Ημερομηνία Ανάρτησης: 18/12/2015
Ημερομηνία Παράδοσης: 4/1/2016, 23:59μμ
Αρχές Γλωσσών Προγραμματισμού

1. (20%) Ορίστε σε Prolog τη σχέση `once(L,M)` η οποία δεδομένης μιας λίστας `L` που περιέχει φυσικούς αριθμούς, επιστρέφει τη λίστα `M` η οποία περιέχει τους αριθμούς που εμφανίζονται μία και μόνο φορά στη λίστα `L`. Για παράδειγμα, `once([1,7,2,1,5,2,8,5],M)` επιστρέφει `M=[7,8]`.

2. (20%) Ορίστε σε Prolog τη σχέση `pairs(L,X,M)` η οποία δεδομένης μιας λίστας $\texttt{L} = [l_1, \ldots, l_n]$ από θετικούς φυσικούς αριθμούς, επιστρέφει τον αριθμό `M` από ζεύγη (l_i, l_j), $1 \leq i < j \leq n$ για τα οποία ισχύει $l_i + l_j = \texttt{X}$. Για παράδειγμα, `pairs([5,12,7,10,9,1,2,3,11],13,M)` επιστρέφει `M=3` (το οποίο αντιστοιχεί στα ζεύγη $(12, 1)$, $(10, 3)$, $(2, 11)$).

3. (20%) Σε ένα νηπιαγωγείο, υπάρχουν `N` παιδιά τα οποία, μετά από μια εξαντλητική μέρα, ξεκουράζονται παρακολουθώντας τηλεόραση. Το πρόγραμμα της τηλεόρασης έχει `M` κανάλια τα οποία συμβολίζονται με τους φυσικούς αριθμούς 1 ως `M`. Κάθε παιδί έχει ένα κανάλι που αγαπάει και ένα κανάλι που μισεί. Αν η τηλεόραση είναι συντονισμένη σε ένα κανάλι που κάποιο παιδί το μισεί, το παιδί αυτό θα σηκωθεί τρέχοντας, θα αλλάξει το κανάλι σε αυτό που προτιμάει και μετά θα επιστρέψει στη θέση του. Αν υπάρχουν πολλά παιδιά που μισούν ένα κανάλι, το μικρότερο από αυτά θα σηκωθεί να το αλλάξει ενώ τα υπόλοιπα θα παραμείνουν καθισμένα. Φυσικά, είναι πιθανό μόλις ένα παιδί αλλάξει κανάλι, κάποιο άλλο παιδί να σηκωθεί και να αλλάξει το νέο κανάλι γιατί το βρίσκει ανυπόφορο. Όπως ξέρουμε, τα μικρά παιδιά είναι πεισματάρικα και κατά συνέπεια η διαδικασία αυτή μπορεί να συνεχιστεί επ'απειρο. Η νηπιαγωγός, η οποία βρίσκεται στα πρόθυρα νευρικής κατάρρευσης, θέλει να ξέρει πότε θα σταματήσει το πανδαιμόνιο. Ορίστε σχέση `allhappy(N,M,I,Ls,C)` η οποία παίρνει ως είσοδο τον αριθμό `N` των παιδιών, τον αριθμό `M` των καναλιών, το αρχικό κανάλι `I` και τη λίστα `Ls` που αποτελείται από ζεύγη `(A,B)` του αγαπημένου και του μισητού καναλιού για το κάθε παιδί, η οποία επιστρέφει τον αριθμό των αλλαγών καναλιών που θα λάβουν χώρα μέχρι όλα τα παιδιά να είναι ευχαριστημένα και καθισμένα στις θέσεις τους. Η διάταξη των παιδιών στη λίστα `Ls` είναι από το μικρότερο στο μεγαλύτερο. Σε περίπτωση που τα παιδιά δεν πρόκειται ποτέ να ηρεμήσουν, το πρόγραμμα σας θα πρέπει να επιστρέφει -1. Παραδείγματα: `allhappy(3, 4, 2, [(1,2),(2,3),(3,2)],C)` επιστρέφει $\texttt{C} = 1$, `allhappy(3, 3, 1, [(1,2),(2,3),(3,1)],C)` επιστρέφει $\texttt{C} = -1$ και `allhappy(4, 5, 2, [(1,3),(2,3),(3,2),(5,1)])` επιστρέφει $\texttt{C} = 3$.

4. (20%) Μια παρέα `N` φίλων πηγαίνει σε μια ταβέρνα και παραγγέλνει `M` διαφορετικά φαγητά. Ένα ακριβώς από τα φαγητά είναι αλλοιωμένο, αλλά κανείς δεν το γνωρίζει. Κάθε μέλος της παρέας που θα δοκιμάσει το προβληματικό φαγητό, θα αρχίσει κάποια στιγμή να νοιώθει άσχημα (είτε κατά τη διάρκεια του φαγητού, είτε αφότου το φαγητό έχει τελειώσει). Σας δίνεται μια περιγραφή της βραδιάς (τι έφαγε δηλαδή ο καθένας και σε ποια χρονική στιγμή). Βασισμένοι σε αυτή την πληροφορία, θα μπορείτε να βρείτε αν ένα φαγητό είναι πιθανό να είναι προβληματικό. Χρησιμοποιώντας αυτή τη γνώση, θα πρέπει να υπολογίσετε τον ελάχιστο αριθμό από δόσεις φαρμάκου που θα πρέπει να έχει το γειτονικό φαρμακείο ώστε να μπορέσει να εξυπηρετήσει όλα τα άτομα της παρέας που ενδέχεται να αρρωστήσουν είτε κατά τη διάρκεια του φαγητού είτε αργότερα.

   Ορίστε σε Prolog τη σχέση `doses(N,M,L,W,D)` η οποία δεδομένων των `N`, `M` και δύο λιστών `L` και `W` (δείτε παρακάτω), μας επιστρέφει τον αριθμό των δόσεων `D` από φάρμακο που ενδεχομένως θα χρειαστεί η παρέα. Η λίστα `L` αποτελείται από τριάδες της μορφής `(P,F,T)` που δείχνουν ότι το άτομο `P` έφαγε το φαγητό `F` τη χρονική στιγμή `T` (όπου $1 \leq \texttt{P} \leq \texttt{N}$, $1 \leq \texttt{F} \leq \texttt{M}$, $1 \leq \texttt{T} \leq 100$). Ένα άτομο μπορεί να φάει από το ίδιο φαγητό σε διαφορετικές χρονικές στιγμές και μπορεί επίσης να φάει πολλά φαγητά την ίδια στιγμή (η παρέα πεινάει...). Η λίστα `W` περιέχει ζεύγη της μορφής `(P,T)` που δείχνουν ότι το άτομο `P` αρρωσταίνει τη χρονική στιγμή $\texttt{T} \leq 100$. Κάθε άτομο αρρωσταίνει το πολύ μία φορά, και αρρωσταίνει επειδή έφαγε το προβληματικό φαγητό σε μια προηγούμενη χρονική στιγμή.

   Το πρόγραμμα σας επιστρέφει το φυσικό αριθμό `D` που είναι ο ελάχιστος αριθμός από δόσεις φαρμάκου που θα πρέπει να έχει το γειτονικό φαρμακείο ώστε να μπορέσει να εξυπηρετήσει όλα τα άτομα της παρέας που ενδέχεται να αρρωστήσουν είτε κατά τη διάρκεια του φαγητού είτε αργότερα. Για παράδειγμα, η κλήση:

   ```
   ?doses(3,4,[(1,1,1),(1,3,4),(3,1,3),(2,1,5),(2,2,7)],[(1,3),(2,8)],D).
   ```

   θα επιστρέψει `D = 3`.

Εξήγηση: Η παρέα αποτελείται από 3 ανθρώπους και υπάρχουν 4 φαγητά. Το άτομο 1 αρρωσταίνει τη χρονική στιγμή 3 και το άτομο 2 τη χρονική στιγμή 8. Το άτομο 3 δεν αρρωσταίνει κατά τη διάρκεια του φαγητού αλλά είναι πιθανό να αρρωστήσει αργότερα. Ας εξετάσουμε τα είδη των φαγητών ένα-ένα για να δούμε ποια από αυτά δεν μπορούμε να αποκλείσουμε ότι είναι προβληματικά. Ένα φαγητό είναι πιθανώς προβληματικό αν κάθε άτομο που αρρώστησε, έφαγε αυτό το φαγητό πριν αρρωστήσει.

Φαγητό 1: Και οι δύο άρρωστοι (1 και 2) το έφαγαν πριν αρρωστήσουν, και επομένως αυτό θα μπορούσε να είναι το προβληματικό. Αν αυτό ισχύει, τότε και το άτομο 3 επίσης το έφαγε και επομένως ενδέχεται να αρρωστήσει ένα ακόμη άτομο, ο 3, λίγο αργότερα.

Φαγητό 2: Και οι δύο που αρρώστησαν έφαγαν αυτό το φαγητό πριν αρρωστήσουν, και επομένως αυτό θα μπορούσε να είναι το προβληματικό. Κανένας άλλος δεν έφαγε αυτό το φαγητό, και επομένως το πολύ 2 άτομα θα μπορούσαν να αρρωστήσουν από το συγκεκριμένο έδεσμα.

Φαγητό 3: Αυτό προφανώς δεν μπορεί να είναι το προβληματικό φαγητό γιατί το άτομο 1 δεν το έφαγε πριν αρρωστήσει.

Φαγητό 4: Αυτό προφανώς δεν μπορεί να είναι το προβληματικό φαγητό γιατί το άτομο 2 αρρώστησε χωρίς να το φάει.

Κατά συνέπεια, το φαρμακείο πρέπει να έχει 3 δόσεις φαρμάκου, γιατί αν το φαγητό 1 είναι το προβληματικό, τότε 3 το πολύ άνθρωποι θα πρέπει να λάβουν φάρμακο.

5. (20%) Ένας από τους φίλους της προηγούμενης παρέας (ας τον πούμε Γιώργο), που δεν έφαγε το προβληματικό φαγητό, αποφασίζει να επιστρέψει στο σπίτι του οδηγώντας. Το σπίτι του βρίσκεται 100 χιλιόμετρα μακριά από την ταβέρνα, και λίγο πριν φτάσει στον προορισμό του, τον σταματάει ένας αστυνομικός και του δίνει κλήση για υπέρβαση του ορίου ταχύτητας. Όταν φτάνει στο σπίτι του, ο Γιώργος, παρά το ξενύχτι, αποφασίζει να μελετήσει τη διαδρομή που έκανε και να επιβεβαιώσει ότι η κλήση ήταν όντως δικαιολογημένη.

   Η διαδρομή που ακολούθησε ο Γιώργος αποτελείται από `N` τμήματα, που το καθένα περιγράφεται από ένα θετικό φυσικό αριθμό που δείχνει πόσα χιλιόμετρα είναι κάθε τμήμα (προφανώς, το άθροισμα των μηκών όλων των τμημάτων είναι 100). Κάθε ένα από τα τμήματα έχει ένα όριο ταχύτητας, που είναι ένας αριθμός από το 1 μέχρι το 100. Για παράδειγμα, η διαδρομή μπορεί να ξεκινάει με ένα τμήμα 45 χιλιομέτρων με όριο ταχύτητας 70, και μετά μπορεί να υπάρχει ένα τμήμα 55 χιλιομέτρων με όριο ταχύτητας 60.

   Το ταξίδι του Γιώργου μπορεί να περιγραφεί από μια σειρά από `M` τμήματα σε κάθε ένα από τα οποία το αυτοκίνητο είχε μια συγκεκριμένη ταχύτητα. Για παράδειγμα, μπορεί να ξεκινήσει διανύοντας 50 χιλιόμετρα με ταχύτητα 65 και μετά άλλα 50 χιλιόμετρα με ταχύτητα 55. Προφανώς, το άθροισμα των μηκών όλων των τμημάτων είναι και πάλι 100. Το αυτοκίνητο του Γιώργου έχει όριο ταχύτητας 100.

   Ορίστε σε Prolog τη σχέση `maxdiff(L,M,D)` η οποία δεδομένων δύο λιστών που περιέχουν η μεν `L` τη διαδρομή μαζί με τα όρια ταχύτητας και η δε `M` το ταξίδι του Γιώργου, επιστρέφει τη μέγιστη υπέρβαση του ορίου ταχύτητας που έγινε κατά τη διάρκεια του ταξιδιού. Αν το όριο δεν ξεπεράστηκε ποτέ, θα πρέπει να επιστρέφεται η τιμή 0. Για παράδειγμα, η κλήση:

   ```
   ?maxdiff([(40,75),(50,35),(10,45)],[(40,76),(20,30),(40,40)],D).
   ```

   θα επιστρέψει `D = 5`.

   Εξήγηση: Ο δρόμος αποτελείται από τρία τμήματα (40 χιλιόμετρα με όριο 75, 50 χιλιόμετρα με όριο 35 και 10 χιλιόμετρα με όριο 45). Ο Γιώργος οδηγεί τα πρώτα 40 χιλιόμετρα με ταχύτητα 76, τα επόμενα 20 με ταχύτητα 30 και τα τελευταία 40 με ταχύτητα 40. Η μέγιστη υπέρβαση γίνεται στο τελευταίο τμήμα της διαδομής του Γιώργου.

Παράδοση Ασκήσεων: Η παράδοση πρέπει να γίνει μέχρι τις 23:59μμ, την 4/1/2016. Θα δημιουργήσετε **ένα** αρχείο το οποίο θα περιέχει τις λύσεις όλων των ασκήσεων το οποίο θα στείλετε με *email* στο `antru@di.uoa.gr`

(με κοινοποίηση στο `prondo@di.uoa.gr`) μέχρι την παραπάνω ημερομηνία. **Δεν θα υπάρξει παράταση στην παράδοση των ασκήσεων**. Τα ονόματα των κατηγορημάτων που θα χρησιμοποιήσετε στα προγράμματά σας πρέπει να είναι **ακριβώς** τα ίδια με αυτά που καθορίζονται από την παραπάνω εκφώνηση. Καθυστερημένες ασκήσεις δεν θα βαθμολογηθούν.

Σημείωση: Για να μπορέσει κάποιος να λάβει μέρος στην τελική εξέταση του μαθήματος, θα πρέπει να έχει παραδώσει τις δύο πρώτες εργασίες (Prolog και Haskell) με προβιβάσιμο βαθμό.