Ασκήσεις για το μάθημα *Λογική για Υπολογιστές*.

2ο σετ ασκήσεων

Ημερομηνία παράδοσης: Πέμπτη 11 Φεβρουαρίου 2010

Άσκηση 1

Δίνονται τα ακόλουθα κατηγορήματα και οι σημασίες τους:

- $nat(x)$: ισχύει αν και μόνο αν $x \in \mathbb{N}$
- $real(x)$: ισχύει αν και μόνο αν $x \in \mathbb{R}$
- $gt\_zero(x)$: ισχύει αν και μόνο αν $x > 0$
- $sq\_root(x, y)$: ισχύει αν και μόνο αν $x^2 = y$
- $equal(x, y)$: ισχύει αν και μόνο αν $x = y$

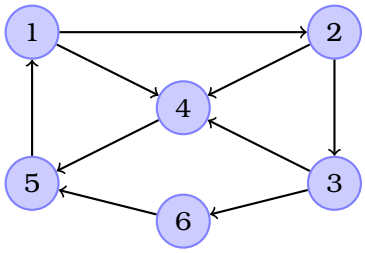
Χρησιμοποιώντας μόνο τα προηγούμενα κατηγορήματα, περιγράψτε σε κατηγορηματική λογική τις ακόλουθες προτάσεις σε φυσική γλώσσα:

1. *«Υπάρχει τουλάχιστον ένας παργματικός αριθμός x»*
2. *«Υπάρχουν τουλάχιστον δύο διαφορετικοί παργματικοί αριθμοί x και y»*
3. *«Υπάρχει τουλάχιστον ένας πραγματικός αριθμός μεγαλύτερος του 0»*
4. *«Για κάθε πραγματικό αριθμό x μεγαλύτερο του 0, υπάρχει ένας πραγματικός αριθμός y που είναι η τετραγωνική του ρίζα».*
5. *«Για κάθε πραγματικό αριθμό x μεγαλύτερο του 0, υπάρχει ένας πραγματικός αριθμός y που είναι η τετραγωνική του ρίζα και η οποία είναι μοναδική».*
6. *«Το 0 και το 1 είναι οι μοναδικοί πραγματικοί οι οποίοι είναι τετραγωνικές ρίζες του εαυτού τους».*
7. *«Κάθε φυσικός αριθμός έχει έναν (μοναδικό) επόμενό του».* Χρησιμοποιήστε και το συναρτησιακό σύμβολο $s(x)$ που επιστρέφει τον επόμενο του x, δηλαδή τον $x + 1$.

Άσκηση 2

Θεωρήστε ότι έχετε μια βάση γνώσης που περιγράφεται από τα ακόλουθα γεγονότα:

```
connected(1,2).
connected(1,4).
connected(2,3).
connected(2,4).
connected(3,4).
connected(3,6).
connected(4,5).
connected(5,1).
connected(6,5).
```



Σημειώνεται ότι, όπως φαίνεται και στο σχήμα, το `connected(X,Y)` σημαίνει ότι υπάρχει σύνδεση από τον κόμβο `X` προς τον κόμβο `Y` χωρίς να σημαίνει ότι υπάρχει σύνδεση από τον `Y` στον `X`.

1. Δώστε σε Prolog ένα κατηγόρημα `path(X,Y)` το οποίο να ισχύει όταν υπάρχει μονοπάτι από τον `X` στον `Y`.

2. Επεκτείνετε το κατηγόρημα `path(X,Y)` σε `path(X,Y,P)` ώστε να ισχύει όταν υπάρχει μονοπάτι από τον `X` στον `Y` και ταυτόχρονα να επιστρέφει το μονοπάτι στο `P`. (Σημείωση: χρησιμοποιήστε την τεχνική του προγραμματισμού με accumulators)

   ```
   ?- path(1,5,P).
   P = [1, 2, 3, 4, 5] ;
   P = [1, 2, 3, 4, 5, 1, 2, 3, 4|...] ;
   P = [1, 2, 3, 4, 5, 1, 2, 3, 4|...] ;
   ...
   ```

3. Επεκτείνετε το `path(X,Y,P)` έτσι ώστε να μην επισκέπτεται κάποιον κόμβο που έχει ήδη επισκευθεί. Ονομάστε το νέο κατηγόρημα που δώσατε `acyclic_path(X,Y,P)`.

   ```
   ?- acyclic_path(1,5,P).
   P = [1, 2, 3, 4, 5] ;
   P = [1, 2, 3, 6, 5] ;
   P = [1, 2, 4, 5] ;
   P = [1, 4, 5] ;
   false.
   ```

4. Με βάση το `acyclic_path(X,Y,P)` δώστε σε Prolog ένα κατηγόρημα `circle(P)` το οποίο να ισχύει αν το μονοπάτι `P` είναι κύκλος. Ένα μονοπάτι ονομάζεται κύκλος αν αρχίζει και τελειώνει στον ίδιο κομβο. Για παράδειγμα:

```
?- circle(X).
X = [1, 2, 3, 4, 5, 1] ;
X = [1, 2, 3, 6, 5, 1] ;
X = [1, 2, 4, 5, 1] ;
X = [1, 4, 5, 1] ;
X = [2, 3, 4, 5, 1, 2] ;
X = [2, 3, 6, 5, 1, 2] ;
X = [2, 4, 5, 1, 2] ;
X = [3, 4, 5, 1, 2, 3] ;
X = [3, 6, 5, 1, 2, 3] ;
X = [4, 5, 1, 4] ;
X = [4, 5, 1, 2, 4] ;
X = [4, 5, 1, 2, 3, 4] ;
X = [5, 1, 2, 3, 4, 5] ;
X = [5, 1, 2, 3, 6, 5] ;
X = [5, 1, 2, 4, 5] ;
X = [5, 1, 4, 5] ;
X = [6, 5, 1, 2, 3, 6] ;
false.
```

5. Γράψτε ένα κατηγόρημα `upath(X,Y,P)` το οποίο ισχύει όταν υπάρχει σύνδεση από τον X στον Y ***χωρίς*** να λαμβάνει υπ' όψην τις φορές των συνδέσεων. (Να θεωρεί, στο παράδειγμά μας, ότι υπάρχει σύνδεση και από τον 1 στον 2 αλλά και από τον 2 στον 1). Επιπρόσθετα, θα πρέπει να αποφεύγει να επισκευθεί τον ίδιο κόμβο περισσότερες από μία φορές ώστε να αποφεύγει τους κύκλους. Για παράδειγμα:

```
?- upath(1,3,P).
P = [1, 2, 3] ;
P = [1, 2, 4, 3] ;
P = [1, 2, 4, 5, 6, 3] ;
P = [1, 4, 3] ;
P = [1, 4, 5, 6, 3] ;
P = [1, 4, 2, 3] ;
P = [1, 5, 4, 3] ;
```

```
P = [1, 5, 4, 2, 3] ;
P = [1, 5, 6, 3] ;
false.
```

Άσκηση 3

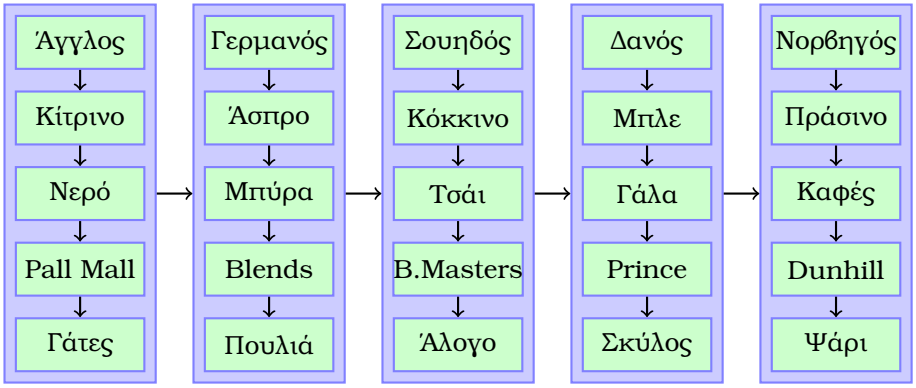
Θεωρήστε ότι υπάρχουν πέντε σπίτια πέντε διαφορετικών χρωμάτων. Σε κάθε σπίτι ζει ένας άνθρωπος διαφορετικής εθνικότητας. Οι πέντε ιδιοκτήτες πίνουν ένα συγκεκριμένο είδος ποτού. Καπνίζουν μία συγκεκριμένη μάρκα τσιγάρων και έχουν ένα συγκεκριμένο κατοικίδιο. Όλοι έχουν μεταξύ τους διαφορετικά κατοικίδια, διαφορετικές μάρκες τσιγάρων και διαφορετικά είδη ποτών. Τα στοιχεία που δίνονται είναι τα εξής:

1. Ο Αγγλος μένει στο κόκκινο σπίτι.
2. Ο Σουηδός έχει σκύλο.
3. Ο Δανός πίνει τσάι.
4. Το πράσινο σπίτι είναι αριστερά από το άσπρο σπίτι.
5. Ο ιδιοκτήτης του πράσινου σπιτιού πίνει καφέ.
6. Αυτός που καπνίζει Pall Mall εκτρέφει πουλιά.
7. Ο ιδιοκτήτης του κίτρινου σπιτιού καπνίζει Dunhill.
8. Αυτός που μένει στο μεσαίο σπίτι πίνει γάλα.
9. Ο Νορβηγός μένει στο πρώτο σπίτι.
10. Αυτός που καπνίζει Blends μένει δίπλα σ' αυτόν που έχει γάτες.
11. Αυτός που έχει το άλογο μένει δίπλα σ' αυτόν που καπνίζει Dunhill.
12. Ο ιδιοκτήτης που καπνίζει BlueMasters πίνει μπύρα.
13. Ο Γερμανός καπνίζει Prince.
14. Ο Νορβηγός μένει δίπλα στο μπλε σπίτι.
15. Αυτός που καπνίζει Blends έχει ένα γείτονα που πίνει νερό.

Γράψτε ένα πρόγραμμα σε Prolog το οποίο να βρίσκει τα στοιχεία κάθε ανθρώπου (πού μένει, τί πίνει, τί καπνίζει, τί κατοικίδιο έχει) ώστε να απαντήσετε στην ερώτηση *«ποιος έχει το ψάρι;»*

Υπόδειξη: Ένας πιθανός τρόπος αντιμετώπισης της άσκησης αυτής είναι να θεωρήσετε τη λύση σαν μια λίστα από λίστες. Για παράδειγμα, μια πιθανή λύση (που βέβαια δεν είναι σωστή) θα μπορούσε να αναπαριστάνεται με το εξής:

```
X = [ [english,yellow,water,pall_mall,cats],
      [german,white,beer,blends,birds],
      [sweedish,red,tea,blue_masters,horse],
      [danish,blue,milk,prince,dog],
      [norwegian,green,coffee,dunhill,fish] ]
```



Έτσι λοιπόν θα πρέπει να γράψετε ένα κατηγόρημα που να επιβάλλει όλους τους περιορισμούς του προβλήματος στη λίστα από λίστες.

Σημείωση 1

Αν η Prolog θεωρεί ότι το αποτέλεσμα είναι πολύ μεγάλο για να το τυπώσει όλο, μπορείτε μόλις μπείτε στο περιβάλλον της SWI-Prolog να δώσετε την εντολή:

```
?- set_prolog_flag(toplevel_print_options,[]).
true.
```

ώστε να μην κάνει περικοπές.

Σημείωση 2

Για τις ασκήσεις σε Prolog, ζητάμε τα κατηγορήματα να είναι γενικά, δηλαδή ναι μεν να δίνουν απάντηση για τη βάση γνώσης που δίνεται, αλλά να δίνουν απάντηση και αν αλλάξουμε τη βάση γνώσης με μια άλλη (πχ, αν προσθέσουμε ακμές ή κόμβους στην Άσκηση 1).