Πολυτεχνείο Κρήτης

Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

ΠΛΗ 102 - Δομημένος Προγραμματισμός

Εαρινό Εξάμηνο 2018-2019 | 1η Άσκηση | Παράδοση: 14/3/2019

Οδηγίες

- Οι ασκήσεις βοηθούν στην εξοικείωσή σας με τον προγραμματισμό και την κατανόηση της ύλης.
- Η ενασχόληση με τις ασκήσεις είναι η καλύτερη προετοιμασία για την εργαστηριακή εξέταση.
- Η συζήτηση των ασκήσεων είναι θεμιτή, αλλά η σύνταξη του κώδικα πρέπει να γίνεται ατομικά.
- Γράψτε κομψό, ευανάγνωστο κώδικα και προσθέστε επεξηγηματικά σχόλια οπού χρειάζεται.
- Τα σχόλια να γράφουν με λατινικούς χαρακτήρες για να μην υπάρχει πρόβλημα με το encoding.
- Τηρήστε κατά γράμμα τις οδηγίες και τις προδιάγραφες που σας δίνονται από την άσκηση.
- Η παράδοση των λύσεων γίνεται μονό μεσώ της ιστοσελίδας http://courses.ece.tuc.gr.
- Το παραδοτέο θα πρέπει να είναι ένα συμπιεσμένο αρχείο (.zip) με τα αρχεία του κώδικά σας.
- Η υποβολή παραδοτέου αυτόματα δηλώνει ότι είστε οι μοναδικοί συγγράφεις των λύσεων.
- Το παραδοτέο σας θα είναι διαθέσιμο για χρήση στη διάρκεια της εργαστηριακής εξέτασης.
- Σε περιπτώσεις ταυτοσήμων παραδοτέων (αντιγραφής) οι εμπλεκόμενοι μηδενίζονται.

Θέμα Α – Δείκτες και Λίστες

Γράψτε ένα πρόγραμμα στη γλώσσα προγραμματισμού C, το οποίο χειρίζεται (απλές) συνδεδεμένες λίστες ακεραίων αριθμών. Κάθε κόμβος της λίστας είναι τύπου **struct list** και περιέχει δύο στοιχεία: α) έναν ακέραιο **value** με τα δεδομένα του κόμβου και β) έναν δείκτη **next** στον επόμενο κόμβο. Η λίστα θα πρέπει να είναι προσβάσιμη μόνο από τον δείκτη **head** που δείχνει στην κεφαλή της. Το πρόγραμμά σας θα πρέπει να εκτελεί τα παρακάτω:

1. Θα ζητάει από τον χρήστη το πλήθος ***N*** των ακεραίων αριθμών που θα εισαχθούν στη λίστα.
2. Θα ζητάει έναν-έναν τους αριθμούς και θα τους εισάγει στη λίστα με τη σειρά που δίνονται.
3. Θεωρήστε ότι ο χρήστης δίνει έγκυρους αριθμούς και ότι όλοι είναι διαφορετικοί μεταξύ τους.
4. Θα τυπώνει τα περιεχόμενα της λίστας (τους αριθμούς) με τη σειρά που είναι αποθηκευμένα.
5. Θα ζητάει από τον χρήστη έναν ακέραιο αριθμό **n** και θα τον αναζητά μέσα στη λίστα.
    a. Αν ο **n** δεν βρεθεί, δεν θα κάνει καμία αλλαγή στη λίστα.
    b. Αν ο **n** βρεθεί και ο αριθμός στην αμέσως επόμενη θέση είναι μεγαλύτερός του, θα μεταφέρει ολόκληρο τον κόμβο που περιέχει τον **n** στο τέλος της λίστας. Αν στην τρέχουσα θέση του n ο επόμενος αριθμός δεν είναι μεγαλύτερός του, θα αφήνει τον **n** στη θέση του.
    c. Αν ο **n** βρεθεί και είναι ήδη στο τέλος της λίστας, θα τον αφήνει στη θέση του.
6. Σε κάθε μία περίπτωση θα πρέπει να ενημερώνει τον χρήστη με σχετικό μήνυμα για το τι έγινε.
7. Θα τυπώνει τα περιεχόμενα της λίστας που προέκυψε με την σειρά που είναι αποθηκευμένα.

Η βασική λειτουργία θα πρέπει να υλοποιείται με επαναληπτικές δομές (χωρίς αναδρομή) από την συνάρτηση **`findAndPushBack`** η οποία θα πρέπει να δηλωθεί επακριβώς με έναν από τους παρακάτω δύο τρόπους (ο πρώτος για χρήση δεικτών σε κόμβους και ο δεύτερος για χρήση δεικτών σε δείκτες σε κόμβους):

- `struct list * findAndPushBack(int n, struct list * node){....}`
- `void findAndPushBack(int n, struct list ** node){.....}`

Για τη δημιουργία/εκτύπωση της λίστας μπορείτε να χρησιμοποιήσετε κώδικα του εργαστηρίου. Προσοχή στη σωστή ενημέρωση του head (εντός ή εκτός της συνάρτησης), όπου χρειάζεται.

Παράδειγμα: Αν ο χρήστης δώσει 7 αριθμούς (4, 8, 21, 14, 6, 3, 19), η λίστα θα έχει ως εξής:

4 8 21 14 6 3 19

Αν στη συνέχεια ο χρήστης δώσει τον αριθμό 6, η νέα λίστα μετά τη μετακίνηση θα πρέπει να τυπωθεί ως εξής:

4 21 14 6 3 19 8

Αν ο χρήστης είχε δώσει 21, 14, 6, ή 19 δεν θα υπήρχε καμία μετακίνηση, καθώς ο αριθμός στον επόμενο κόμβο δεν είναι μεγαλύτερος ή βρισκόμαστε ήδη στο τέλος της λίστας (για τον 19). Αν είχε δώσει 4 ή 3, θα είχαμε μετακίνηση στο τέλος της λίστας. Προσοχή στη σωστή ενημέρωση του head (εντός ή εκτός της συνάρτησης), όπου χρειάζεται.

Παραδοτέο-Λύση: ένα αρχείο **`LAB102xxxxx_set1_ex1.c`** με το πρόγραμμά σας

Θέμα Β - Αναδρομή

Γράψτε ένα πρόγραμμα στη γλώσσα προγραμματισμού C το οποίο χειρίζεται (απλές) συνδεδεμένες λίστες ακεραίων αριθμών. Κάθε κόμβος της λίστας είναι τύπου struct list και περιέχει δύο στοιχεία: έναν ακέραιο value για την τιμή του κόμβου και έναν δείκτη next στον επόμενο κόμβο. Η λίστα θα πρέπει να είναι προσβάσιμη μόνο από τον δείκτη head που δείχνει στην κεφαλή της. Το πρόγραμμά σας θα πρέπει να εκτελεί τα παρακάτω:

1. Θα ζητάει από τον χρήστη το πλήθος ***N*** των ακεραίων αριθμών που θα εισαχθούν στη λίστα.
2. Θα ζητάει έναν-έναν τους αριθμούς και θα τους **εισάγει** στη λίστα **με τη σειρά** που δίνονται.
3. Θα τυπώνει τα περιεχόμενα της λίστας (τους αριθμούς) με την σειρά που είναι αποθηκευμένα.
4. Θα ελέγχει εάν ο αριθμός σε κάθε θέση **διαιρεί ακριβώς** το άθροισμα των επόμενων.
5. Ως «επόμενοι» νοούνται οι αριθμοί που ακολουθούν στη λίστα μετά από αυτή τη θέση.
6. Το άθροισμα των αριθμών σε μια κενή υπο-λίστα (αν δεν υπάρχουν επόμενοι) είναι 0.
8. Το «διαιρεί ακριβώς» σημαίνει να δίνει υπόλοιπο μηδέν (0) μετά από ακέραια διαίρεση.
9. Για κάθε θέση, θα τυπώνει τον αριθμό, το άθροισμα και αν ικανοποιείται η συνθήκη ή όχι.
10. Η **εκτύπωση** των αποτελεσμάτων πρέπει να γίνεται με την **αντίστροφη σειρά** αποθήκευσης.

Η βασική λειτουργία ελέγχου και εκτύπωσης των αποτελεσμάτων θα πρέπει να υλοποιείται από την **αναδρομική** συνάρτηση **`checkModuloOfSum`** η οποία θα πρέπει να δηλωθεί ως εξής:

```
int checkModuloOfSum(struct list * node){.......}
```

Δεν επιτρέπεται η χρήση βρόχων (`while, for`), καθολικών ή στατικών μεταβλητών στη συνάρτηση, ούτε η προσθήκη και κλήση άλλων (αναδρομικών ή επαναληπτικών) συναρτήσεων. Η διάσχιση της λίστας θα πρέπει να γίνεται αποκλειστικά μέσω αναδρομής (μία διάσχιση, μπρος-πίσω) και δεν επιτρέπονται πολλαπλές διασχίσεις της λίστας.
Παράδειγμα: Αν ο χρήστης δώσει 7 αριθμούς (5, 2, 3, 6, 1, 7, 4), η λίστα θα έχει ως εξής:

```
5    2    3    6    1    7    4
```

Ο έλεγχος του προγράμματος στην παραπάνω λίστα θα πρέπει να τυπώσει:

```
4[0](Yes)  7[4](No)  1[11](Yes)  6[12](Yes)  3[18](Yes)  2[21](No)  5[23](No)
```

Στην έξοδο, τυπώνεται πρώτα ο αριθμός, μετά το άθροισμα των επομένων σε τετράγωνες αγκύλες, και τέλος το αποτέλεσμα του ελέγχου για τέλεια ακέραια διαίρεση σε παρενθέσεις.

Παραδοτέο-Λύση: ένα αρχείο **`LAB102xxxxx_set1_ex2.c`** με το πρόγραμμά σας