Παρατηρήσεις πάνω στο lab6

Σημειώσεις σχετικά με τη βαθμολόγηση από εδώ και πέρα:

Από το lab7 και μετά, οι βαθμοί που παίρνετε από το autolab για τα επιμέρους τεστ θα είναι τελικοί. Αυτό σημαίνει ότι δε θα προσθέτουμε "χειροκίνητα" μονάδες όταν βλέπουμε ότι ο λόγος που μηδενίστηκε κάποιο τεστ είναι μικροδιαφορές στην έξοδο. Μην αφήνετε την αποστολή της άσκησής σας για το τελευταίο λεπτό γιατί τότε δεν θα έχετε τη δυνατότητα να διορθώσετε πιθανά λάθη.

Επίσης, από εδώ και στο εξής περιμένουμε ότι η στοίχιση και γενική αναγνωσιμότητα του προγράμματός σας θα πρέπει να ακολουθούν πιστά τις προδιαγραφές που αναφέρονται στο φυλλάδιο "Αρχές καλού προγραμματισμού".

Συνάρτηση εγκυρότητας εισόδου:

Ζητήθηκε να γράψετε μια συνάρτηση που παίρνει ως παραμέτρους τρεις ακεραίους, target, low, high. Ο target είναι ένας ακέραιος την τιμή του οποίου θέλουμε να ελέγξουμε και οι low, high προσδιορίζουν το επιτρεπτό εύρος του target. Ένας από τους λόγους που χρησιμοποιούμε συναρτήσεις είναι για να μην επαναλαμβάνουμε κώδικα. Επομένως, η συνάρτηση έπρεπε να κληθεί σε όλες τις περιπτώσεις ελέγχου εύρους. Στην άσκηση της Πέμπτης αυτό γινόταν για τον έλεγχο της επιλογής του χρήστη καθώς και για τον έλεγχο του ακεραίου n στο μέρος Γ. Στην άσκηση της Παρασκευής αυτό γινόταν στον έλεγχο της επιλογής του χρήστη και για τους ακεραίους που έδινε ο χρήστης για τον υπολογισμού του παραγοντικού και του e^x.

Πέρασμα παραμέτρων

Μέσω των παραμέτρων διοχετεύουμε δεδομένα σε μια συνάρτηση. Για παράδειγμα, στη συνάρτηση ελέγχου έπρεπε να περάσουμε ως παράμετρο τον ακέραιο που θέλαμε να ελέγξουμε και τις συγκεκριμένες τιμές που προσδιορίζουν το επιθυμητό εύρος.

Είναι ΛΑΘΟΣ να διαβάσετε μέσα στη συνάρτηση τις τιμές γιατί τότε αναιρείτε τις ποσότητες που έχουν περαστεί στη συνάρτηση:

```
void validate(int target, int low, int high) {
    scanf("%d", &low);  // ΛΑΘΟΣ !!
    scanf("%d", &high); // ΛΑΘΟΣ !!
    ...
}
```

Τύπος επιστροφής

Πρέπει να προσέχετε τον τύπο της ποσότητας που επιστρέφει μια συνάρτηση. Για παράδειγμα, είχατε να γράψετε μια συνάρτηση που υπολογίζει κάποιο άθροισμα, με το αποτέλεσμα να είναι πραγματικός αριθμός. Αυτό σημαίνει ότι

(α) ο τύπος επιστροφής της συνάρτησης είναι double

(β) οποιαδήποτε μεταβλητή χρησιμοποιείτε μέσα στη συνάρτηση για να αποθηκεύσετε το αποτέλεσμα που θα επιστραφεί πρέπει να είναι double.

Το (α) το είχατε γενικά σωστό, αλλά αρκετοί κάνατε λάθος το (β). Αποθηκεύσατε το αποτέλεσμα σε μεταβλητή τύπου int, με αποτέλεσμα να χαθούν τα δεκαδικά ψηφία του αποτελέσματος.

Δείτε το παρακάτω ΛΑΘΟΣ παράδειγμα:

```
double mysqrt(double num) {
       int result;  // ΛΑΘΟΣ τύπος!
       result = sqrt(num);
       return result;
}
```

Αν καλέσουμε αυτή τη συνάρτηση με num ίσο με 2.5, τότε η sqrt(num) θα επιστρέψει 1.5811 αλλά επειδή αποθηκεύουμε την ποσότητα σε ακέραιο, το result θα είναι 1. Μετά, επειδή έχουμε προσδιορίσει ότι η τιμή που επιστρέφει η συνάρτηση είναι double, τελικά θα πάρουμε πίσω την τιμή 1.0000

Κλήση συναρτήσεων και χρήση αποτελεσμάτων:

Όταν μια συνάρτηση επιστρέφει κάποιο αποτέλεσμα, τότε αυτό πρέπει να ελέγχεται είτε άμεσα είτε έμμεσα μέσω ανάθεσής του σε μεταβλητή. Για παράδειγμα, η συνάρτηση ελέγχου εγκυρότητας επέστρεφε 0 ή 1.

Μπορούσατε να ελέγξετε και να χρησιμοποιήσετε το αποτέλεσμα έμμεσα ως εξής:

```
    do {
        printMenu();
        scanf("%d", &selection);
        isOk = validate(selection, 1, 3);
    } while (!isOk);
```

ή άμεσα ως εξής:

```
    do {
        printMenu();
        scanf("%d", &selection);
    } while (!validate(selection, 1, 3));
```

Κάποιο άλλο λάθος που είδαμε αρκετές φορές ήταν να γίνεται διπλή κλήση συνάρτησης:

```
do {
    printMenu();
    scanf("%d", &selection);
    validate(selection, 1, 3)); // ΛΑΘΟΣ!
} while (!validate(selection, 1, 3));
```

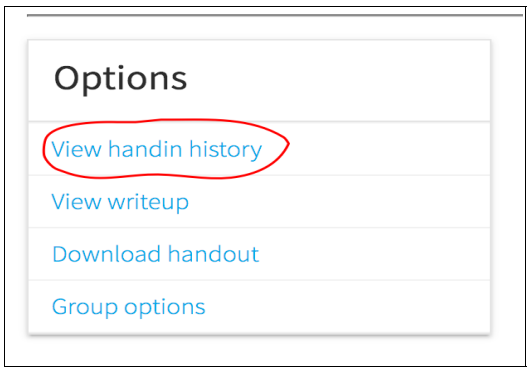
Η πρώτη κλήση της validate είναι λάθος. Κάνει τον έλεγχο, αλλά δεν αποθηκεύουμε το αποτέλεσμά της ώστε να το χρησιμοποιήσουμε. Μετά, στη συνθήκη, την ξανακαλούμε. Στο συγκεκριμένο παράδειγμα, επειδή η συνάρτηση έχει την παρανέργεια να εκτυπώνει κάποιο μήνυμα αν το selection δεν είναι κατάλληλο, το αποτέλεσμα θα είναι να εκτυπωθεί το Try Again δύο φορές χωρίς να πρέπει. Σε άλλες περιπτώσεις, η διπλή εκτέλεση μιας συνάρτησης μπορεί να έχει πιο σοβαρές επιπτώσεις.

Επαναλήψεις:

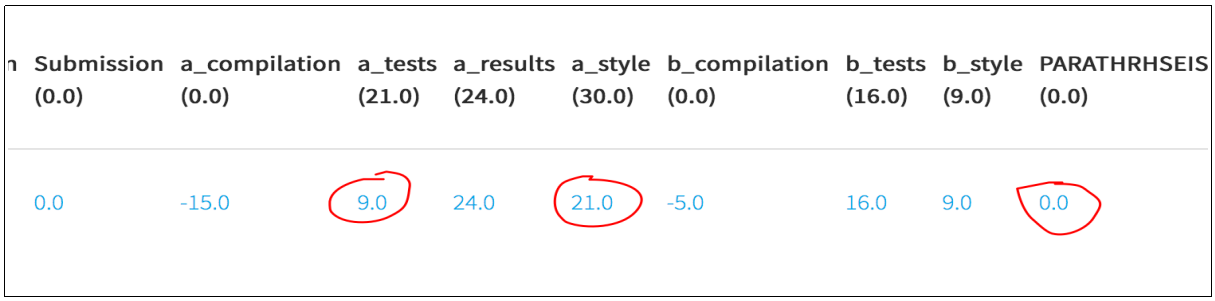
Κατά κανόνα, όταν γνωρίζουμε τον αριθμό επαναλήψεων χρησιμοποιούμε for ενώ όταν δεν τον γνωρίζουμε χρησιμοποιούμε do-while ή while ανάλογα με το αν θέλουμε να γίνει πάντα μια πρώτη επανάληψη ή όχι. Επομένως, στο lab6 έπρεπε να χρησιμοποιήσετε do-while για τον έλεγχο ορθότητας της εισόδου και for στις συναρτήσεις που υπολόγιζαν αθροίσματα.

Πώς να βλέπετε τα σχόλια βαθμολόγησης

Για να δείτε τα σχόλια βαθμολόγησης, κάντε login στο autolab, πηγαίνετε στο lab που σας ενδιαφέρει και επιλέξτε View Handin History:



Κάτω από κάθε κατηγορία βαθμολόγησης θα δείτε το βαθμό σας σε αυτή. Κάνετε κλικ πάνω στο βαθμό για να δείτε συγκεκριμένες παρατηρήσεις. Κάντε το ακόμη κι αν έχετε πάρει το μέγιστο δυνατό βαθμό.



ıı	Submission (0.0)	a_compilation (0.0)	a_tests (21.0)	a_results (24.0)	a_style (30.0)	b_compilation (0.0)	b_tests (16.0)	b_style (9.0)	PARATHRHSEIS (0.0)
	0.0	-15.0	9.0	24.0	21.0	-5.0	16.0	9.0	0.0

Ακόμη κι αν έχετε μηδέν στις PARATHRHSEIS, κάντε click σε αυτό για να δείτε αν έχουμε προσθέσει επιπλέον σχόλια για τον κώδικά σας.

Αντιστοιχία βαθμού lab6 και πόντων εργαστηρίου

Βαθμός lab6	**Πόντοι εργαστηρίου**
0-24	0
25-44	1
45-100	3