Εργαστήριο #6: Έξτρα άσκηση

ΜΕΡΟΣ Α

Καλείστε να γράψετε ένα πρόγραμμα το οποίο υπολογίζει τον φυσικό λογάριθμο ln(x) με δύο τρόπους: **(α)** χρησιμοποιώντας τη συνάρτηση log από το math.h και **(β)** υπολογίζοντας το αποτέλεσμα της παρακάτω μαθηματικής σειράς.

$$\ln(x) = 2\sum_{n=1}^{N} \frac{\left((x-1)/(x+1)\right)^{2n-1}}{(2n-1)}, \gamma\iota\alpha\ x>0$$

Γράψτε μια συνάρτηση η οποία παίρνει ως παραμέτρους έναν αριθμό κινητής υποδιαστολής που αποτελεί τη βάση, και έναν ακέραιο που αποτελεί τον εκθέτη και επιστρέφει το αποτέλεσμα της ύψωσης σε δύναμη του αριθμού της βάσης στον εκθέτη. Δεν επιτρέπεται η χρήση συναρτήσεων από το math.h σε αυτή τη συνάρτηση.

Γράψτε μια δεύτερη συνάρτηση η οποία παίρνει ως παραμέτρους έναν αριθμό κινητής υποδιαστολής (το x) και έναν ακέραιο (το n) και υπολογίζει κι επιστρέφει το αποτέλεσμα της παρακάτω έκφρασης η οποία όπως βλέπετε είναι ένας όρος του παραπάνω αθροίσματος. Για την ύψωση σε δύναμη χρησιμοποιήστε την προηγούμενη συνάρτηση που κατασκευάσατε.

$$\frac{\left((x-1)/(x+1)\right)^{2n-1}}{(2n-1)}, \gamma\iota\alpha\ x>0$$

Γράψτε μια τρίτη συνάρτηση η οποία παίρνει ως παραμέτρους ένα θετικό αριθμό κινητής υποδιαστολής και το επιθυμητό πλήθος όρων της σειράς (η ποσότητα N στον τύπο) και υπολογίζει κι επιστρέφει το ln(x) σύμφωνα με τον τύπο που δίνεται παραπάνω, χρησιμοποιώντας τη δεύτερη συνάρτηση για τον υπολογισμό κάθε όρου του αθροίσματος.

Γράψτε τη συνάρτηση main η οποία κάνει τα εξής:

1. Εκτυπώνει το μήνυμα "`Enter x:\n`" και διαβάζει ένα θετικό αριθμό κινητής υποδιαστολής διπλής ακρίβειας. Δε χρειάζεται να κάνετε έλεγχο για το αν είναι θετικός.
2. Υπολογίζει το φυσικό λογάριθμο του x αναλυτικά μέσω των παραπάνω συναρτήσεων, δίνοντας 10 ως το πλήθος όρων της σειράς, και εκτυπώνει το μήνυμα "`series: Y\n`", όπου Y ο φυσικός λογάριθμος που υπολογίστηκε, με 15 δεκαδικά ψηφία.
3. Υπολογίζει το φυσικό λογάριθμο μέσω της συνάρτησης log που ορίζεται στο math.h και εκτυπώνει το μήνυμα "`  math: Y\n`", όπου Y ο φυσικός λογάριθμος που υπολογίστηκε, με 15 δεκαδικά ψηφία.

Ακολουθούν παραδείγματα εκτέλεσης. Η είσοδος του χρήστη εμφανίζεται με κόκκινο χρώμα:

Enter x:
0.8
series: -0.223143551314210
  math: -0.223143551314210
	Enter x:
	3.1
	series: 1.131402012364728
	  math: 1.131402111491101

ΜΕΡΟΣ Β

Στο μέρος Α είδαμε πώς μπορούμε να υπολογίσουμε το φυσικό λογάριθμο ενός αριθμού x χρησιμοποιώντας είτε μια σειρά είτε τη συνάρτηση που παρέχει η μαθηματική βιβλιοθήκη, κρατώντας σταθερό τον αριθμό επαναλήψεων.

Σε αυτό το μέρος θα εξετάσουμε πόσο πρέπει να είναι το N ώστε να πλησιάσουμε στην ποσότητα που μας δίνει η συνάρτηση της μαθηματικής βιβλιοθήκης, εντός κάποιας απόστασης που θεωρούμε ικανοποιητική. Αυτή η απόσταση ονομάζεται προσεγγιστικό σφάλμα.

Γράψτε ένα νέο πρόγραμμα το οποίο λειτουργεί ως εξής:

1. Εκτυπώνει το μήνυμα "`Enter x:\n`" και διαβάζει ένα θετικό αριθμό κινητής υποδιαστολής διπλής ακρίβειας.
2. Εκτυπώνει το μήνυμα "`Enter acceptable error:\n`" και διαβάζει ένα θετικό αριθμό κινητής υποδιαστολής διπλής ακρίβειας, ο οποίος είναι το αποδεκτό σφάλμα
3. Υπολογίζει το φυσικό λογάριθμο του αριθμού που διάβασε στο βήμα 1 χρησιμοποιώντας τη συνάρτηση log της μαθηματικής βιβλιοθήκης.
4. Σε επανάληψη, ξεκινώντας από N ίσο με 10:
    a. Υπολογίζει το φυσικό λογάριθμο του αριθμού χρησιμοποιώντας την τρίτη συνάρτηση που γράψατε στο μέρος Α.
    b. Υπολογίζει την απόλυτη τιμή της διαφοράς ανάμεσα στις δύο τιμές του λογαρίθμου.
    c. Εκτυπώνει το μήνυμα: "N, E" όπου N η τρέχουσα τιμή του N και E η ποσότητα που υπολογίστηκε στο βήμα 4b με 20 δεκαδικά ψηφία.
    d. Εφόσον αυτή η ποσότητα είναι μεγαλύτερη του αποδεκτού σφάλματος, αυξάνει το N και συνεχίζει την επανάληψη, διαφορετικά σταματάει.
5. Όταν σταματήσει η επανάληψη, εκτυπώστε τα μηνύματα των βημάτων 2 και 3 του πρώτου μέρους, για να δείτε πόσο πλησιάσατε. Χρησιμοποιήστε 20 δεκαδικά ψηφία.

Δοκιμάστε να τρέξετε το πρόγραμμά σας για x ίσο με 3.1 και σφάλμα ίσο 1E-10 (αυτό είναι το 1^{-10} με επιστημονική γραφή και διαβάζεται κανονικά από τη scanf με χρήση %lf). Θα πρέπει να βρείτε N=15.

Για δυνατούς λύτες: Διαβάστε το μέρος Γ πριν κάνετε δοκιμές για μεγαλύτερο σφάλμα.

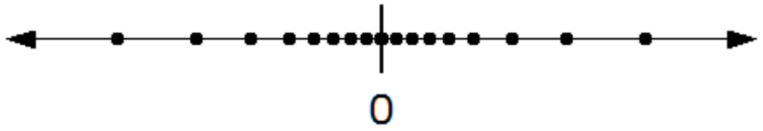
ΜΕΡΟΣ Γ

Αν δοκιμάσετε να δώσετε ως σφάλμα το 1E-20 θα παρατηρήσετε ότι η επανάληψη του βήματος 4 δε φαίνεται να τερματίζει.

Αυτό οφείλεται στο εξής πρόβλημα: Όταν υπολογίζετε το άθροισμα στην τρίτη συνάρτηση που γράψατε στο μέρος Α, σε κάθε επανάληψη προσθέτετε στο άθροισμα ένα νέο όρο για n μεγαλύτερο κατά 1. Όταν το n είναι μεγάλο, αυτός ο νέος όρος είναι τόσο πολύ μικρότερος από το άθροισμα από το άθροισμα που έχει υπολογιστεί μέχρι στιγμής, ώστε πρακτικά είναι σα να προσθέτετε μηδέν στο άθροισμα.

Για παράδειγμα, για x = 3.1 και n=100, το άθροισμα που έχει υπολογιστεί μέχρι και n=99 είναι περίπου ίσο με 1.131402111491100370699314225931 ενώ ο όρος για n=100 που θα προστεθεί σε αυτό το άθροισμα είναι περίπου $1.5*1^{-58}$. Εαν προστεθούν αυτοί οι δύο αριθμοί μεταξύ τους, το αποτέλεσμα παραμένει 1.131402111491100370699314225931

Διαισθητικά, σκεφτείτε το ως εξής: Έχετε δει στο μάθημα μια εικόνα σαν τη παρακάτω που παρουσιάζει (ως κουκίδες) τους αριθμούς κινητής υποδιαστολής που μπορούν να αναπαρασταθούν ακριβώς:



Βλέπετε πως όσο απομακρυνόμαστε από το μηδέν, είναι πιο αραιά οι αριθμοί που μπορούν να αναπαρασταθούν. Τι γίνεται λοιπόν αν πάτε να προσθέσετε σε έναν αριθμό που είναι μεγάλος (ας πούμε εκεί που είναι η δεξιότερη κουκίδα) έναν που είναι πολύ πολύ μικρός? Ο κοντινότερος αριθμός στο αποτέλεσμα της πρόσθεσης είναι τελικά ο ίδιος μεγάλος αριθμός που έχουμε (η δεξιότερη κουκίδα του σχήματος), επομένως το άθροισμα παραμένει ίσο με τον μεγάλο αριθμό.

Για μια πιο τεχνική ερμηνεία, δείτε τη σελίδα http://www.ecs.umass.edu/ece/koren/arith/simulator/FPAdd/ η οποία παρουσιάζει πώς γίνεται πρόσθεση δύο αριθμών κινητής υποδιαστολής. Όπως θα μάθετε στο μάθημα της Οργάνωσης στο 3ο εξάμηνο, κάθε αριθμός κινητής υποδιαστολής γράφεται στο δυαδικό σύστημα ως $1.x * 2^y$. Για να προστεθούν δύο αριθμοί, πρέπει να έχουν την ίδια δύναμη. Βάλτε στην παραπάνω σελίδα το 1.5E-58 στο πεδίο A, το 1.1314 στο πεδίο B, επιλέξτε dec, add και double, πατήστε compute και δείτε δεξιά τις πράξεις που γίνονται εσωτερικά. Αρχικά ας δείχνει τους αριθμούς που εισάγατε, στο δυαδικό σύστημα, όπως δηλαδή είναι αποθηκευμένοι στη μνήμη. Το alignment step είναι η μετατροπή ώστε να έχουν ίδιο εκθέτη. Όπως βλέπετε, το A έγινε μηδέν.

Το ερώτημα τώρα είναι πώς τελικά μπορούμε να υπολογίσουμε το άθροισμα ώστε να αποφύγουμε αυτό το πρόβλημα. Η απάντηση είναι ότι πρέπει να αποφύγουμε την προσθήκη πολύ μικρών αριθμών σε πολύ μεγάλους αριθμούς. Εφόσον το άθροισμά μας έχει τη μορφή
"πολύ μεγάλος αριθμός" + "ένα σωρό όλο και πιο μικροί αριθμοί"
μας συμφέρει να προσθέσουμε τους όρους ανάποδα: ξεκινώντας από Ν και κατεβαίνοντας προς το 1, προσθέτουμε μικρούς αριθμούς με μικρούς αριθμούς. Προσθέτοντας αριθμούς που καταλήγουν να έχουν κοντινούς εκθέτες (ή αν προτιμάτε το σχήμα, αριθμούς που είναι σε σημείο με πολλές κουκίδες), το άθροισμα αυξάνεται ομαλά και τα αποτελέσματα μπορούν να αναπαρασταθούν με καλύτερη ακρίβεια.

Αλλάξτε λοιπόν την τρίτη συνάρτηση που γράψατε, ώστε η επανάληψη να μην πηγαίνει από 1 έως και το πάνω όριο, αλλά από το πάνω όριο έως και 1. Μετά δοκιμάστε να τρέξετε το μέρος Β για x ίσο με 3.1 και σφάλμα ίσο 1E-20. Θα δείτε ότι το Ν βγαίνει 25 και ο αριθμός που προέκυψε από τη δική σας συνάρτηση φαίνεται ίδιος με αυτόν που δίνει η log.