, κτλ.**

8.5. Δομές Δεδομένων σε Prolog

8.5.2. Ακολουθίες/Sequences

- Για να υπάρχει διάκριση στο συμβολισμό μεταξύ **συνόλων { }**, **πολυσυνόλων [ ]** και **ακολουθιών** θα χρησιμοποιήσουμε για τις ακολουθίες τον εξής συμβολισμό **<….>**. Γενικότερα, όταν θα έχουμε μια δομή δεδομένων της οποίας **τα στοιχεία ακολουθούν μια διάταξη** θα χρησιμοποιούμε τα σύμβολα **<….>**. Τέτοιες δομές δεδομένων είναι για παράδειγμα οι **στοίβες**, οι **ουρές** και άλλες.
- Έστω για παράδειγμα η ακολουθία των περιττών φυσικών αριθμών **S = <1,3,5,..>={S_1,S_2,..,>** η οποία είναι **μη πεπερασμένη**, **S_v=2κ+1** όπου **κ** φυσικός αριθμός. **Μας ενδιαφέρουν οι πεπερασμένες ακολουθίες**.
- Οι ακολουθίες ως δομές δεδομένων είναι κοντά στους «πίνακες» των συμβατικών γλωσσών προγραμματισμού και στις «λίστες» των γλωσσών Τεχνητής Νοημοσύνης.

8.5. Δομές Δεδομένων σε Prolog

8.5.2. Ακολουθίες/Sequences

- Ακολουθούν μερικά παραδείγματα ακολουθιών στα οποία φαίνονται τα στοιχεία της συλλογής **S**, και η απεικόνιση των φυσικών αριθμών στα στοιχεία της συλλογής **S** δηλαδή η σειρά την οποία έχουν τα στοιχεία της συλλογής **S**.
  - S = <8,5,8,6> = < S_1,S_2,S_3,S_4>, όπου f(1)= S_1=8, f(2)= S_2=5, f(3)= S_3=8, f(4) = S_4 = 6.
  - S = <cba,bb,aa,ba,cb,bc> = <S_1,S_2,S_3,S_4,S_5,S_6>, όπου f(1)= S_1=cba, f(2)= S_2=bb, f(3)= S_3= aa, κτλ.
  - S = <0,1,0,0,1,1> = <S_1,S_2,S_3,S_4,S_5,S_6>, όπου f(1)= S_1=0, f(2)= S_2=1, f(3)= S_3= 0, κτλ.
  - S = <a,b,1,1,a> =< S_1,S_2,S_3,S_4,S_5>, όπου f(1)= S_1=a, f(2)= S_2=b, f(3)= S_3= 1, κτλ.
- Οι ακολουθίες έχουν υλοποιηθεί με τις λίστες της Prolog.

8.5. Δομές Δεδομένων σε Prolog

8.5.2. Ακολουθίες: Ορισμός και υλοποίηση πράξεων.

- Στη συνέχεια παρουσιάζουμε για μερικές πράξεις των ακολουθιών τον περιγραφικό ορισμό της καθώς και την υλοποίηση της σε Prolog. Όλες τις πράξεις θα τις βρείτε στο βιβλίο του εργαστηρίου.
- **Πρόγραμμα 8.22: Άδεια ακολουθία.** Το κατηγόρημα **empty_seq(Q:seq(T))** είναι αληθές εάν η ακολουθία **Q** είναι άδεια, mode(a).

```
empty_seq([]).
```

- **Πρόγραμμα 8.23: Η κεφαλή ακολουθίας.** Το κατηγόρημα **head(Q:seq(T), Elem:T)** είναι αληθές εάν **Elem** είναι το πρώτο στοιχείο της ακολουθίας **Q**, λαμβάνοντας ότι η **Q** δεν είναι άδεια, mode(g, a).

```
head([H|T], H).
```

- **Πρόγραμμα 8.24: Η ουρά ακολουθίας.** Το κατηγόρημα **tail(Q:seq(T), Tail:seq(T)),** είναι αληθές εάν **Tail** είναι τα υπόλοιπα στοιχεία της ακολουθίας **Q** εκτός από το πρώτο, λαμβάνοντας ότι η **Q** δεν είναι άδεια, mode(g, a).

```
tail([H|T], T).
```

8.5. Δομές Δεδομένων σε Prolog

8.5.2. Ακολουθίες: Ορισμός και υλοποίηση πράξεων (συνέχεια).

- **Πρόγραμμα 8.25: Δημιουργία ακολουθίας.** Το κατηγόρημα **seq_cons(Tail:seq(T), Head:T, Q:seq(T)),** είναι αληθές εάν **Q** είναι μια ακολουθία με πρώτο στοιχείο το **Head** και τα υπόλοιπα στοιχεία στο **Tail,** mode(g,g,a).

```
seq_cons(Q, X, [X|Q]).
```

- **Πρόγραμμα 8.26: Όλα τα στοιχεία ακολουθίας πλην του τελευταίου.** Το κατηγόρημα **front(Q:seq(T), Front:seq(T)),** είναι αληθές εάν **Front** έχει όλα τα στοιχεία της ακολουθίας **Q** εκτός από το τελευταίο, λαμβάνοντας ότι η **Q** δεν είναι άδεια, mode(g,a).

```
front([H], []).

front([H|T], [H|Rest])  :-  front(T, Rest).
```

- **Πρόγραμμα 8.27: Το τελευταίο στοιχείο ακολουθίας.** Το κατηγόρημα **last(Q:seq(T), Last:T),** είναι αληθές εάν **Last** έχει το τελευταίο στοιχείο της ακολουθίας **Q**, λαμβάνοντας ότι η **Q** δεν είναι άδεια, mode(g,a).

```
last([H], H).

last([H|T], Last) :-  last(T, Last).
```

8.5. Δομές Δεδομένων σε Prolog

8.5.2. Ακολουθίες: Ορισμός και υλοποίηση πράξεων (συνέχεια).

- **Πρόγραμμα 8.28: Καταχώρηση στοιχείου στο τέλος ακολουθίας.** Το κατηγόρημα **add_last_elem(Q:seq(T), Elem:T, R:seq(T)),** είναι αληθές εάν **R** είναι η ακολουθία **Q** με το στοιχείο **Elem** στο τέλος της **Q**, mode(g,g, a). Παράδειγμα: add_last_elem(<a,a,b,bb,c>, dd, R), R = <a,a,b,bb,c,dd>

```
add_last_elem([], Elem, [Elem]).

add_last_elem([H|T], Elem, [H|Rest]) :-   add_last_elem(T, Elem, Rest).
```

- **Πρόγραμμα 8.29: Καταχώρηση N εμφανίσεων στοιχείου στη κεφαλή ακολουθίας.** Το κατηγόρημα **add_head_N_occ(Q:seq(T), Elem:T, N:natural, R:seq(T)),** είναι αληθές εάν **R** είναι είναι η ακολουθία **Q** με **N** εμφανίσεις του στοιχείου **Elem** στην κεφαλή της ακολουθίας **Q**. Τα στοιχεία της ακολουθίας **Q** μετακινούνται **N** θέσεις προς τα δεξιά. mode(g,g, g,a). «natural» σημαίνει ότι ο τύπος δεδομένων του αντίστοιχου ορίσματος είναι οι φυσικοί αριθμοί. Παράδειγμα: add_head_N_occ(<a,a,b,bb,c>, ee, 3, R), R = <ee,ee,ee,a,a,b,bb,c>.

```
add_head_N_occ(Q, Elem, 0, Q).

add_head_N_occ(Q, Elem, N, [Elem|Rest]) :-
        N1 is N-1,
        add_head_N_occ(Q, Elem, N1, Rest).
```

8.5. Δομές Δεδομένων σε Prolog

8.5.2. Ακολουθίες: Παράδειγμα.

- Θα μελετήσουμε την υλοποίηση της ταξινόμησης με παρεμβολή χρησιμοποιώντας ακολουθίες.
  - Ο αλγόριθμος της **ταξινόμησης με παρεμβολή** **παίρνει ένα-ένα τα στοιχεία από την ακολουθία S και τα τοποθετεί στην ταξινομημένη ακολουθία Q στη σωστή θέση χωρίς να επηρεάζει τη σειρά των ήδη ταξινομημένων στοιχείων.**
- Το κατηγόρημα **ins_sort(S:seq(T), Q:seq(T))**, είναι αληθές εάν η ακολουθία **Q** είναι η ακολουθία **S** με τα στοιχεία της ταξινομημένα.
  - Για την υλοποίηση του **ins_sort/2** εκτός από τις πράξεις των ακολουθιών χρησιμοποιούμε και το κατηγόρημα **ins_elem/3** το οποίο ορίζεται ως εξής. Το κατηγόρημα **ins_elem(E:T,Q1:seq(T), Q:seq(T))** είναι αληθές εάν η ακολουθία **Q** είναι η ταξινομημένη ακολουθία **Q1** αυξημένη με το στοιχείο **E** χωρίς να επηρεάζει την υπάρχουσα ταξινόμηση.

8.5. Δομές Δεδομένων σε Prolog

8.5.2. Ακολουθίες: Παράδειγμα.

```
ins_sort(S, Q) :-
    empty_seq(S),    % Πράξη   ακολουθιών
    empty_seq(Q).    % Πράξη   ακολουθιών
ins_sort(S, Q) :-
    \+ empty_seq(S), % Πράξη ακολουθιών
    head(S, E),      % Πράξη   ακολουθιών
    tail(S, S1),     % Πράξη   ακολουθιών
    ins_sort(S1, Q1), % Αναδρομή
    ins_elem(E,Q1, Q).   % Νέο κατηγόρημα
```

```
ins_elem(E,Q1, Q) :-
    empty_seq(Q1), % Πράξη  ακολουθιών
    seq_cons(Q1,E, Q). % Πράξη   ακολουθιών
ins_elem(E,Q1, Q) :-
    head(Q1, V1), % Πράξη   ακολουθιών
    E @=<V1, % Πράξη σύγκρισης όρων
    seq_cons(Q1,E, Q). % Πράξη   ακολουθιών
ins_elem(E,Q1, Q) :-
    head(Q1, V),    % Πράξη   ακολουθιών
    E @> V,      % Πράξη σύγκρισης όρων
    tail(Q1, Q1a),    % Πράξη   ακολουθιών
    ins_elem(E,Q1a, Qa), % Αναδρομή
    seq_cons(Qa,V, Q).   % Πράξη   ακολουθιών
```

- **Πρόγραμμα 8.38: Υλοποίηση ταξινόμησης με παρεμβολή με ακολουθίες.**

8.5. Δομές Δεδομένων σε Prolog

8.5.2. Ακολουθίες: Παράδειγμα.

- Ακολουθούν μερικοί στόχοι που τρέξαμε στο Πρόγραμμα 8.38. Θα πρέπει να σημειωθεί ότι το πρόγραμμα ακολουθεί τη στάνταρντ σειρά ταξινόμηση όρων της Prolog.
  - | ?- ins_sort([2,1,1,4,2,5,5,0,0],S).

    S = [0,0,1,1,2,2,4,5,5] ? yes
  - | ?- ins_sort([2,1,1,4,5,5,0,0,-9,-8],S).

    S = [-9,-8,0,0,1,1,2,4,5,5] ? yes
  - | ?- ins_sort([a,2,b,b,1,3,1,a,2],S).

    S = [1,1,2,2,3,a,a,b,b] ? yes
  - | ?- ins_sort([a,2,b,a,b,1,3,2,a,2],S).

    S = [1,2,2,2,3,a,a,a,b,b] ? yes

Τέλος Διάλεξης

Ευχαριστώ!

Ερωτήσεις;