ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ Η/Υ ΚΑΙ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΜΑΘΗΜΑ: ΑΡΧΕΣ ΓΛΩΣΣΩΝ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ
ΑΚΑΔ. ΕΤΟΣ: 2019-20

1η Σειρά Εργαστηριακών Ασκήσεων

Οι εργαστηριακές ασκήσεις είναι **ατομικές**. Οι απαντήσεις θα πρέπει να υποβληθούν με **turnin**, το αργότερο μέχρι την **Πέμπτη 7 Μαΐου 2020**. Πριν ξεκινήσετε να γράφεται τα προγράμματα που ζητούνται στις ασκήσεις της σειράς αυτής, **διαβάστε πολύ προσεκτικά τις αναλυτικές οδηγίες** που ακολουθούν.

Οδηγίες

- Για μεγαλύτερη ευκολία στην εκπόνηση αυτής και της επόμενης εργαστηριακής άσκησης σας συνιστώ:

  1. να δημιουργήσετε έναν κατάλογο Haskell κάτω από το home directory σας.
  2. να χρησιμοποιήτε αυτόν τον κατάλογο για αποθήκευση των αρχείων με τα προγράμματα Haskell που θα γράψετε.
  3. να μεταβαίνετε σε αυτόν τον κατάλογο πριν εκτελέσετε τον διερμηνέα hugs της Haskell.

- Αν θέλετε να χρησιμοποιήσετε τη Haskell στο δικό σας υπολογιστή, μπορείτε να κατεβάσετε το διερμηνέα hugs από το σύνδεσμο

  https://www.haskell.org/hugs/pages/downloading-May2006.htm

  Συνοπτικές οδηγίες για τη χρήση του hugs υπάρχουν στις σημειώσεις.

- Πριν ξεκινήσετε να γράφετε τα προγράμματα που ζητούνται στις παρακάτω ασκήσεις, θα ήταν χρήσιμο να γράψετε σε ένα αρχείο ορισμένες από τις συναρτήσεις των σημειώσεων, να φορτώσετε το αρχείο στον hugs και να αποτιμήσετε παραστάσεις που χρησιμοποιούν τις συναρτήσεις αυτές, έτσι ώστε να εξοικειωθείτε με την γλώσσα Haskell και το διερμηνέα της.

- Για τη συγγραφή των προγραμμάτων επιτρέπεται να χρησιμοποιήσετε προκαθορισμένες συναρτήσεις και προκαθορισμένους τελεστές **μόνο εφόσον αναφέρονται στις σημειώσεις του μαθήματος**. Δεν επιτρέπεται η χρήση του import.

- Για τη συγγραφή των συναρτήσεων θα πρέπει να χρησιμοποιήσετε το αρχείο πρότυπο Lab1.hs (που υπάρχει στην ιστοσελίδα του μαθήματος), στο οποίο υπάρχουν έτοιμες οι δηλώσεις τύπων των συναρτήσεων που θα πρέπει να κατασκευάσετε καθώς και μία ισότητα που ορίζει τις συναρτήσεις ώστε να επιστρέφουν μία προκαθοριμένη τιμή για όλες τις τιμές των ορισμάτων. Για να απαντήσετε σε μία άσκηση μπορείτε να αντικαταστήσετε την παραπάνω ισότητα με τις κατάλληλες ισότητες που ορίζουν την τιμή της συνάρτησης. **Δεν θα πρέπει να τροποποιήσετε το τύπο ούτε το όνομα της συνάρτησης**.

- Μπορείτε να χρησιμοποιήσετε όσες βοηθητικές συναρτήσεις θέλετε, οι οποίες θα καλούνται από τις συναρτήσεις που σας ζητείται να υλοποιήσετε. Σε καμία περίπτωση δεν θα πρέπει να προσθέσετε άλλα ορίσματα στις συναρτήσεις που σας ζητούνται (καθώς αυτό συνεπάγεται αλλαγή του τύπου τους).

- **Αν χρησιμοποιήσετε προκαθορισμένες συναρτήσεις ή τελεστές που δεν αναφέρονται στις σημειώσεις του μαθήματος ή αν χρησιμοποιήσετε το import για να ενσωματώσετε έτοιμο κώδικα, η αντίστοιχη άσκηση δεν θα βαθμολογηθεί.**

- Ο έλεγχος της ορθότητας των απαντήσεων θα γίνει με ημι-αυτόματο τρόπο. Σε καμία περίπτωση δεν θα πρέπει ο βαθμολογητής να χρειάζεται να κάνει παρεμβάσεις στο αρχείο που θα υποβάλετε. Συνεπώς θα πρέπει να λάβετε υπόψη τα παρακάτω:

  1. Κάθε μία από τις συναρτήσεις που σας ζητείται να υλοποιήσετε θα πρέπει να έχει το συγκεκριμένο όνομα και το συγκεκριμένο τύπο που περιγράφεται στην εκφώνηση της αντίστοιχης άσκησης και που υπάρχει στο αρχείο πρότυπο Lab1.hs. **Αν σε κάποια άσκηση το όνομα ή ο τύπος της συνάρτησης δεν συμφωνεί με αυτόν που δίνεται στην εκφώνηση, η άσκηση δεν θα βαθμολογηθεί**.
  2. Το αρχείο που θα παραδώσετε δεν θα πρέπει να περιέχει συντακτικά λάθη. Αν υπάρχουν τμήματα κώδικα που περιέχουν συντακτικά λάθη, τότε θα πρέπει να τα διορθώσετε ή να τα αφαιρέσετε πριν από την παράδοση. **Αν το αρχείο που θα υποβάλετε περιέχει συντακτικά λάθη, τότε ολόκληρη η εργαστηριακή άσκηση θα μηδενιστεί**.
  3. Οι συναρτήσεις θα πρέπει να επιστρέφουν αποτέλεσμα για όλες τις τιμές των ορισμάτων που δίνονται για έλεγχο στο τέλος κάθε άσκησης. Αν κάποιες από τις τιμές που επιστρέφουν οι συναρτήσεις δεν είναι σωστές, αυτό θα ληφθεί υπόψη στη βαθμολογία, ωστόσο η άσκηση θα βαθμολογηθεί κανονικά. **Αν ωστόσο οι συναρτήσεις δεν επιστρέφουν τιμές για κάποιες από τις τιμές ελέγχου (π.χ. προκαλούν υπερχείλιση στοίβας, ατέρμονο υπολογισμό ή κάποιο σφάλμα χρόνου εκτέλεσης) τότε η αντίστοιχη άσκηση δεν θα βαθμολογηθεί**.
  4. Κατα τη διόρθωση των ασκήσεων οι βαθμολογητές δεν θα κάνουν κλήσεις στις βοηθητικές συναρτήσεις που ενδεχομένως θα χρησιμοποιήσετε. Η χρήση των βοηθητικών συναρτήσεων θα πρέπει να γίνεται μέσα από τις συναρτήσεις που σας ζητείται να υλοποιήσετε.

- Μετά το τέλος της εκφώνησης κάθε άσκησης δίνονται τιμές που μπορείτε να χρησιμοποιήσετε για έλεγχο της ορθότητας των συναρτήσεων.

- Για όποιο πρόβλημα συναντήσετε κατά τη συγγραφή των προγραμμάτων στο πλάισιο των εργαστηριακών ασκήσεων μπορείτε να στέλνετε email στην κ. Βίκυ Σταμάτη (vstamati@cs.uoi.gr) με θέμα 'ΜΥΥ401- ασκήσεις '.

- Για υποβολή με turnin γράψτε (υποθέτοντας ότι έχετε γράψει το πρόγραμμα στο αρχείο Lab1.hs):

turnin Haskell-1@myy401 Lab1.hs

Ασκηση 1.

Γράψτε μία συνάρτηση `count` σε Haskell, η οποία θα υπολογίζει το πλήθος των ψηφίων ενός ακεραίου αριθμού, τα οποία είναι $0, 3, 6$ ή 9. Ο τύπος της συνάρτησης `count` θα πρέπει να είναι `Integer -> Int`

Για έλεγχο χρησιμοποιήστε τις παρακάτω τιμές:

```
Main> count 0
1
Main> count 3
1
Main> count 7
0
Main> count 10
1
Main> count (-33)
2
Main> count (-77)
0
Main> count 3690
4
Main> count 175824
0
Main> count (-1234567890)
4
Main> count 457309582
3
Main> count (3^150)
27
Main> count (3^1500)
285
```

Ασκηση 2.

Γράψτε μία συνάρτηση `kgcd` σε Haskell, η οποία θα δέχεται ως ορίσματα τρεις θετικούς ακέραιους αριθμούς m, n, k και θα βρίσκει τον k-οστο μεγαλύτερο κοινό διαιρέτη των m και n. Αν οι αριθμοί m και n έχουν λιγότερους από k κοινούς διαιρέτες, τότε η συνάρτηση θα πρέπει να επιστρέφει την τιμή 0. Ο τύπος της συνάρτησης `kgcd` θα πρέπει να είναι `Int->Int->Int`.

Για έλεγχο χρησιμοποιήστε τις παρακάτω τιμές:

```
Main> kgcd 35 24 1
1
Main> kgcd 36 24 1
12
Main> kgcd 35 24 2
0
Main> kgcd 36 24 3
4
Main> kgcd 1001 887 1
1
Main> kgcd 648 432 1
216
Main> kgcd 648 432 5
36
Main> kgcd 648 432 10
9
Main> kgcd 648 432 12
6
Main> kgcd 648 432 15
2
Main> kgcd 648 432 17
0
Main> kgcd 1365 910 1
455
Main> kgcd 1365 910 5
13
Main> kgcd 1365 910 7
5
Main> kgcd 1365 910 9
0
```

Ασκηση 3.

Σκοπός αυτής της άσκησης είναι η κατασκευή μίας συνάρτησης η οποία θα επιστρέφει το πλήθος των δευτερολέπτων που έχουν περάσει από την αρχή του έτους μεχρι μία συγκεκριμένη χρονική στιγμή μέσα στο ίδιο έτος. Για απλούστευση υποθέτουμε ότι το έτος δεν είναι δίσεκτο.

Μια ημερομηνία περιγράφεται ως ένα ζεύγος ακεραίων αριθμών, όπου το δεύτερο στοιχείο του είναι ένας αριθμός ανάμεσα στο 1 και το 12 που αναπαριστά τον μήνα και το πρώτο στοιχείο του είναι ένας αριθμός ανάμεσα στο 1 και το μέγιστο πλήθος ημερών του μήνα που υποδεικνύει το δεύτερο στοιχείο του.

Η ώρα περιγράφεται ως μία τριάδα ακεραίων αριθμών, όπου το πρώτο στοιχείο της είναι ένας αριθμός ανάμεσα στο 0 και το 23 που αναπαριστά τις ώρες, ενώ το δεύτερο και το τρίτο στοιχείο της είναι αριθμοί ανάμεσα στο 0 και το 59 που αναπαριστούν αντίστοιχα τα λεπτά και τα δευτερόλεπτα.

Γράψτε μία συνάρτηση `seconds` σε Haskell, η οποία με είσοδο ένα ζεύγος ακεραίων που αναπαριστά μία ημερομηνία και μία τριάδα ακεραίων που αναπαριστά την ώρα, θα επιστρέφει το πλήθος των δευτερολέπτων που έχουν περάσει από την αρχή του έτους μέχρι τη χρονική στιγμή που προσδιορίζουν η δεδομένη ημερομηνία και ώρα. Η συνάρτηση θα πρέπει να επιστρέφει την τιμή -1 αν το πρώτο όρισμά της δεν καθορίζει μία έγκυρη ημερομηνία ή αν το δεύτερο όρισμά της δεν καθορίζει μία έγκυρη ώρα. Ο τύπος της συνάρτησης `seconds` θα πρέπει να είναι `(Int,Int)->(Int,Int,Int)->Int`.

Για έλεγχο χρησιμοποιήστε τις παρακάτω τιμές:

```
Main> seconds (1,1) (0,0,8)
8
Main> seconds (1,1) (0,2,0)
120
Main> seconds (1,1) (1,0,0)
3600
Main> seconds (2,1) (0,0,0)
86400
Main> seconds (1,2) (0,0,0)
2678400
Main> seconds (3,3) (9,9,25)
5303365
Main> seconds (15,4) (23,47,52)
9071272
Main> seconds (27,6) (5,1,36)
15310896
Main> seconds (12,8) (12,12,12)
19311132
Main> seconds (30,9) (7,45,0)
23528700
Main> seconds (14,11) (14,18,32)
27440312
Main> seconds (31,12) (23,59,59)
31535999
```

```
Main> seconds (30,0) (8,45,30)
-1
Main> seconds (30,13) (8,45,30)
-1
Main> seconds (31,4) (8,45,30)
-1
Main> seconds (1,7) (24,15,15)
-1
Main> seconds (10,10) (8,-8,11)
-1
```

Ασκηση 4.

Γράψτε μία συνάρτηση υψηλότερης τάξης `sumfab` σε Haskell, η οποία με ορίσματα μία συνάρτηση f τύπου `Int->Int->Int->Int` και δύο ακέραιους a και b θα επιστρέφει το άθροισμα

$$\sum_{k=a}^{b} f(a,k,b)$$

Ο τύπος της συνάρτησης `sumfab` θα πρέπει να είναι `(Int->Int->Int->Int)->Int->Int->Int`.

Δεν επιτρέπεται να χρησιμοποιήσετε λίστες.

Για έλεγχο χρησιμοποιήστε τις παρακάτω τιμές:

```
Main> sumfab (\x y z -> x^2) (-4) 4
144
Main> sumfab (\x y z -> 2^z) 1 8
2048
Main> sumfab (\x y z -> 2^y) 0 9
1023
Main> sumfab (\x y z -> (z-x)^y) 1 4
120
Main> sumfab (\x y z -> (y-x+1)*(z-y+1)) 1 120
295240
Main> sumfab (\x y z -> (div z y)^x) 3 500
9519400
```