ΣΕΤ ΑΣΚΗΣΕΩΝ 4

ΕΡΓΑΣΤΗΡΙΟ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ Ι, ΑΚΑΔΗΜΑΪΚΟ ΕΤΟΣ 2016-2017

Προθεσμία: 22/12/2016, 21:00

Διαδικαστικά

Η εργασία αυτή μπορεί να γίνει σε ομάδες μέχρι 2 ατόμων. Δεν είναι απαραίτητο να συνεργαστείτε με το ίδιο άτομο που κάνατε τα εβδομαδιαία εργαστήρια ή τα τρία πρώτα σετ ασκήσεων. Μπορείτε να συζητάτε τις ασκήσεις με συμφοιτητές σας αλλά δεν επιτρέπεται να δίνετε ή να παίρνετε κώδικα με οποιοδήποτε τρόπο.

Ξεκινήστε νωρίς! Ο προγραμματισμός είναι πάντα ΠΟΛΥ πιο χρονοβόρος από ό,τι περιμένετε.

Μη διστάζετε να ζητήσετε βοήθεια! Μπορείτε να χρησιμοποιήσετε το forum προγραμματισμού

Προσθέστε σε σχόλια στην αρχή του κάθε αρχείου με κώδικα τα πλήρη ονόματα και ΑΜ των μελών της ομάδας. Παρακαλούμε να γράφετε τα σχόλια ΜΟΝΟ με λατινικούς χαρακτήρες.

Για να παραδώσετε τη δουλειά σας:

Κατασκευάστε ένα φάκελο με όνομα `hw4_epwnumo1_AM1_epwnumo2_AM2` και αντιγράψτε μέσα σε αυτόν τα `hw4a.c, hw4b.c .` Μην ξεχάσετε να προσθέσετε τα ονόματά σας στο αρχείο, σε σχόλια.

Πηγαίνετε στο φάκελο μέσα στον οποίο βρίσκεται το `hw4_epwnumo1_AM1_epwnumo2_AM2` που κατασκευάσατε και πακετάρετε και συμπιέστε τον με την παρακάτω εντολή:

```
tar czvf epwnumo1_AM1_epwnumo2_AM2.tgz epwnumo1_AM1_epwnumo2_AM2
```

Στείλτε email:

- στη διεύθυνση **ce120lab@gmail.com**
- αντίγραφο (**CC**) στο άλλο μέλος της ομάδας σας, αν υπάρχει
- θέμα (subject) **CE120 hw4**
- και επικολλημένο αρχείο το `hw4_epwnumo1_AM1_epwnumo2_AM2.tgz`

Εκπρόθεσμες ασκήσεις δε γίνονται δεκτές.

Άσκηση 1: Συμμετοχή σε εργαστήρια

Εισαγωγή

Σε αυτό το σετ ασκήσεων θα υλοποιήσετε μια δομή στην οποία αποθηκεύονται τα στοιχεία των φοιτητών που συμμετέχουν στα εργαστήρια ενός μαθήματος.

Οι φοιτητές είναι χωρισμένοι σε πολλαπλά τμήματα ανάλογα με το πλήθος των φοιτητών και τον αριθμό υπολογιστών σε κάθε εργαστήριο. Ένα τμήμα αναπαρίσταται ως μια μη-κυκλική, διπλά διασυνδεδεμένη λίστα χωρίς τερματικό, κάθε κόμβος της οποίας αντιστοιχεί σε ένα φοιτητή.

Η δομή που αναπαριστά το σύνολο των φοιτητών είναι ένας (δυναμικά δεσμευμένος) πίνακας από struct όπου κάθε κελί αντιστοιχεί σε ένα εργαστηριακό τμήμα. Για την ακρίβεια, κάθε κελί του πίνακα περιέχει τη διεύθυνση της κεφαλής μιας λίστας όπως αυτή περιγράφεται πιο πάνω, καθώς και την ώρα που ξεκινά το εργαστήριο.

Το πρόγραμμά σας θα παρέχει διάφορες λειτουργίες: Εισαγωγή ενός φοιτητή σε εργαστήριο, αμοιβαία ανταλλαγή φοιτητών που ανήκουν σε διαφορετικά τμήματα, αναζήτηση φοιτητή, οριστική αφαίρεση ενός φοιτητή από το τμήμα στο οποίο ανήκει και φυσικά εκτύπωση των στοιχείων των φοιτητών ανά τμήμα.

Επιπλέον, θα είναι δυνατό για το χρήστη να τρέξει το πρόγραμμα με δύο τρόπους: είτε κανονικά είτε με επιπρόσθετα διαγνωστικά μηνύματα. Στην παρακάτω περιγραφή, τα διαγνωστικά μηνύματα θα γράφονται με κόκκινο χρώμα και τα "κανονικά" με μπλε. Τα κανονικά μηνύματα εκτυπώνονται σε κάθε περίπτωση.

Μορφή κώδικα και άλλες απαιτήσεις

Το πρόγραμμά σας πρέπει να λειτουργεί σωστά και να εκτυπώνει όλα τα μηνύματα και αποτελέσματα με τον τρόπο που σας περιγράφουμε στις επόμενες ενότητες. Φροντίστε να ελέγχετε αν οι συναρτήσεις που σχετίζονται με δέσμευση μνήμης επιστρέφουν άκυρες τιμές (NULL) γιατί σε αυτές τις περιπτώσεις το πρόγραμμα πρέπει να τερματίζει, αφού αποδεσμεύσει όλη τη δυναμικά δεσμευμένη μνήμη.

Παρατηρήστε ότι αρκετές από τις συναρτήσεις που θα γράψετε χρησιμοποιούνται σε περισσότερα από ένα σημεία του προγράμματος. Είναι επιθυμητό να υπάρχει μια συνάρτηση για κάθε λειτουργία του προγράμματος που γίνεται συχνά ώστε να καλείτε αυτή τη συνάρτηση όποτε χρειάζεται. Αν "πιάσετε" τον εαυτό σας να κάνει copy+paste κώδικα μέσα στο πρόγραμμά σας, τότε κατά πάσα πιθανότητα αυτό που γράφετε θα μπορούσε να μπει σε μια συνάρτηση.

Πέρα από τις συναρτήσεις που σας ζητάμε στην εκφώνηση, είναι αναμενόμενο ότι θα γράψετε και δικές σας για να βελτιώσετε την ποιότητα του κώδικά σας. Για παράδειγμα, θα μπορούσατε να γράψετε μια συνάρτηση η οποία παίρνει ως παραμέτρους τα argc, argv και κάνει τους αρχικούς ελέγχους, ή μια συνάρτηση που αναλαμβάνει την υλοποίηση της βασικής επανάληψης στη main.

Χρησιμοποιήστε συναρτήσεις από τη βιβλιοθήκη `string` σε όσα σημεία κάνετε διαχείριση συμβολοσειρών.

Θα σας δώσουμε ενδεικτικά αρχεία εισόδου και εξόδου. Χρησιμοποιήστε ανακατεύθυνση εισόδου/εξόδου και την εντολή diff όπως κάνατε και στα προηγούμενα σετ ασκήσεων για να επιβεβαιώσετε ότι το πρόγραμμά σας βγάζει ακριβώς την αναμενόμενη έξοδο.

Όπως πάντα, φροντίστε να έχετε καλογραμμένο κώδικα που ακολουθεί τις αρχές που περιγράφονται σε αυτά τα φυλλάδια.

Οι αστερίσκοι σε κάθε ενότητα υποδηλώνουν σχετικό βαθμό δυσκολίας.

Στάδιο 0: Κατασκευή δομών και χειρισμός παραμέτρων προγράμματος (∗)

Ορίστε ένα struct το οποίο περιγράφει ένα φοιτητή (κόμβο λίστας), κι έχει τα πεδία:

- όνομα (δείκτης σε χαρακτήρα)
- τμήμα (ακέραιος αριθμός, αύξων αριθμός τμήματος ξεκινώντας από το 1)
- δείκτη προς τον επόμενο κόμβο
- δείκτη προς τον προηγούμενο κόμβο

Ορίστε ένα τύπο enum που αναπαριστά εργάσιμη ημέρα. Μεταβλητές αυτού του τύπου θα παίρνουν τιμές από Δευτέρα έως και Παρασκευή.

Ορίστε ένα struct το οποίο περιγράφει ένα τμήμα εργαστηρίου, κι έχει τα πεδία:

- διεύθυνση κεφαλής λίστας φοιτητών
- ώρα που ξεκινά το εργαστήριο (ακέραιος αριθμός).
- ημέρα που γίνεται το εργαστήριο (enum που κατασκευάσατε στο προηγούμενο βήμα)

Ορίστε ένα τύπο enum που θα παίρνει τις τιμές ON, OFF.

Δηλώστε μια καθολική[1] μεταβλητή τύπου enum (που ορίσατε στο προηγούμενο βήμα) με όνομα debug.

- Η debug θα παίρνει την τιμή ON ή OFF ανάλογα με τις παραμέτρους του προγράμματος. Όταν είναι ON θέλουμε το πρόγραμμα να εκτυπώνει και διαγνωστικά μηνύματα (πέρα των κανονικών) ενώ όταν είναι OFF το πρόγραμμα εκτυπώνει μόνο τα "κανονικά" μηνύματα.

Γράψτε μια συνάρτηση `main` στην οποία ελέγχετε τις παραμέτρους προγράμματος που δίνονται από τη γραμμή εντολών και αρχικοποιείτε αναλόγως τη μεταβλητή `debug`.

- Αν δεν έχει δοθεί κάποια παράμετρος, τότε το πρόγραμμα δε θα εκτυπώνει διαγνωστικά μηνύματα.
- Αν έχει δοθεί η παράμετρος `-d` ή `-debug` τότε το πρόγραμμα θα εκτυπώνει διαγνωστικά μηνύματα.
- Αν έχει δοθεί οποιαδήποτε άλλη παράμετρος, τότε το πρόγραμμα εκτυπώνει το μήνυμα:

  ```
  X: illegal option -- Y
  Usage: X [-d]
  ```

  όπου X το όνομα του εκτελέσιμου και Y η παράμετρος που δόθηκε, χωρίς την αρχική παύλα (αν αυτή υπάρχει). Ακολούθως, το πρόγραμμα εκτυπώνει χαρακτήρα αλλαγής γραμμής και τερματίζει.
- Αν δοθεί λάθος πλήθος παραμέτρων, το πρόγραμμα εκτυπώνει το μήνυμα:

  ```
  X: wrong number of arguments Y
  ```

  όπου X το όνομα του εκτελέσιμου και Y το πλήθος των παραμέτρων. Ακολούθως, το πρόγραμμα εκτυπώνει χαρακτήρα αλλαγής γραμμής και τερματίζει.

1 Αυτή είναι η μοναδική καθολική μεταβλητή που επιτρέπεται να έχει το πρόγραμμα.

Στάδιο 1: Εισαγωγή δεδομένων(∗∗∗)

Γράψτε μια συνάρτηση εισαγωγής δεδομένων και αρχικοποίησης δομών η οποία

- Εκτυπώνει το μήνυμα `Enter number of students:` ακολουθούμενο από ένα κενό (space) και διαβάζει το πλήθος των φοιτητών
- Εκτυπώνει το μήνυμα `Enter maximum lab size:` ακολουθούμενο από ένα κενό (space) και διαβάζει το πλήθος των υπολογιστών στο εργαστήριο.
- Υπολογίζει πόσα τμήματα χρειάζεται να υπάρχουν (πλήθος φοιτητών ανά πλήθος υπολογιστών) και δεσμεύει δυναμικά μνήμη για τον πίνακα τμημάτων.
- Αρχικά, όλα τα τμήματα είναι άδεια (κενές λίστες). Η ώρα και η μέρα κάθε τμήματος αποφασίζονται ως εξής: Εργαστήρια γίνονται από Δευτέρα έως και Παρασκευή, μεταξύ 8:00 και 22:00 και έχουν διάρκεια από μία ώρα. Το πρώτο τμήμα ξεκινά Δευτέρα στις 8:00, και κάθε επόμενο τμήμα γίνεται την ίδια μέρα, μια ώρα μετά. Το τελευταίο τμήμα της κάθε ημέρας ξεκινά στις 21:00 και το επόμενο τμήμα, αν υπάρχει, ξεκινά στις 8:00 την επόμενη μέρα.
- Εκτυπώνει το διαγνωστικό μήνυμα `>>Created X sections<<` όπου X το πλήθος των τμημάτων, με ένα χαρακτήρα αλλαγής γραμμής πριν και μετά.
- Για κάθε ένα φοιτητή, εκτυπώνει το μήνυμα `Enter student name:` ακολουθούμενο από ένα κενό (space), και μετά διαβάζει το όνομά του. Τα ονόματα έχουν μήκος το πολύ 30, συμπεριλαμβανομένου του '\0', αλλά όταν τα αποθηκεύετε σε κόμβο πρέπει να δεσμεύετε *ακριβώς* όση μνήμη χρειάζεται.
- Κατασκευάζει ένα νέο κόμβο λίστας για αυτόν τον φοιτητή, βρίσκει σε ποιο τμήμα θα ανήκει αρχικά ο φοιτητής και τον προσθέτει σε αυτό *χρησιμοποιώντας τη συνάρτηση που περιγράφεται παρακάτω*. Σε αυτό το στάδιο οι φοιτητές εισάγονται κυκλικά στα τμήματα. Για παράδειγμα, αν υπάρχουν 3 τμήματα, ο πρώτος φοιτητής μπαίνει στο πρώτο, ο δεύτερος στο δεύτερο, ο τρίτος στο τρίτο, ο τέταρτος στο πρώτο, ο πέμπτος στο δεύτερο κ.ο.κ.
- Η συνάρτηση στο τέλος επιστρέφει τη διεύθυνση που είναι αποθηκευμένος ο πίνακας. Επιπλέον, *μέσω της λίστας παραμέτρων* επιστρέφει το πλήθος των τμημάτων.
- Αν έχει παρουσιαστεί οποιοδήποτε πρόβλημα στη δέσμευση μνήμης, η συνάρτηση πρέπει να εκτυπώνει κατάλληλο μήνυμα, να απελευθερώνει *όλη* τη δυναμικά δεσμευμένη μνήμη και να επιστρέφει NULL.

Γράψτε μια συνάρτηση η οποία παίρνει ως παραμέτρους τη διεύθυνση του πίνακα τμημάτων, τη διεύθυνση ενός κόμβου λίστας που αναπαριστά ένα φοιτητή και το τμήμα στο οποίο πρέπει να μπει ο φοιτητής, και

- Εκτυπώνει το διαγνωστικό μήνυμα `>>Added student "X" in section Y<<` , όπου X το όνομα του φοιτητή και Y το τμήμα, με ένα χαρακτήρα αλλαγής γραμμής πριν και μετά.
- Προσθέτει το φοιτητή στο κατάλληλο τμήμα, στην αρχή της λίστας.

Καλέστε τη συνάρτηση αρχικοποίησης από τη main για να διαβάσετε τα στοιχεία των φοιτητών και να τα αποθηκεύσετε στον δυναμικό πίνακα λιστών. Εννοείται πως αν η δημιουργία του πίνακα αποτύχει, το πρόγραμμα πρέπει να τερματίζει.

Ολοκληρώστε αυτό το στάδιο και βεβαιωθείτε ότι λειτουργεί σωστά πριν προχωρήσετε στο επόμενο.

Στάδιο 2: Εκτύπωση τμημάτων (∗)

Γράψτε μια συνάρτηση η οποία παίρνει ως παραμέτρους τη διεύθυνση που είναι αποθηκευμένος ο πίνακας τμημάτων και το πλήθος τμημάτων, και :

- Εκτυπώνει δύο χαρακτήρες αλλαγής γραμμής, το μήνυμα `==========LABS==========` (10 '=' σε κάθε πλευρά) και χαρακτήρα αλλαγής γραμμής.
- Για κάθε τμήμα εκτυπώνει χαρακτήρα αλλαγής γραμμής, το μήνυμα `*Section X, Y Z:00*` όπου X ο αύξων αριθμός του τμήματος, Y η ημέρα που γίνεται το τμήμα και Z η ώρα που ξεκινά, και χαρακτήρα αλλαγής γραμμής. Οι ημέρες πρέπει να εμφανίζονται ως `Monday`, `Tuesday`, `Wednesday`, `Thursday` ή `Friday`.
- Για κάθε φοιτητή μέσα σε ένα τμήμα εκτυπώνει ένα χαρακτήρα tab ακολουθούμενο από το όνομα του φοιτητή.
- Αφού εκτυπωθούν όλα τα τμήματα, εκτυπώνει χαρακτήρα αλλαγής γραμμής, το μήνυμα `========================` (24 '=' για να είναι ίσου μήκους με το προηγούμενο) και χαρακτήρα αλλαγής γραμμής

Καλέστε τη συνάρτηση από τη `main`, αμέσως μετά την αρχικοποίηση.

Ολοκληρώστε αυτό το στάδιο και βεβαιωθείτε ότι λειτουργεί σωστά πριν προχωρήσετε στο επόμενο.

Στάδιο 3: Υλοποίηση αμοιβαίας ανταλλαγής (∗∗∗)

Γράψτε μια συνάρτηση η οποία παίρνει ως παραμέτρους το όνομα ενός φοιτητή και ότι άλλο χρειάζεται και:

- Αναζητά το φοιτητή στον πίνακα τμημάτων, κι αν τον βρει, επιστρέφει τη διεύθυνση του κόμβου που αντιστοιχεί σε αυτόν τον φοιτητή. Αν δεν τον βρει, επιστρέφει NULL.
- Όταν αρχίζει την αναζήτηση σε κάποιο τμήμα η συνάρτηση πρέπει να εκτυπώνει το διαγνωστικό μήνυμα `>>Searching section X for Y...<<` με ένα χαρακτήρα αλλαγής γραμμής πριν και μετά, όπου X ο αύξων αριθμός του τμήματος και Y το όνομα του φοιτητή που ψάχνουμε. Μετά, για ένα-ένα φοιτητή που ελέγχει, εκτυπώνει `Z...` όπου Z το όνομά του. Εκτυπώνεται και το όνομα του φοιτητή που ψάχνουμε, αν αυτό βρεθεί.

Γράψτε μια συνάρτηση η οποία παίρνει ως παραμέτρους τη διεύθυνση κόμβου λίστας που περιέχει ένα φοιτητή και ότι άλλο χρειάζεται και αφαιρεί τον κόμβο από τη λίστα (χωρίς να αποδεσμεύσει τη δυναμικά δεσμευμένη μνήμη γι αυτόν). Μετά την αφαίρεση, εκτυπώνει το διαγνωστικό μήνυμα `>>Removed student "X" from section Y<<` όπου X το όνομα του φοιτητή και Y το τμήμα του, με χαρακτήρα αλλαγής γραμμής πριν και μετά.

Γράψτε μια συνάρτηση η οποία παίρνει ως παραμέτρους τα ονόματα δύο φοιτητών και ότι άλλο χρειάζεται και:

- Χρησιμοποιεί τη συνάρτηση αναζήτησης που γράψατε νωρίτερα για να βρει αν οι φοιτητές υπάρχουν.
- Αν κάποιος από τους φοιτητές δεν υπάρχει σε κανένα τμήμα, η συνάρτηση πρέπει να εκτυπώνει το μήνυμα `Student "X" not in lab.` όπου X είναι το όνομα του φοιτητή, με ένα χαρακτήρα αλλαγής γραμμής πριν και μετά. Αυτό εκτυπώνεται για όποιον φοιτητή δεν υπάρχει. Στο τέλος εκτυπώνει το μήνυμα `Swap canceled.` ακολουθούμενο από χαρακτήρα αλλαγής γραμμής και τερματίζει. Ελέγξτε τους φοιτητές με τη σειρά που δόθηκαν αρχικά τα ονόματα, για να βγάλετε ίδια αποτελέσματα με τα αρχεία ελέγχου.
- Αν και οι δύο φοιτητές είναι στο ίδιο τμήμα, η συνάρτηση πρέπει να εκτυπώνει το μήνυμα

`Students "X" and "Y" are in the same section. Swap canceled.` όπου X και Y τα ονόματα των δύο φοιτητών, ένα χαρακτήρα αλλαγής γραμμής και να τερματίζει. Μπορείτε να υποθέσετε ότι δεν υπάρχουν συνωνυμίες και δε θα ζητηθεί ποτέ να αλλάξει κάποιος με τον εαυτό του.

- Διαφορετικά, χρησιμοποιεί τη συνάρτηση διαγραφής φοιτητών για να αφαιρέσει τους φοιτητές από τα τμήματά τους και τη συνάρτηση προσθήκης φοιτητών για να τους προσθέσει στα νέα τους τμήματα.

Στη `main` προσθέστε κώδικα ο οποίος:

- Εκτυπώνει χαρακτήρα αλλαγής γραμμής, το μήνυμα `"Enter action: "` ακολουθούμενο από ένα κενό (space) και διαβάζει μια εντολή από το χρήστη. Η εντολή είναι μια λέξη μεγέθους το πολύ 20 (συμπεριλαμβανομένου του χαρακτήρα '\0')
- Αν η εντολή είναι SWAP τότε εκτυπώνει το μήνυμα `Enter first name:` ακολουθούμενο από ένα κενό (space), διαβάζει το όνομα του πρώτου φοιτητή, εκτυπώνει το μήνυμα `Enter second name:` ακολουθούμενο από ένα κενό (space), διαβάζει το όνομα του δεύτερου φοιτητή και μετά καλεί τη συνάρτηση ανταλλαγής φοιτητών που γράψατε σε αυτό το στάδιο για να τους αλλάξει τμήματα.
- Αν η εντολή είναι PRINT, καλεί τη συνάρτηση εκτύπωσης τμημάτων.
- Αν η εντολή είναι EXIT, τερματίζει αφού αποδεσμεύσει όλη τη δυναμικά δεσμευμένη μνήμη κι εκτυπώσει ένα χαρακτήρα αλλαγής γραμμής.
- Για οποιαδήποτε άλλη εντολή εκτυπώνει το μήνυμα `Invalid action. Valid actions are SWAP, REMOVE, PRINT, EXIT.` ακολουθούμενο από χαρακτήρα αλλαγής γραμμής.
- Σε κάθε περίπτωση εκτός της EXIT, τα παραπάνω βήματα επαναλαμβάνονται.

Ολοκληρώστε αυτό το στάδιο και βεβαιωθείτε ότι λειτουργεί σωστά πριν προχωρήσετε στο επόμενο.

Στάδιο 4: Οριστική αφαίρεση φοιτητή από εργαστήριο (*)

Γράψτε μια συνάρτηση η οποία παίρνει ως παραμέτρους το όνομα ενός φοιτητή και ότι άλλο χρειάζεται και τον αφαιρεί οριστικά από το εργαστήριο στο οποίο αυτός βρίσκεται, ελευθερώνοντας όλη τη μνήμη που έχει δεσμευθεί γι αυτόν δυναμικά. Αν ο φοιτητής δε βρίσκεται σε κανένα τμήμα, τότε εκτυπώνει το μήνυμα `Student "X" not in lab. Remove canceled.` που X το όνομα του φοιτητή και Y το τμήμα του, με χαρακτήρα αλλαγής γραμμής πριν και μετά.

- Χρησιμοποιήστε συναρτήσεις που γράψατε στο βήμα 3.

Στη `main` προσθέστε κώδικα στην επανάληψη που γράψατε στο βήμα 3 ώστε:

- Αν ο χρήστης έδωσε την εντολή REMOVE, το πρόγραμμα να εκτυπώνει το μήνυμα `Enter name:` ακολουθούμενο από ένα κενό (space), να διαβάζει το όνομα του φοιτητή και να καλεί την παραπάνω συνάρτηση για να τον αφαιρέσει.

Σημείωση: Υπάρχει περίπτωση ένα τμήμα να μείνει χωρίς φοιτητές μετά από πολλαπλές αφαιρέσεις. Δε χρειάζεται να κάνετε κάτι ιδιαίτερο. Απλά στην επιλογή PRINT δε θα εμφανίζονται ονόματα φοιτητών για το συγκεκριμένο τμήμα.

Αρχείο προς παράδοση: `hw4a.c`