

Συμπληρωματικές Ασκήσεις 3

Επιμέλεια : Αλέξανδρος Αργυρόπουλος

Επικοινωνία: alex.argiropoulos2004@gmail.com

1. Φτιάξτε ένα πρόγραμμα που χρησιμοποιεί το module random , για να φτιάξει μία τυχαία λίστα ακεραίων . Έπειτα θα ζητάει από τον χρήστη να εισαγει έναν αριθμό , θα εκτυπώνει το πλήθος των εμφανίσεών του στη λίστα και θα εμφανίζει μία νέα εκδοχή της λίστας , έχοντας αφαιρέσει εντελώς τον αριθμό από τη λίστα .

π.χ. $[1, 4, 1, 5, 16] \xrightarrow{1} 2$ και $[4, 5, 16]$

2. **Πρόβλημα** (Δημιουργία module πράξεων σε λίστες) : Να φτιάξετε ένα αρχείο list_operations στο οποίο θα εισάγεται τις εξής πράξεις / συναρτήσεις :

Έστω l_1, l_2 λίστες ακεραίων αριθμών .

Δ Add(l_1, l_2) : Προσθέτει στην l_1 τα στοιχεία της l_2 που δεν περιέχονται σε αυτή α) Κάθε φορά στο τέλος , β) Χρησιμοποιώντας την τιμή ως δείκτη εισόδου (π.χ. αν βρεθεί η τιμή 13 θα πρέπει να εισαχθεί ως στοιχείο με δείκτη 13 . Δεν είναι απαραίτητο πως τελικά θα βρεθεί στο δείκτη 13 .) και επιστρέφει την τροποποιημένη l_1 .

Δ Mult(l_1, l_2) : Επιστρέφει νέα λίστα η οποία προκύπτει αντιγράφοντας κάθε στοιχείο της l_1 $|l_2[i]|$ φορές , όπου i είναι ο δείκτης του στοιχείου . Αν $len(l_1) < len(l_2)$ τότε χρησιμοποιούνται μόνο τα πρώτα $len(l_1)$ στοιχεία της l_2 , αντίστοιχα αν $len(l_2) < len(l_1)$ τότε τα τελευταία $len(l_1) - len(l_2)$ στοιχεία παραμένουν ίδια .

π.χ. Mult([2,5], [1,3]) = [2,5,5,5] Mult([1,7], [2,-2,15]) = [1,1,7,7] Mult([0,1,2], [-3,4]) = [0,0,0,1,1,1,2]

Επιπλέον, τώρα θεωρούμε ότι οι λίστες έχουν ίδιο πλήθος στοιχείων τότε :

Δ point_add(l_1, l_2) : Επιστρέφει λίστα της οποίας το i -οστο στοιχείο της είναι το άθροισμα των αντίστοιχων στοιχείων των l_1, l_2 .

Δ point_mult(l_1, l_2) : Επιστρέφει λίστα της οποίας το i -οστο στοιχείο της είναι το γινόμενο των αντίστοιχων στοιχείων των l_1, l_2 .

Αν ένα από τα δύο ορίσματα είναι ακέραιος τότε κάθε στοιχείο της λίστας θα πολλαπλασιαστεί με αυτόν .

Πώς μπορεί να προκύψει η δεύτερη λειτουργία χρησιμοποιώντας την πρώτη ;

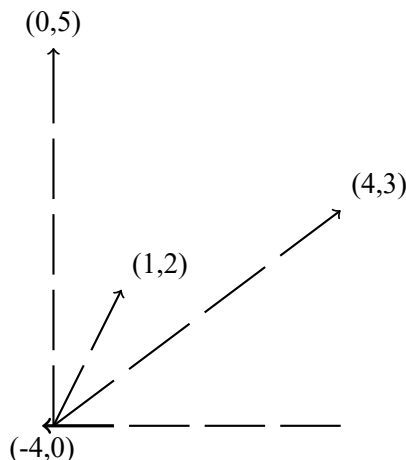
3. **Εφαρμογές** του Προβλήματος 2 :

- Να φτιάξετε ένα πρόγραμμα το οποίο θα έχει τις εξής λειτουργίες : Θα δημιουργεί μία λίστα με τις βαθμολογίες από 0 - 10, θα βάζει το χρήστη να δημιουργεί μία λίστα ακεραίων με βάση το πλήθος των μαθημάτων στα οποία πήρε τον κάθε βαθμό . Έπειτα θα εκτυπώνει τη λίστα όλων των βαθμολογιών , το πλήθος των μαθημάτων που δεν πέρασε , το ποσοστό εμφάνισης κάθε βαθμού , το ποσοστό αριστείας (9,10) και τέλος το μέσο όρο (με βάση τις ακαδημαϊκές στρογγυλοποιήσεις) .

- Χρησιμοποιώντας κατάλληλα το module `list_operations` να φτιαχτεί συνάρτηση `point_diff( $l_1, l_2$ )` : Επιστρέφει νέα λίστα, της οποίας το i -οστό στοιχείο της είναι η διαφορά των αντίστοιχων στοιχείων των l_1, l_2 . Έπειτα να εμπλουτιστεί το πρόγραμμα ώστε να υπολογίζει τις συντεταγμένες ενός διανύσματος (με άκρα, σημεία δοσμένα από το χρήστη) . Για το διάνυσμα αυτό θα εκτυπώνει τις συντεταγμένες , το μέτρο του και εφόσον είναι δύο διαστάσεων , θα εκτυπώνει το συντελεστή διεύθυνσης αυτού και των κάθετων σε αυτό διανυσμάτων (όταν ορίζεται) και τέλος θα το αναπαριστά γραφικά. Σημείωση: Όπως στο Πρόβλημα 3 οι είσοδοι πρέπει να είναι ακέραιοι αριθμοί.

Παράδειγμα γραφικής αναπαράστασης :

Προτείνεται η γραφική αναπαράσταση να γίνεται με `-->` (`<...`), σε περίπτωση οριζόντιου , με `|` και `^` , `v` για κατακόρυφα και για πλάγια `/` , `-|` , `|_` . Στα πλάγια το πλήθος των `/` να καθορίζεται από την μεγαλύτερη συντεταγμένη .



- (Επεξεργασία πολωνύμων) : Να φτιάξετε πρόγραμμα που θα δέχεται τους συντελεστές δύο πολωνύμων P, Q , ως δύο λίστες πραγματικών αριθμών . Το πλήθος των συντελεστών που θα εισαχθούν για κάθε πολώνυμο θα πρέπει να ζητηθεί από το χρήστη. Το πρόγραμμα θα έχεις τις εξής συναρτήσεις :
 - α) $P(x)$: Θα επιστρέφει την τιμή του P στο x ,
 - β) $\text{add}(P, Q)$: Θα επιστρέφει τη λίστα συντελεστών για το άθροισμα και αντίστοιχα
 - γ) $\text{dif}(P, Q)$ για τη διαφορά ,
 - δ) $\text{deg}(P)$: Θα επιστρέφει το βαθμό του πολωνύμου ,
 - ε) $\text{main}()$: θα βάζει το χρήστη να εισάγει , επιπλέον , έναν αριθμό x_0 και θα εκτυπώνει τα πολώνυμα του αθροίσματος και της διαφοράς, τους βαθμούς των $P, Q, P + Q, P - Q$ και τις τιμές τους στο x_0 .

- Να φτιαχτεί πρόγραμμα που θα διαβάζει τους συντελεστές δύο πολωνύμων P, Q και θα εκτυπώνει το όριο στο $\pm\infty$ των $P + Q, P - Q, P \cdot Q, \frac{P}{Q}$. Να εφοδιάσετε το πρόγραμμα με μία συνάρτηση $\text{limit}(P)$ που να επιστρέφει σε λίστα τα όρια του πολωνύμου (στο $\pm\infty$) . Επίσης θα χρειαστεί μία $\text{rational_lim}(P, Q)$ για τα όρια της ρητής πολωνυμικής. Τέλος θα περιέχει μία $\text{main}()$ στην οποία θα γίνεται η είσοδος και η έξοδος , χρησιμοποιώντας κατάλληλα τις άλλες συναρτήσεις .

Σχόλιο: Μπορεί να φανούν χρήσιμες κάποιες από τις συναρτήσεις της προηγούμενης εφαρμογής .

5. **Πρόβλημα** (Ανάλυση ενός φυσικού σε γινόμενο πρώτων παραγόντων) : Έχοντας δεδομένο το module `logic_functions` να φτιαχτεί πρόγραμμα , το οποίο θα αποτελείται από :

- ▶ Μία `main()` που θα ζητάει από τον χρήστη τον ακέραιο $n \geq 2$ και θα χειρίζεται τις υπόλοιπες συναρτήσεις για να εκτυπώνει τη λίστα των διαιρετών, τη λίστα των πρώτων διαιρετών και την ανάλυση σε γινόμενο πρώτων παραγόντων του n .
- ▶ Μία `divisors(n)` που θα επιστρέφει τη λίστα των διαιρετών του αριθμού κατά αύξουσα σειρά .
- ▶ Μία `prime_divisors(n)` που θα επιστρέφει τη λίστα των πρώτων διαιρετών του n , κατά αύξουσα σειρά .
- ▶ Μία `prime_analysis(n)` που θα επιστρέφει την ανάλυση σε γινόμενο πρώτων παραγόντων σε συμβολοσειρά της μορφής : $p^{e_p} * q^{e_q} * \dots * z^{e_z}$.

6. **Εφαρμογή** του Προβλήματος 5 (Εύρεση Ε.Κ.Π. και Μ.Κ.Δ. 2 ή περισσότερων αριθμών) : Έπειτα από προσαρμογή στην **έξοδο** της συνάρτησης `prime_analysis` να δημιουργήσετε πρόγραμμα που θα αποτελείται από :

- ◇ Το πρόγραμμα του Προβλήματος 5 , με κατάλληλες αλλαγές στην `main` , ώστε να εκτυπώνει την ανάλυση σε γινόμενο πρώτων παραγόντων κάθε αριθμού και έπειτα τα Ε.Κ.Π. και Μ.Κ.Δ. .
- ◇ Μία συνάρτηση `gcd(l)` , με `gcd` : Μ.Κ.Δ. και l να είναι η λίστα των φυσικών αριθμών .
- ◇ Μία συνάρτηση `lcm(l)` , με `lcm` : ΕΚΠ και l να είναι η λίστα των φυσικών αριθμών .

Σχόλιο: Πόσα στοιχεία θέλουμε να έχει η λίστα l ;

7. Αξιοποιώντας τις συναρτήσεις συμμετρίας που περιέχονται στο module `logic_functions` , να φτιάξετε συναρτήσεις συμμετρίας τριγώνων για κάθε περίπτωση. Το πρόγραμμα θα περιέχει μία συνάρτηση `main()` , η οποία θα ζητάει από τον χρήστη τη δημιουργία των τριγώνων και θα εμφανίζει αν υπάρχει συμμετρία και αν ναι ποια/ες .

Υπόδειξη: Τα τρίγωνα θα πρέπει να γραφτούν ως δισδιάστατες λίστες (λίστα 3 σημείων) .

Σημείωση: Σε οποιαδήποτε από τις παραπάνω ασκήσεις μπορείτε να αξιοποιήσετε το module `logic_functions` από το Φυλλάδιο 2 ή οποιαδήποτε άλλη συνάρτηση, ακόμη και αν δεν έχουν υλοποιηθεί σε κώδικα .