Γενικά σχόλια lab8

Μεταγλώττιση: Ο κώδικας θα πρέπει να μεταγλωττίζεται χωρίς **warnings**. Χαρακτηρισμός “μη ικανοποιητικό” σημαίνει ότι είχατε τουλάχιστον ένα warning. Κώδικας που παράγει λάθη (errors) κατά τη μεταγλώττιση βαθμολογείται με FAIL.

Στοίχιση: Ο κώδικας πρέπει να είναι στοιχισμένος σωστά, με ιδιαίτερη προσοχή στη στοίχιση εμφωλευμένων for/while/if.
Χαρακτηρισμός “μέτρια” γενικά σημαίνει ότι έχετε κατά το πλείστον καλή στοίχιση Χαρακτηρισμός “μη ικανοποιητική” σημαίνει ότι δεν έχετε στοιχίσει τον κώδικά σας ή η στοίχιση είναι πολύ ασυνεπής ειδικά όσον αφορά τις δομές ελέγχου/επανάληψης.

Ονόματα: Όπως πάντα, θέλουμε περιγραφικές μεταβλητές και παραμέτρους με κατάλληλους τύπους. Τα ονόματα των συναρτήσεων πρέπει επίσης να είναι περιγραφικά και αντιπροσωπευτικά της λειτουργίας της συνάρτησης. Για παράδειγμα, το όνομα metatropi δεν είναι καλό γιατί δε μας λέει τι είδους μετατροπή γίνεται. Ένα καλύτερο όνομα είναι convert2rad ή degrees2rad. Συχνά διαλέγουμε ρηματικές φράσεις ως ονόματα συναρτήσεων διότι κάθε συνάρτηση κάνει κάποια ενέργεια. Για παράδειγμα get_limit και όχι απλά limit.
Ονόματα τοπικών μεταβλητών διαφορετικών συναρτήσεων μπορούν να είναι ίδια, και ενδείκνυται κάτι τέτοιο αν πρόκειται για όμοιες ποσότητες. Αν τη μεταβλητή για το όριο τη λέτε limit στη main, limit1 στη συνάρτηση που τη διαβάζει και limit2 στη συνάρτηση υπολογισμού του αθροίσματος, τότε το πρόγραμμα είναι πιο δύσκολο στην κατανόηση από ότι αν είχατε limit σε όλες αυτές τις περιπτώσεις.
Ονόματα του ενός γράμματος πρέπει να αποφεύγονται με μόνη εξαίρεση μετρητές για for loops. Επομένως, το π πρέπει να λέγεται PI και όχι απλά p.

Σταθερές: Το π πρέπει να οριστεί ως σταθερά (είτε με #define είτε μέσα στη συνάρτηση που χρησιμοποιείται με const)

Ορισμός και κλήσεις συναρτήσεων: Για κάθε συνάρτηση ελέγχουμε προσεκτικά το prototype (παράμετροι και τύποι αυτών, τύπος επιστροφής), αν καλείται σωστά και, στην περίπτωση που επιστρέφει κάτι, αν η τιμή επιστροφής αποθηκεύεται σε κατάλληλη μεταβλητή. Ο τύπος της μεταβλητής που περιέχει την τιμή επιστροφής πρέπει να είναι ίδιος με τον τύπο επιστροφής της συνάρτησης. Για παράδειγμα, εφόσον η συνάρτηση παραγοντικού επιστρέφει long int, θα πρέπει η τοπική μεταβλητή στην οποία αποθηκεύεται το αποτέλεσμα πριν επιστραφεί να είναι επίσης long int.
Ένα συχνό λάθος είναι να καλείται 2 φορές μια συνάρτηση, με το αποτέλεσμα της πρώτης κλήσης να μη χρησιμοποιείται. Για παράδειγμα:
factorial(n);
x = factorial(n);

Αυτό είναι ακόμη χειρότερο αν η συνάρτηση παίρνει είσοδο. Για παράδειγμα, αν η συνάρτηση που διαβάζει τη γωνία λέγεται get_angle και καλέσουμε στη main:
my_cos = cosine(get_angle(), get_limit());
pc_cos = cos(get_angle());
τότε το πρόγραμμα θα ζητήσει 2 γωνίες από το χρήστη και θα χρησιμοποιήσει τη μία στον υπολογισμό συνημιτόνου με άθροισμα και την άλλη στον υπολογισμό με χρήση της συνάρτησης cos.

Ορθότητα αποτελεσμάτων: Ελέγχουμε αν γίνεται σωστά ο υπολογισμός παραγοντικού και της σειράς. Ιδιαίτερη προσοχή στο αν αρχικοποιούνται σωστά οι μεταβλητές (accumulators) στις οποίες αποθηκεύεται το επιμέρους αποτέλεσμα κατά τη διάρκεια του υπολογισμού (το άθροισμα σε μηδέν, το γινόμενο σε 1).
Ένα πολύ συχνό λάθος (και λόγος για οριακό pass) ήταν συνθήκες της μορφής angle < 0 && angle > 360. Μια τέτοια συνθήκη είναι πάντα ψευδής, επομένως δε θα λειτουργήσει σωστά η επανάληψη στην οποία χρησιμοποιείται.
Ένα άλλο συχνό λάθος ήταν στα όρια της επανάληψης που υπολογίζει το άθροισμα. Το n παίρνει τιμές από 0 μέχρι ΚΑΙ το όριο, επομένως το for loop έπρεπε να ήταν for (n=0; n<=limit; n++) και όχι for(n=0; n<limit; n++)

Έξοδος προγράμματος: Η έξοδος του προγράμματος πρέπει να είναι ίδια με την εκφώνηση. Σημείωση: επειδή είχαμε ξεχάσει να ζητήσουμε 15 δεκαδικά και στο δεύτερο αποτέλεσμα που εκτυπώνεται και το είπαμε μόνο προφορικά στο εργαστήριο, γίνονται δεκτές και οι δύο περιπτώσεις (με ή χωρίς 15 δεκαδικά για το δεύτερο νούμερο).