ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ
ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΑΝΑΠΤΥΞΗ ΚΑΙ ΣΧΕΔΙΑΣΗ ΛΟΓΙΣΜΙΚΟΥ

Η γλώσσα προγραμματισμού C

ΕΡΓΑΣΤΗΡΙΟ 2: Εκφράσεις, πίνακες και βρόχοι

19 Απριλίου 2018

Το σημερινό εργαστήριο έχει ως θέμα τις εκφράσεις, τους απλούς πίνακες και τους βρόχους της γλώσσας C. Οι βρόχοι είναι δομές επανάληψης, και θα είναι απαραίτητοι σε κάθε πρόγραμμα που θα γράφετε από εδώ και πέρα. Θα δείτε τους βρόχους των εντολών επανάληψης και θα μάθετε να χρησιμοποιείτε συνθήκη για έξοδο από ένα βρόχο, αλλά και τις εντολές *break* και *continue*. Από τους πίνακες της C, θα δουλέψετε προς το παρόν πάνω σε μονοδιάστατους πίνακες, και θα δείτε ότι με τη βοήθεια των δομών επανάληψης η επεξεργασία τους γίνεται πολύ απλή.

Τις παρακάτω ασκήσεις θα τις ξεκινήσετε από σήμερα, και θα τις ολοκληρώσετε μέχρι τη μεθεπόμενη εβδομάδα. Η παράδοσή τους θα γίνει όπως έγινε με το Εργαστήριο 1. Υπενθυμίζεται η στοίχιση και ο σχολιασμός που πρέπει να έχουν τα προγράμματά σας για να είναι κατανοητά. Επίσης, να έχετε υπόψη ότι στην αξιολόγηση των ασκήσεών σας θα μετράει όλο και περισσότερο η ποιότητα του κώδικα· δεν αρκεί δηλαδή να τρέχει σωστά ένα πρόγραμμα, αλλά να είναι και έξυπνα γραμμένο. Την ικανότητα να γράφετε προγράμματα με έξυπνο τρόπο θα την αποκτήσετε με την εξάσκηση. Να θυμάστε γενικά ότι έξυπνο πρόγραμμα είναι το απλό και γρήγορο πρόγραμμα.

Άσκηση 1: «Υπολογισμός τιμής μαθηματικής συνάρτησης και της παραγώγου της»

Στην άσκηση αυτή σας ζητείται να υλοποιήσετε τον κώδικα που υπολογίζει την τιμή μιας πραγματικής μαθηματικής συνάρτησης μιας μεταβλητής, για τιμή μεταβλητής που διαβάζεται από το πληκτρολόγιο, και στη συνέχεια υπολογίζει κατά προσέγγιση την τιμή της παραγώγου της συνάρτησης στην ίδια τιμή της μεταβλητής.

Η συνάρτηση που μελετάμε είναι η ακόλουθη:

$$f(x) = \begin{cases} 6x^2 + 2x - 13, & x \geq 10 \\ |60x + 7|, & -10 \leq x \leq 10 \\ -x^3 - 4x^2 + x + 3, & x \leq -10 \end{cases}$$

Δημιουργήστε την καινούργια εργασία *lab2*, καθώς και ένα καινούργιο αρχείο προγράμματος *ONOMA_lab2.1.c*, όπου αντί για *ONOMA* θα βάλετε το ονοματεπώνυμό σας. Συνδέστε το αρχείο με την εργασία και αποσυνδέστε από αυτήν το αρχείο προγράμματος που δημιουργείται αυτόματα. Εισάγετε στην αρχή του προγράμματος τη δήλωση των δύο βιβλιοθηκών *stdio* και *stdlib*, καθώς και τη δήλωση της συνάρτησης *main()*.

Εισάγετε τη δήλωση δύο τοπικών πραγματικών μεταβλητών τύπου double, έστω *x* και *y*, εντός της *main()*. Γράψτε τον κώδικα εκτύπωσης κατάλληλου μηνύματος υπόδειξης

και ανάγνωσης από το πληκτρολόγιο της τιμής της μεταβλητής *x*, με χρήση των συναρτήσεων *printf()* και *scanf()*. Προσθέστε και μια γραμμή κλήσης της *printf()* για να εκτυπώνει ως επιβεβαίωση την τιμή που διαβάστηκε.

Η διαμόρφωση εκτύπωσης της *printf()* για πραγματικούς αριθμούς είναι "%f" για μορφή με υποδιαστολή μόνο (πχ. "0.314"), "%e" για επιστημονική μορφή (πχ. "3.14e-1") ή "%g" για οποιαδήποτε μορφή προκύπτει ως απλούστερη. Πριν από τον αλφαβητικό χαρακτήρα διαμόρφωσης εκτύπωσης μπορείτε να δώσετε ακέραια σταθερά που να δηλώνει το *ελάχιστο* εύρος του πεδίου εκτύπωσης, κάτι που ισχύει και για τις διαμορφώσεις που έχετε δει στα προηγούμενα εργαστήρια. Επίσης, μπορείτε να δώσετε και μια ακέραια σταθερά μετά από '.' που να δηλώνει την επιθυμητή ακρίβεια εκτύπωσης, η οποία στην περίπτωση διαμόρφωσης πραγματικών αριθμών είναι το πλήθος των ψηφίων μετά την υποδιαστολή. Για παράδειγμα, η διαμόρφωση "%6.2f" εκτυπώνει έναν πραγματικό αριθμό με εύρος τουλάχιστον 6 χαρακτήρων, από τα οποία τα 2 είναι μετά την υποδιαστολή. Αν ο αριθμός έχει μικρότερο εύρος, το πεδίο εκτύπωσης συμπληρώνεται με κενά μέχρι το ελάχιστο εύρος.

Για ανάγνωση πραγματικών αριθμών διπλής ακρίβειας (τύπου double) με τη *scanf()*, η διαμόρφωση εκτύπωσης πρέπει να είναι "%lf". Μην ξεχάσετε τον τελεστή '&' για να δώσετε τη διεύθυνση της μεταβλητής που διαβάζετε.

Γράψτε τώρα τον κώδικα που υπολογίζει την τιμή της πιο πάνω συνάρτησης *f(x)* για την τιμή της μεταβλητής *x* που διαβάστηκε. Χρησιμοποιήστε κατάλληλα την εντολή *if*, με ή χωρίς επιλογές *else*, ώστε να κάνετε τον υπολογισμό που πρέπει για κάθε περιοχή μεταβολής του *x*. Οι υπολογισμοί σε κάθε περίπτωση θα γίνονται με τη βοήθεια των αλγεβρικών τελεστών της C, και ειδικότερα με τους τελεστές '*' (πολλαπλασιασμού), '/' (διαίρεσης), '+' (πρόσθεσης και θετικού προσήμου) και '–' (αφαίρεσης και αρνητικού προσήμου). Μπορείτε να χρησιμοποιήσετε τον τελεστή '?:' (επιλογής) για να υπολογίσετε την τιμή της συνάρτησης στο διάστημα $[-10, 10]$, δεδομένου ότι $60x + 7 \geq 0$ όταν $x \geq -7/60$.

Προσέξτε ότι αν θέλετε να εισάγετε ένα κλάσμα όπως το 7/60 σε πραγματική έκφραση, θα πρέπει να γράψετε τον αριθμητή (ή τον παρονομαστή) σε μη ακέραια μορφή, πχ. *7.0/60* ή ισοδύναμα *(double)7/60*. Το πρόγραμμα δεν αποθηκεύει την τιμή ως κλάσμα, αλλά εκτελεί τη διαίρεση και αποθηκεύει το αποτέλεσμα. Αν και οι δύο όροι του κλάσματος δίνονται ως ακέραιοι, το πρόγραμμα θα βγάλει ακέραιο αποτέλεσμα στη διαίρεση, που εδώ θα είναι *0*. Αρκεί ο ένας όρος του κλάσματος να είναι πραγματικός, ώστε η διαίρεση να είναι πραγματική και όχι ακέραια.

Αποθηκεύστε το αποτέλεσμα στη μεταβλητή *y*. Εισάγετε εκτύπωση της τιμής του *y* με κατάλληλο μήνυμα, χρησιμοποιώντας τη συνάρτηση *printf()*.

Μεταφράστε και εκτελέστε το πρόγραμμά σας.

Στη συνέχεια, με βάση τη σχέση:

$$f'(x) = \lim_{\alpha \to 0} \frac{f(x+\alpha) - f(x)}{\alpha}$$

και για τις τιμές της μεταβλητής *x* όπου η παράγωγος είναι ορισμένη, μπορείτε να υπολογίσετε την παράγωγο της συνάρτησης *f(x)* προσεγγιστικά, υπολογίζοντας το πιο πάνω πηλίκο για πολύ μικρή τιμή του *α*.

Δηλώστε μια πραγματική σταθερά *a* πριν από τη *main()*, με κάποια μικρή τιμή, έστω 0,001, ως εξής:

```
const double a = 0.001;
```

Η λέξη-κλειδί *const* ορίζει ότι στο όνομα *a* αντιστοιχείται μια σταθερή τιμή, και επομένως δεν θα επιτραπεί ανάθεση σε αυτό μετά την πιο πάνω αρχικοποίησή του. Δηλώστε επίσης τις τοπικές πραγματικές μεταβλητές *x1*, *y1* και *lim*, οι οποίες θα αποθηκεύουν τις τιμές *x+a*, *f(x+a)*, καθώς και την τιμή του πηλίκου. Γράψτε τον κώδικα υπολογισμού της τιμής του πηλίκου, χρησιμοποιώντας κατάλληλα τις μεταβλητές *x1*, *y* και *y1*. Αποθηκεύστε την τιμή αυτή στη μεταβλητή *lim*. Παρατηρήστε ότι για τον κώδικα υπολογισμού της τιμής *f(x+a)* πρέπει να αντιγράψετε τον κώδικα που γράψατε νωρίτερα, αλλάζοντας μόνο τα *x* και *y* σε *x1* και *y1*. Αργότερα στο εξάμηνο θα δείτε πώς να αποφεύγετε αυτή την αντιγραφή, ορίζοντας συναρτήσεις για την υλοποίηση κώδικα που εκτελείται πάνω από μία φορά.

Γράψτε τον κώδικα εκτύπωσης της τιμής της μεταβλητής *lim* με κατάλληλο μήνυμα, και ολοκληρώστε το πρόγραμμα με τη γνωστή εντολή επιστροφής και τερματισμού του προγράμματος.

Μεταφράστε και εκτελέστε τον κώδικά σας.

Στη συνέχεια προσθέστε τον κώδικα υπολογισμού της ακριβούς τιμής της παραγώγου της συνάρτησης *f(x)*, ώστε να τη συγκρίνετε με την προσέγγιση που υπολογίσατε. Η συνάρτηση της παραγώγου είναι:

$$f'(x) = \begin{cases} 12x + 2, & x > 10 \\ 60, & -7/60 < x < 10 \\ -60, & -10 < x < -7/60 \\ -3x^2 - 8x + 1, & x < -10 \end{cases}$$

Γράψτε λοιπόν τον κώδικα που υπολογίζει την παραπάνω συνάρτηση για την τιμή του *x* που έχετε διαβάσει. Χρησιμοποιήστε κάποια νέα τοπική μεταβλητή, έστω *df* για να αποθηκεύσετε το αποτέλεσμα του υπολογισμού. Προσθέστε κώδικα εκτύπωσης του αποτελέσματος αυτού, καθώς και της διαφοράς μεταξύ της ακριβούς τιμής και της προσεγγιστικής που υπολογίσατε νωρίτερα. Μεταφράστε και εκτελέστε το πρόγραμμά σας. Μπορείτε να δοκιμάσετε και άλλες τιμές του α, ώστε να προσπαθήσετε να επιτύχετε μεγαλύτερη ακρίβεια στην προσέγγισή σας.

Για να ολοκληρώσετε την άσκηση, θα πρέπει να προσθέσετε έλεγχο για τα σημεία στα οποία η παράγωγος δεν ορίζεται. Τα σημεία αυτά είναι τα -10, $-7/60$ και $+10$. Αν λοιπόν η μεταβλητή *x* έχει μία από αυτές τις τιμές, δεν θα πρέπει να υπολογίζεται η παράγωγος. Προσθέστε τον κώδικα που εκτελεί αυτόν τον έλεγχο μετά τον υπολογισμό της τιμής της *f(x)*, και τερματίζει την εκτέλεση του προγράμματος σε περίπτωση αληθούς συνθήκης.

Μεταφράστε και εκτελέστε το πρόγραμμα.

Αποθηκεύστε το αρχείο του προγράμματος και προχωρήστε στην επόμενη άσκηση. Μην ξεχάσετε να αντιγράψετε το αρχείο σε ασφαλή αποθήκευση.

Άσκηση 2: «Ταξινόμηση ακεραίων»

Δημιουργήστε ένα καινούργιο αρχείο προγράμματος *ONOMA_lab2.2.c*, όπου αντί για *ONOMA* θα βάλετε το ονοματεπώνυμό σας. Συνδέστε το με την εργασία *lab2* και αποσυνδέστε από αυτήν το αρχείο προγράμματος της προηγούμενης άσκησης.

Στην άσκηση αυτή θα γράψετε ένα πρόγραμμα ταξινόμησης ακεραίων σε αύξουσα σειρά, το οποίο να υλοποιεί διαδοχικά δύο αλγορίθμους, τους οποίους και θα πρέπει να

συγκρίνετε. Και οι δύο αλγόριθμοι είναι απλοί, και βασίζονται σε συγκρίσεις και εναλλαγές θέσεων στους αριθμούς που ταξινομούνται. Για την υλοποίησή τους δεν θα σας δοθούν συγκεκριμένες οδηγίες, παρά μόνο οι ίδιοι οι αλγόριθμοι, δηλαδή μια περιγραφή υψηλού επιπέδου (σχεδόν σε φυσική γλώσσα, με προστακτική μορφή) του ζητούμενου κώδικα.

Κατ' αρχήν, θα πρέπει να δηλώσετε έναν πίνακα ακεραίων, μεγέθους τουλάχιστον n. Για το σκοπό αυτό, θα χρησιμοποιήσετε μια νέα οδηγία προς τον προεπεξεργαστή, την οδηγία *define*, με την οποία θα δηλώσετε ως σταθερά τη μέγιστη επιτρεπτή τιμή του n, για παράδειγμα:

```
#define n_max 100
```

Με την οδηγία αυτή ορίσατε μια σταθερά με όνομα *n_max* και τιμή 100, ως το άνω φράγμα για το πλήθος των αριθμών που θα ταξινομήσετε, δηλαδή ένα μέγιστο n για το οποίο το πρόγραμμα θα μπορεί να δουλεύει.

Οι σταθερές που ορίζονται με τον τρόπο αυτόν είναι τιμές που αντιστοιχίζονται απ' ευθείας στο όνομά τους από τον προεπεξεργαστή, ώστε ο μεταγλωττιστής να βλέπει μόνο την τιμή αντί του ονόματος. Αντίθετα, οι σταθερές που ορίζονται με δήλωση *const* δεν αντιστοιχίζονται με αυτόν τον τρόπο με το όνομά τους, αλλά ο μεταγλωττιστής τις διαχειρίζεται ως μεταβλητές που δεν επιτρέπεται να αλλάζουν τιμή. Το νόημα της δήλωσης *define* είναι να έχετε την τιμή της σταθεράς *n_max* γραμμένη σε ένα μόνο σημείο του προγράμματος, ώστε να την αλλάξετε εύκολα αν το θελήσετε.

Έχοντας ορίσει τη σταθερά *n_max* μέσω οδηγίας *define*, μπορείτε να ορίσετε πίνακα ακεραίων – έστω *akeraioi* – μεγέθους *n_max*:

```
int akeraioi[n_max];
```

Αν η σταθερά *n_max* ήταν δηλωμένη μέσω δήλωσης *const*, δεν θα μπορούσατε να ορίσετε τον πίνακα με αυτό τον τρόπο, μια που ο μεταγλωττιστής δεν θα δεχόταν μεταβλητή ως μέγεθος πίνακα[1]!

Με την παραπάνω δήλωση, μπορείτε να δεχτείτε από το πληκτρολόγιο οποιαδήποτε τιμή για το n μέχρι τη σταθερά *n_max*. Δηλώστε λοιπόν τη μεταβλητή *n*, τοπικά στην *main()*, και γράψτε τον κώδικα με τον οποίο διαβάζετε από το πληκτρολόγιο την τιμή της, καθώς και τον κώδικα με τον οποίο ελέγχετε αν η τιμή είναι επιτρεπτή, δηλαδή μικρότερη ή ίση με τη σταθερά *n_max*.

Στη συνέχεια, διαβάστε από το πληκτρολόγιο *n* ακέραιες τιμές, και αποθηκεύστε τες σε συνεχόμενες θέσεις του πίνακα ακεραίων.

Ο πρώτος αλγόριθμος ταξινόμησης είναι ο εξής:

1. Αντιγράψτε τον πίνακα ακεραίων σε κάποιο νέο πίνακα Α.
2. Για κάθε θέση i του πίνακα Α από το τέλος προς την αρχή:
   Για κάθε θέση j του πίνακα Α από την αρχή μέχρι την i:
   Αν η τιμή στη θέση i είναι μικρότερη της τιμής στη θέση j, τότε άλλαξε μεταξύ τους τις δύο τιμές στον πίνακα Α.

Ο δεύτερος αλγόριθμος ταξινόμησης είναι ο εξής:

1. Αντιγράψτε τον πίνακα ακεραίων σε κάποιο νέο πίνακα Β.

[1] Η έκδοση C99 της C επιτρέπει τέτοια δήλωση και τη χειρίζεται ως δυναμικής μνήμης. Ο μεταγλωττιστής gcc υποστηρίζει την έκδοση C99, αλλά επειδή η δυναμική μνήμη είναι άλλο κεφάλαιο στη C, προτείνεται να μη χρησιμοποιείτε τέτοια δήλωση προς το παρόν.

2. Όσο γίνεται εναλλαγή τιμών:
    Για κάθε θέση i του πίνακα Β από την αρχή προς το τέλος:
        Αν η τιμή στη θέση i είναι μεγαλύτερη της τιμής στη θέση i+1, τότε άλλαξε μεταξύ τους τις δύο τιμές στον πίνακα Β.

Στη συνέχεια υλοποιήστε τους δύο πιο πάνω αλγόριθμους. Παρατηρήστε ότι απαιτούνται διπλά φωλιασμένοι βρόχοι και για τους δύο αλγόριθμους. Για όλους τους βρόχους, θα πρέπει να δηλώσετε και να χρησιμοποιήσετε κατάλληλες τοπικές ακέραιες μεταβλητές ως μετρητές των επαναλήψεων. Επίσης, χρησιμοποιήστε κατάλληλα τις εντολές *break* – για διακοπή – και *continue* – για συνέχεια στην επόμενη επανάληψη, όπου και αν νομίζετε ότι είναι χρήσιμες.

Προσθέστε τον κώδικα που εκτυπώνει τον αρχικό πίνακα, καθώς και τους πίνακες Α και Β. Ας σημειωθεί ότι η αντιγραφή του αρχικού πίνακα έγινε ώστε να μη χαθεί ο αρχικός πίνακας και δεν είναι ουσιαστικά μέρος της ταξινόμησης!

Μεταφράστε και δοκιμάστε τον κώδικά σας.

Αντί να εκτυπώνετε και τους δύο πίνακες Α και Β, προσθέστε τον κώδικα που ελέγχει αν οι δύο πίνακες είναι οι ίδιοι, ώστε αν δεν είναι ίδιοι, να βγάζετε μήνυμα σφάλματος και να βγαίνετε από το πρόγραμμα. Στη συνέχεια κρατήστε τον κώδικα που εκτυπώνει τον ένα μόνο από τους δύο πίνακες.

Μεταφράστε και δοκιμάστε τον κώδικά σας.

Προσθέστε τώρα δύο ακέραιες μεταβλητές, ώστε να μετράτε τις εναλλαγές τιμών που κάνατε σε κάθε αλγόριθμο, και στο τέλος – μετά την εκτύπωση των ταξινομημένων αριθμών – να εκτυπώνετε και το αποτέλεσμα των μετρήσεων.

Μεταφράστε, δοκιμάστε τον κώδικά σας και αποθηκεύστε το αρχείο του προγράμματος.

Μην ξεχάσετε να υποβάλετε τις ασκήσεις σας *κατά προτίμηση* εντός της προθεσμίας.