Εργαστήριο 5

Εαρινό Εξάμηνο 2010-2011

Στόχοι του εργαστηρίου

- Χρήση στοίβας
- Αναδρομή

Σκοπός της 5ης εργαστηριακής άσκησης είναι να εξοικειωθείτε με την χρήση της στοίβας. Αυτό θα σας επιτρέψει να υλοποιείτε καλύτερα συναρτήσεις οι οποίες θα μπορούν να καλέσουν με την σειρά τους άλλες συναρτήσεις ή ακόμα και τον εαυτό τους οπότε και λέγονται αναδρομικές. Ανατρέξτε στα παραδείγματα που κάνατε στην τάξη πάνω στην αναδρομή.

Για το επόμενο εργαστήριο έχετε να υλοποιήσετε τις 2 παρακάτω εργαστηριακές ασκήσεις. Την ώρα του εργαστηρίου θα εξετασθείτε προφορικά πάνω στους κώδικες που θα παραδώσετε.

ΠΡΟΣΟΧΗ: Λύσεις που δεν χρησιμοποιούν αναδρομή ΔΕΝ θα γίνουν δεκτές!

Άσκηση 1 (3,5 μονάδες)

Να γραφεί μία αναδρομική συνάρτηση int func(int *array, int size, int num) η οποία θα ψάχνει στον πίνακα ακεραίων array (μεγέθους size) το πλήθος των στοιχείων μικρότερα ή ίσα από τον αριθμό num. Θα επιστρέφει αυτήν την τιμή στην main, η οποία θα την εκτυπώσει στην οθόνη.
Θα διαβάζετε το μέγεθος του πίνακα και τα αντίστοιχα στοιχεία από το πληκτρολόγιο.

Παράδειγμα:
Με array = [**1**,**4**,**3**,**2**,5,8,**3**,9,7,6,5,8,**0**] και num = 4, τότε η συνάρτηση επιστρέφει 6.

Άσκηση 2 (6,5 μονάδες)

Η ρωμαϊκή αρίθμηση χρησιμοποιεί τα παρακάτω ψηφία:

- M = 1000
- D = 500
- C = 100
- L = 50
- X = 10
- V = 5
- I = 1

Σε γενικές γραμμές, τα ρωμαϊκά ψηφία είναι γραμμένα σε φθίνουσα σειρά από αριστερά προς τα δεξιά, και προστίθενται διαδοχικά, για παράδειγμα MMVI (2006) ερμηνεύεται ως 1000 + 1000 + 5 + 1.

Ορισμένοι συνδυασμοί όμως τηρούν την αφαιρετική αρχή, η οποία ορίζει ότι όταν ένα σύμβολο μικρότερης αξίας προηγείται ένα σύμβολο μεγαλύτερης αξίας, η μικρότερη τιμή αφαιρείται από τη μεγαλύτερη τιμή, και το αποτέλεσμα προστίθεται στο σύνολο. Για παράδειγμα, MCMXLIV (1944), τα σύμβολα C, X και I προηγούνται ένα σύμβολο μεγαλύτερης αξίας, και το αποτέλεσμα ερμηνεύεται ως εξής:
1000 + (1000 - 100) + (50 - 10) + (5 - 1).

1 = I	11 = XI	10 = X	100 = C	1 000 = M
2 = II	12 = XII	20 = XX	200 = CC	2 000 = MM
3 = III	13 = XIII	30 = XXX	300 = CCC	3 000 = MMM
4 = IV	14 = XIV	40 = XL	400 = CD	4 000 = MMMM
5 = V	15 = XV	50 = L	500 = D	
6 = VI	16 = XVI	60 = LX	600 = DC	
7 = VII	17 = XVII	70 = LXX	700 = DCC	
8 = VIII	18 = XVIII	80 = LXXX	800 = DCCC	
9 = IX	19 = XIX	90 = XC	900 = CM	

Να γραφτεί πρόγραμμα το οποίο θα διαβάζει από το πληκτρολόγιο μία συμβολοσειρά και θα καλέσει ανάλογα τις δύο παρακάτω αναδρομικές συναρτήσεις:

α) Αναδρομική συνάρτηση int check_roman(char *s)

Η συνάρτηση αυτή επιστρέφει 1 όταν η συμβολοσειρά αποτελείται μόνο από ρωμαϊκά ψηφία, αλλιώς επιστρέφει 0. Μπορείτε να θεωρήσετε ότι δεχόμαστε μόνο τα κεφαλαία αντίστοιχα γράμματα: DCXI επιστρέφει 1, dCXi επιστρέφει 0, dcxi επιστρέφει 0.

β) Αναδρομική συνάρτηση int roman_to_decimal(char *s)

Η συνάρτηση αυτή δέχεται μία συμβολοσειρά που αποτελείται μόνο από ρωμαϊκά ψηφία (δηλαδή η main θα την καλέσει μόνο εφόσον έχει επιστρέψει 1 η συνάρτηση check_roman). Θα υπολογίσει τον αντίστοιχο δεκαδικό αριθμό και θα τον επιστρέψει. Μπορείτε να θεωρήσετε ότι η συμβολοσειρά που θα δοθεί τηρεί τους κανονισμούς της αναπαράστασης ρωμαϊκών αριθμών και είναι έγκυρο αριθμό.

Στην συνέχεια η main θα εκτυπώσει στην οθόνη το αποτέλεσμα της roman_to_decimal εφόσον η check_roman επέστρεψε 1, αλλιώς θα εμφανίσει κατάλληλο μήνυμα λάθους στην οθόνη και θα διαβάσει νέα συμβολοσειρά από το πληκτρολόγιο μέχρι που να δοθεί έγκυρη συμβολοσειρά.

<u>ΠΡΟΣΟΧΗ:</u> Όταν διαβάζετε μία συμβολοσειρά από το πληκτρολόγιο, διαβάζεται και το '\n' όταν πατάτε enter. Μπορείτε να υλοποιήσετε μία βοηθητική συνάρτηση που θα αντικαταστήσει αυτό το '\n' με το '\0' για να είναι κανονικό NULL-terminated string.

Σημείωση

Μία διευκρίνηση για τις ασκήσεις αναδρομής του Lab5. Αναδρομή είναι ο ορισμός μίας συνάρτησης F χρησιμοποιώντας πάλι την ίδια συνάρτηση με διαφορετικές όμως παραμέτρους. Για παράδειγμα ο ορισμός του παραγοντικού ως F(n) = n*F(n-1), n>1 και F(0)=0 είναι μια αναδρομική σχέση, ενώ ο ορισμός F(n) = 1*2*…*(n-1)*n, δεν είναι αναδρομική σχέση. Συνεπώς, στις ασκήσεις του Lab5 οι σωστές αναδρομικές λύσεις θα πρέπει να καλούν την ίδια συνάρτηση με διαφορετικές όμως τιμές στις παραμέτρους $a0, $a1 κοκ. για κάθε κλήση της F κατά την διάρκεια εκτέλεσης του προγράμματος. Λύσεις που δεν βασίζονται σε αυτήν την αρχή και που απλά κάνουν χρήση της εντολής κλήσης *jal F* σαν ένα απλό *goto* δεν θα γίνονται δεκτές.

Είναι σημαντικό η υλοποίηση των ασκήσεων να γίνει με κλήση συναρτήσεων ακριβώς όπως περιγράφει η εκφώνηση.
Κάθε συνάρτηση πρέπει να εξασφαλίζει ότι οι λεγόμενοι Saved Temporary καταχωρητές διατηρούν το περιεχόμενο που είχαν πριν την κλήση της.
Μην κάνετε άσκοπα χρήση της στοίβας. Η χωρίς όρια δέσμευση περιττής μνήμης αποτελεί προγραμματιστικό λάθος.

Θα πρέπει να στέλνετε με email τις λύσεις των εργαστηριακών ασκήσεων σας στους διδάσκοντες στο ce134lab@gmail.com.

Το email σας θα πρέπει να περιέχει ως attachment **<u>ένα zip file</u>** με τον κώδικα σας.

Κάθε διαφορετική άσκηση στην εκφώνηση θα βρίσκεται και σε διαφορετικό asm file. **<u>Το όνομα των asm files θα ΠΡΕΠΕΙ να αρχίζει με το ΑΕΜ σας.</u>**
Για παράδειγμα, το lab2.zip θα περιέχει 3 asm files, ένα για κάθε μία από τις ασκήσεις του lab2, με ονόματα 999_lab2a.asm, 999_lab2b.asm, 999_lab2c.asm για τον φοιτητή με ΑΕΜ 999.

Το email σας θα έχει Subject: CE134, lab N, Section X (N ο αριθμός του lab, N=2 ..., και X=1 έως 7).
Το email σας θα έχει body: το όνομα σας και το ΑΕΜ σας.

Θα πρέπει να στέλνετε το email σας πριν βγείτε από την εξέταση του εργαστηρίου.