Ημερομηνία Ανάρτησης: 15/12/2017
Ημερομηνία Παράδοσης: 7/1/2018, 23:59μμ
Αρχές Γλωσσών Προγραμματισμού

1. (10%) Ορίστε σε Haskell την άπειρη λίστα `allbin` που περιέχει όλες τις λίστες με τα δυαδικά ψηφία 0 και 1 (πχ. `allbin = [[0],[1],[0,0],[0,1],[1,0],[1,1],[0,0,0],[0,0,1],...]`). Υπόδειξη: για κάθε γραμμή του προγράμματος σας παραπάνω από τις δύο, θα αφαιρείται 20% από το ποσοστό της άσκησης.

2. (15%) 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 για πάντα. Η νηπιαγωγός, η οποία βρίσκεται στα πρόθυρα νευρικής κατάρρευσης, θέλει να ξέρει πότε θα σταματήσει το πανδαιμόνιο. Ορίστε συνάρτηση `allhappy n m i ls` 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 `(l,h)` του αγαπημένου και του μισητού καναλιού για το κάθε παιδί, και 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)] = 1`, `allhappy 3 3 1 [(1,2),(2,3),(3,1)] = -1` και `allhappy 4 5 2 [(1,3),(2,3),(3,2),(5,1)] = 3`.

3. (20%) Ο μικρός Μήτσος έχει ενθουσιαστεί με το νέο του ηλεκτρονικό παιγνίδι skyscrapers, στο οποίο υπάρχουν N ουρανοξύστες διατεταγμένοι από αριστερά προς τα δεξιά και ένας μικρός ήρωας, ο Jumpy. Ο i-οστός ουρανοξύστης έχει ύψος H_i και έχει στην ταράτσα του G_i χρυσά νομίσματα. Το παιγνίδι ξεκινάει με ένα αρχικό άλμα του Jumpy σε έναν οποιονδήποτε ουρανοξύστη και εξελίσσεται με μια σειρά από επόμενα βήματα. Σε κάθε βήμα, ο Jumpy μπορεί να μεταβεί σε έναν ουρανοξύστη που βρίσκεται στα δεξιά του (μπορεί να υπερπηδήσει κάποιους ουρανοξύστες) αρκεί να μην είναι πιο χαμηλός από αυτόν στον οποίο βρίσκεται. Από κάθε ουρανοξύστη στον οποίο προσγειώνεται, ο Jumpy μαζεύει όλα τα χρυσά νομίσματα. Ο Jumpy μπορεί να ολοκληρώσει το παιγνίδι μετά από οσαδήποτε βήματα (ακόμη και 0), αλλά πρέπει να έχει μαζέψει τουλάχιστον K χρυσά νομίσματα για να μπορέσει να προχωρήσει στο επόμενο επίπεδο του παιγνιδιού. Ο Μήτσος θέλει να ξέρει ποιος είναι ο αριθμός των τρόπων που μπορεί να παίξει το παιγνίδι ώστε να προχωρήσει στο επόμενο επίπεδο. Ορίστε σε Haskell τη συνάρτηση `skyscrapers k ls` η οποία δεδομένου ενός αριθμού $k \geq 1$ και μιας λίστας `ls` από ζεύγη (H_i, G_i), επιστρέφει τον αριθμό των διαφορετικών τρόπων που μπορεί να παιχτεί το παιγνίδι. Για παράδειγμα, `skyscrapers 6 [(2,1),(6,3),(7,2),(5,6)]`, θα επιστρέψει `3` (οι διαφορετικοί τρόποι είναι $\{1,2,3\}$, $\{1,4\}$ και $\{4\}$).

4. (25%) Ένας μη κατευθυνόμενος γράφος μπορεί να αναπαρασταθεί στη Haskell με μια λίστα από ζεύγη φυσικών αριθμών, όπου ο κάθε φυσικός αντιστοιχεί σε μια κορυφή του γράφου και το κάθε ζεύγος σε μια πλευρά του γράφου. Να γραφεί συνάρτηση `isomorphic` σε Haskell η οποία δεδομένων δύο γράφων, μας επιστρέφει `True` αν οι δύο γράφοι είναι ισομορφικοί και `False` διαφορετικά. Για παράδειγμα, `isomorphic [(1,3),(3,5),(5,1),(5,7)] [(4,5),(2,3),(3,4),(4,2)]` επιστρέφει `True`, ενώ `isomorphic [(1,2),(1,3),(1,4),(1,5)] [(3,4),(4,5),(5,6),(6,7)]` επιστρέφει `False`.

5. (30%) Έστω μια λίστα η οποία περιέχει όλους τους αριθμούς από το 1 μέχρι το n, χωρίς επαναλήψεις και όχι υποχρεωτικά στη σειρά. Υποθέτουμε ότι τη λίστα αυτή αρχικά δεν τη γνωρίζουμε. Γνωρίζουμε όμως πέντε λίστες που έχουν προκύψει από την άγνωστη λίστα με τον ακόλουθο τρόπο. Η πρώτη λίστα έχει προκύψει από την άγνωστη λίστα με μετακίνηση ενός από τους αριθμούς σε μια άλλη θέση. Η δεύτερη λίστα έχει προκύψει από τη μετακίνηση ενός *διαφορετικού* αριθμού από την *αρχική* λίστα σε μια άλλη θέση, κλπ. Γράψτε συνάρτηση `findlist xss` σε Haskell η οποία δεδομένης μιας λίστας `xss`

που έχει για στοιχεία πέντε λίστες, επιστρέφει ως αποτέλεσμα την αρχική λίστα. Για παράδειγμα, το ερώτημα `findlist [[1,2,5,3,4],[1,5,3,4,2],[4,2,1,5,3],[2,3,1,5,4],[2,1,3,4,5]]` θα πρέπει να επιστρέφει `[2,1,5,3,4]`.

Παράδοση Ασκήσεων: Η παράδοση πρέπει να γίνει μέχρι τις 23:59μμ, την 7/1/2018. Θα δημιουργήσετε **ένα** αρχείο το οποίο θα περιέχει τις λύσεις όλων των ασκήσεων και θα το στείλετε με *email* και στις δύο παρακάτω διευθύνσεις: `antru@di.uoa.gr` και `prondo@di.uoa.gr`. Ερωτήσεις σχετικά με τις ασκήσεις θα πρέπει να απευθύνονται στο πρώτο mail (Αντώνης Τρουμπούκης). **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.