ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ Η/Υ ΚΑΙ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΜΑΘΗΜΑ: ΑΡΧΕΣ ΓΛΩΣΣΩΝ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ
ΑΚΑΔ. ΕΤΟΣ: 2016-17
ΔΙΔΑΣΚΩΝ: Χ.ΝΟΜΙΚΟΣ

2η Σειρά Εργαστηριακών Ασκήσεων

Οι εργαστηριακές ασκήσεις είναι ατομικές. Οι απαντήσεις θα πρέπει να υποβληθούν με **turnin**, το αργότερο μέχρι την **Παρασκευή 7 Απριλίου 2017**.

Οι ώρες οι οποίες έχουν δεσμευτεί για το εργαστήριο του μαθήματος είναι την Παρασκευή 5-7μμ. Η παρουσία στο εργαστήριο τις παραπάνω ώρες δεν είναι υποχρεωτική. Μπορείτε να έρχεστε στο εργαστήριο τις ώρες αυτές για όποια βοήθεια χρειάζεστε σχετικά με την εκπόνηση των εργαστηριακών ασκήσεων και γενικότερα τον προγραμματισμό στη γλώσσα Haskell, καθώς και για την επίλυση προβλημάτων που παρουσιάζονται κατά τη συγγραφή των προγραμμάτων στο πλαίσιο των εργαστηριακών ασκήσεων. **Σημειώνεται ότι σε καμία περίπτωση δεν θα παρέχεται σχετική βοήθεια μέσω e-mail.**

- Για τη συγγραφή των συναρτήσεων συνίσταται να χρησιμοποιήσετε το αρχείο πρότυπο Lab2.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).

- 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 Lab2.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 αφαιρέσεται 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 υποβολή με turnin γράψτε (υποθέτοντας ότι έχετε γράψει το πρόγραμμα στο αρχείο Lab2.hs):

turnin Haskell-2@myy401 Lab2.hs

Ασκηση 1.
Γράψτε μία συνάρτηση update σε Haskell η οποία θα δέχεται ως ορίσματα έναν ακέραιο n και μία λίστα ακεραίων και θα επιστρέφει τη λίστα που προκύπτει αν διαγραφούν όλες οι εμφανίσεις του n από τη δεδομένη λίστα (εφόσον υπάρχουν) και εισαχθεί το n στο τέλος της. Ο τύπος της συνάρτησης update θα πρέπει να είναι Int− >[Int]− >[Int].

Για έλεγχο χρησιμοποιήστε τις παρακάτω τιμές:

```
Main> update 1 []
[1]
Main> update 2 [2]
[2]
Main> update 3 [3,4,5,6,7,8]
[4,5,6,7,8,3]
Main> update 4 [2,3,4,5,6,7]
[2,3,5,6,7,4]
Main> update 5 [1,2,3,4,5]
[1,2,3,4,5]
Main> update 6 [1,2,5,8]
[1,2,5,8,6]
Main> update 7 [7,7,7,7,3,4,5,8]
[3,4,5,8,7]
Main> update 8 [1,2,3,4,5,8,8,8,8]
[1,2,3,4,5,8]
Main> update 9 [9,9,9,9,9,9,9,9]
[9]
Main> update 10 [10,2,10,3,5,10,7,10,8,10]
[2,3,5,7,8,10]
```

Ασκηση 2.

Γράψτε μία συνάρτηση fromTo σε Haskell η οποία θα δέχεται ως ορίσματα δύο ακέραιους i και j και μία λίστα με στοιχεία οποιουδήποτε τύπου και θα επιστρέφει τη υπολίστα η οποία περιέχει τα στοιχεία της δεδομένης λίστας που βρίσκονται σε θέσεις που είναι μεγαλύτερες ή ίσες του i και μικρότερες ή ίσες του j. Θεωρήστε ότι η κεφαλή μίας λίστας βρίσκεται στη θέση 1. Η συνάρτηση θα πρέπει να επιστρέφει αποτέλεσμα ακόμη και όταν η δεδομενη λίστα είναι άπειρη. Ο τύπος της συνάρτησης fromTo θα πρέπει να είναι Int− >Int− >[u]− >[u].

Για έλεγχο χρησιμοποιήστε τις παρακάτω τιμές:

```
Main> fromTo 3 7 [1..10]
[3,4,5,6,7]
Main> fromTo 1 7 [1..10]
[1,2,3,4,5,6,7]
Main> fromTo 4 10 [1..10]
[4,5,6,7,8,9,10]
Main> fromTo 1600 1600 [0..]
[1599]
Main> fromTo 1 2 []
[]
Main> fromTo 20 30 ['a'..'z']
"tuvwxyz"
Main> fromTo 12 20 "Haskell"
""
Main> fromTo (-4) 6 [0, 25..]
[0,25,50,75,100,125]
Main> fromTo (-4) (-3) [False,True]
[]
Main> fromTo (-18) (18) [0,45..200]
[0,45,90,135,180]
```

Ασκηση 3.
Γράψτε μία συνάρτηση hosum υψηλότερης τάξης σε Haskell η οποία θα δέχεται ως όρισμα μία συνάρτηση f από ακέραιους σε ακέραιους και θα επιστρέφει ως αποτέλεσμα τη συνάρτηση (επίσης από ακέραιους σε ακέραιους) της οποίας η τιμή για το n είναι:

$$\sum_{i=-|n|}^{|n|} f(i)$$

Ο τύπος της συνάρτησης hosum θα πρέπει να είναι (Int− >Int)− >(Int− >Int).

Για έλεγχο χρησιμοποιήστε τις παρακάτω τιμές:

```
Main> hosum (\x->1) 0
1
Main> hosum (\x->1) 5
11
Main> hosum (\x->1) (-8)
17
Main> hosum (\x->x) 10
0
Main> hosum (\x->abs x) 10
110
Main> hosum (\x->x*x) 12
1300
Main> hosum (\x->2^(abs x)) 9
2045
Main> hosum (\x->x `mod` 3) 1000
2001
Main> hosum (hosum (\x->1)) 7
127
Main> hosum (hosum (\x->x^2)) 4
200
```

Ασκηση 4.
Γράψτε μία συνάρτηση apply υψηλότερης τάξης σε Haskell η οποία θα δέχεται ως ορίσματα (α) μία λίστα συναρτήσεων με πεδίο ορισμού οποιονδήποτε τύπο και πεδίο τιμών οποιονδήποτε διατεταγμένο τύπο και (β) μία λίστα με στοιχεία του ίδιου τύπου με το πεδίο ορισμού της συνάρτησης και θα επιστρέφει ως αποτέλεσμα μία λίστα η οποία θα περιέχει όλες τις τιμές που προκύπτουν από από την εφαρμογή κάποιας συνάρτησης της πρώτης λίστας σε κάποιο στοιχείο της δεύτερης λίστας. Η επιστρεφόμενη λίστα θα πρέπει να είναι ταξινομημένη σε αύξουσα τάξη, χωρίς επαναλήψεις στοιχείων. Ο τύπος της συνάρτησης apply θα πρέπει να είναι Ord u => [v− >u]− >[v]− >[u].

Για έλεγχο χρησιμοποιήστε τις παρακάτω τιμές:

```
Main> apply [abs] [-1]
[1]
Main> apply [(^2)] [1..5]
[1,4,9,16,25]
Main> apply [(^2),(^3),(^4),(2^)] [10]
[100,1000,1024,10000]
Main> apply [(^0),(0^),(\x->div x x),(\x->mod x x)] [1..1000]
[0,1]
Main> apply [(^2),(^3),(^4),(2^)] [2..8]
[4,8,9,16,25,27,32,36,49,64,81,125,128,216,256,343,512,625,1296,2401,4096]
Main> apply [head,last] ["abc","aaaa","cbbc","cbbca"]
"ac"
Main> apply [head.tail,last.init] ["abc","aaaa","cbbc","cbbca"]
"abc"
Main> apply [reverse,(++"ing"),reverse.(++"ing"),(++"ing").reverse] ["play","do"]
["doing","gniod","gniyalp","od","oding","playing","yalp","yalping"]
Main> apply [\x->mod x 10, \x->rem x 10] [-100..100]
[-9,-8,-7,-6,-5,-4,-3,-2,-1,0,1,2,3,4,5,6,7,8,9]
Main> apply [(*5)] (apply [(`div`5)] [1..100])
[0,5,10,15,20,25,30,35,40,45,50,55,60,65,70,75,80,85,90,95,100]
```