ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ
ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΔΠΜΣ – ΡΟΗ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΑΝΑΠΤΥΞΗ ΚΑΙ ΣΧΕΔΙΑΣΗ ΛΟΓΙΣΜΙΚΟΥ

Η γλώσσα προγραμματισμού C

ΕΡΓΑΣΤΗΡΙΟ 4: Δομές, συναρτήσεις και αρχεία

Μάιος-Ιούνιος 2019

Το σημερινό εργαστήριο έχει ως θέμα τις δομές struct και τις συναρτήσεις της C. Η γλώσσα C ορίζει τη συνάρτηση ως μια ενιαία δομή για το κυρίως πρόγραμμα και όλα τα υποπρογράμματά της. Οι συναρτήσεις – εκτός από τη *main()* – συνήθως δηλώνονται με μορφή πρωτοτύπου στην αρχή του προγράμματος, και ορίζονται πλήρως στη συνέχεια με οποιαδήποτε σειρά. Έχετε ήδη συναντήσει συναρτήσεις, και εδώ θα μελετήσετε πιο προχωρημένα θέματα συναρτήσεων, όπως είναι η αναδρομή. Οι δομές struct της C είναι από τους πιο χρήσιμους τύπους της γλώσσας, και θα μάθετε να τις χρησιμοποιείτε στη στατική τους μορφή. Στο εργαστήριο αυτό θα εξοικειωθείτε καλύτερα με τη χρήση αρχείων για είσοδο και έξοδο.

Τις παρακάτω ασκήσεις θα τις ολοκληρώσετε κατά προτίμηση μέχρι την τελική εξέταση. Η παράδοσή τους θα γίνει όπως έγινε με τα προηγούμενα εργαστήρια. Υπενθυμίζεται η στοίχιση και ο σχολιασμός που πρέπει να έχουν τα προγράμματά σας για να είναι κατανοητά.

Άσκηση 1: «Αραιοί πίνακες»

Ξεκινήστε μια νέα εργασία *lab4* και συνδέστε σε αυτή ένα νέο πρόγραμμα με όνομα αρχείου *ONOMA_lab4.1.c*, βάζοντας όπου ONOMA το ονοματεπώνυμό σας.

Στην άσκηση αυτή θα γράψετε ένα πρόγραμμα, το οποίο θα επεξεργάζεται κάποιους πίνακες, οι οποίοι ονομάζονται *αραιοί*, επειδή οι περισσότερες τιμές τους είναι μηδενικές. Για το λόγο αυτό, αντί να υλοποιούνται με τον κλασικό τρόπο, δηλαδή με πλήρεις γραμμές και στήλες, αποθηκεύονται μόνο οι μη μηδενικές τιμές τους, καθώς και η θέση τους στον πίνακα. Έτσι λοιπόν, κάθε μη μηδενική τιμή ενός αραιού πίνακα θα περιλαμβάνει τη γραμμή, τη στήλη και την τιμή του στοιχείου.

Το πρόγραμμά σας πρέπει να χειρίζεται δισδιάστατους αραιούς πίνακες πραγματικών αριθμών τύπου double, μεγέθους N×N, με M μη μηδενικά στοιχεία, όπου $M << N^2$. Ένας τέτοιος αραιός πίνακας δίνεται ως είσοδος στο πρόγραμμά σας μέσω ενός αρχείου που περιέχει μόνο τα μη μηδενικά στοιχεία του, όπου κάθε στοιχείο δίνεται ως μια τριάδα: ενός ακεραίου που δίνει τη γραμμή, ενός ακεραίου που δίνει τη στήλη και ενός πραγματικού που δίνει την τιμή του στοιχείου. Το πλήθος M των στοιχείων που περιέχονται στο αρχείο δίνεται στην αρχή του αρχείου.

Ορίστε έναν τύπο struct της πιο πάνω μορφής στοιχείου αραιού πίνακα, και στη συνέχεια ορίστε έναν τύπο αραιού πίνακα ως πίνακα τέτοιων δομών struct μεγέθους κατάλληλης σταθεράς – έστω *M_max*. Χρησιμοποιήστε τη δήλωση *typedef* για να δώσετε όνομα στους τύπους σας, έστω *SparseElement* και *SparseMatrix*.

Γράψτε μια συνάρτηση η οποία θα διαβάζει έναν αραιό πίνακα από ένα αρχείο, αποθηκεύοντάς τον σε κατάλληλη μεταβλητή του τύπου *SparseMatrix*. Παρόμοια, γράψτε και μία συνάρτηση, η οποία θα αποθηκεύει έναν αραιό πίνακα από μια μεταβλητή του τύπου *SparseMatrix* σε ένα αρχείο της παραπάνω μορφής. Και στις δύο συναρτήσεις το όνομα του αρχείου να δίνεται από το χρήστη.

Στη συνέχεια, γράψτε δύο συναρτήσεις, οι οποίες θα μετασχηματίζουν έναν αραιό πίνακα σε νέο αραιό πίνακα, όπου όμως τα στοιχεία είναι ταξινομημένα στη μία συνάρτηση με βάση τη γραμμή και τη στήλη στον πίνακα, σε αύξουσα σειρά γραμμής και στήλης, και στην άλλη συνάρτηση με βάση τη στήλη και τη γραμμή, σε αύξουσα σειρά στήλης και γραμμής.

Με βάση τις ταξινομημένες μορφές ενός αραιού πίνακα, μπορούν να υλοποιηθούν εύκολα πράξεις πινάκων. Γράψτε έτσι τρεις συναρτήσεις, οι οποίες θα υπολογίζουν (α) το άθροισμα, (β) τη διαφορά και (γ) το γινόμενο δύο ταξινομημένων αραιών πινάκων. Μην ξεχνάτε ότι οι αραιοί πίνακες είναι πίνακες, όπου απλά η αναπαράστασή τους δεν είναι πλήρης. Για παράδειγμα, αν η πρώτη γραμμή του πρώτου πίνακα περιέχει ως μη μηδενικά τα στοιχεία στις στήλες 2, 7 και 10, και η πρώτη γραμμή του δεύτερου πίνακα περιέχει ως μη μηδενικά τα στοιχεία στις στήλες 3,7 και 12, τότε το άθροισμα των πινάκων θα περιέχει ως μη μηδενικά στοιχεία της πρώτης γραμμής τα στοιχεία στις στήλες 2, 3, 7, 10 και 12, όπου τα στοιχεία στις στήλες 2 και 10 προέρχονται από τον πρώτο πίνακα – ως άθροισμα με τα αντίστοιχα μηδενικά στοιχεία του δεύτερου πίνακα, τα στοιχεία στις στήλες 3 και 12 προέρχονται από το δεύτερο πίνακα – ως άθροισμα με τα αντίστοιχα μηδενικά στοιχεία του πρώτου πίνακα, ενώ το στοιχείο στη στήλη 7 προέρχεται από το άθροισμα των αντίστοιχων στοιχείων των δύο πινάκων.

Γράψτε τώρα το κυρίως πρόγραμμα, όπου ο χρήστης θα επιλέγει μέσα από ένα μενού (α) απλή ταξινόμηση γραμμής-στήλης ενός αραιού πίνακα, (β) απλή ταξινόμηση στήλης-γραμμής ενός αραιού πίνακα, (γ) πρόσθεση δύο αραιών πινάκων, (δ) αφαίρεση δύο αραιών πινάκων και (ε) πολλαπλασιασμό δύο αραιών πινάκων. Για κάθε περίπτωση, το πρόγραμμα θα διαβάζει τον κατάλληλο αριθμό πινάκων, θα εκτελεί τη ζητούμενη λειτουργία, και στο τέλος θα αποθηκεύει σε αρχείο το αποτέλεσμα.

Μεταφράστε και δοκιμάστε το πρόγραμμά σας, χρησιμοποιώντας κατάλληλα αρχεία εισόδου.

Αποθηκεύστε το πρόγραμμά σας και συνεχίστε με την επόμενη άσκηση.

Άσκηση 2: «Πολλαπλασιασμός αριθμών πολύ μεγάλης ακρίβειας»

Ξεκινήστε ένα νέο πρόγραμμα με όνομα αρχείου *ONOMA_lab4.2.c* κατά τα γνωστά. Συνδέστε το αρχείο αυτό με την εργασία *lab4* και αποσυνδέστε το αρχείο του προγράμματος της προηγούμενης άσκησης.

Στην άσκηση αυτή θα γράψετε ένα πρόγραμμα που θα υπολογίζει το γινόμενο δύο αριθμών πολύ μεγάλης ακρίβειας, δηλαδή πολύ μεγάλου πλήθους ψηφίων. Ένας τέτοιος υπολογισμός γίνεται με διαίρεση του πλήθους των ψηφίων ώστε να προκύψουν πολλοί αριθμοί κανονικής ακρίβειας, και στη συνέχεια υπολογισμός του ζητούμενου γινομένου ως άθροισμα πολλών απλούστερων γινομένων. Για τον υπολογισμό του γινομένου θα χρησιμοποιήσετε τον αλγόριθμο του Karatsuba, ο οποίος εκτελεί τον πολλαπλασιασμό σε λιγότερες από το συνηθισμένο πράξεις.

Για να κατανοήσετε τον αλγόριθμο Karatsuba, θεωρήστε δύο αριθμούς X και Y εύρους n ψηφίων σε βάση B. Αν τους χωρίσετε σε δύο μέρη, έτσι ώστε το δεξιό μέρος και των δύο αριθμών να αποτελείται από m ψηφία, τότε οι αριθμοί γράφονται ως:

$$X = X_1 B^m + X_0$$
$$Y = Y_1 B^m + Y_0$$

Το γινόμενο των δύο αριθμών γράφεται επομένως ως:

$$\begin{aligned} X * Y &= (X_1 B^m + X_0) * (Y_1 B^m + Y_0) = \\ &= (X_1 * Y_1) B^{2m} + (X_1 * Y_0 + Y_1 * X_0) B^m + (X_0 * Y_0) \end{aligned}$$

όπου κατά το σύνηθες απαιτούνται τέσσερεις επιμέρους πολλαπλασιασμοί αριθμών μικρότερου εύρους. Ο Karatsuba παρατήρησε ότι αν υπολογιστούν πρώτα οι δύο όροι:

$$Z_2 = X_1 * Y_1$$
$$Z_0 = X_0 * Y_0$$

τότε ο μεσαίος όρος γράφεται ως:

$$Z_1 = X_1 * Y_0 + Y_1 * X_0 = (X_1 + X_0) * (Y_1 + Y_0) - Z_2 - Z_0$$

κι επομένως απαιτούνται συνολικά τρεις αντί τεσσάρων πολλαπλασιασμών αριθμών μικρότερου εύρους. Οι περισσότερες προσθέσεις και αφαιρέσεις που απαιτούνται με τον αλγόριθμο Karatsuba θεωρούνται αμελητέου κόστους σε σύγκριση με το κόστος των πολλαπλασιασμών.

Έστω το ακόλουθο παράδειγμα από την Wikipedia, για τον υπολογισμό του γινομένου των αριθμών 12345 και 6789, με $B = 10$ και $m = 3$:

12345 = **12** · *1000* + **345**
6789 = **6** · *1000* + **789**
z_2 = **12 × 6** = 72
z_0 = **345 × 789** = 272205
z_1 = (**12** + **345**) **×** (**6** + **789**) − z_2 − z_0 = 357 **×** 795 − 72 − 272205 = 11538
12345 **×** 6789 = 72 · *1000*^{2} + 11538 · *1000* + 272205 = **83810205**.

Ο πιο πάνω αλγόριθμος είναι εύκολο να προγραμματιστεί με τη βοήθεια αναδρομής (τεχνική «διαίρει και βασίλευε»), με την παρατήρηση ότι η κάθοδος στις διαδοχικές αναδρομές οδηγεί σε πολλαπλασιασμούς μικρού εύρους, η άνοδος όμως προς τον τερματισμό της αναδρομής οδηγεί σε προσθέσεις και αφαιρέσεις αυξανόμενου εύρους. Έτσι, πριν υλοποιήσετε την αναδρομική συνάρτηση, θα πρέπει να γράψετε μια συνάρτηση πρόσθεσης και αφαίρεσης αριθμών μεγάλου εύρους, η οποία θα πρέπει να δέχεται ως είσοδο δύο συμβολοσειρές, να προσθέτει ή να αφαιρεί τους αριθμούς που παριστάνουν, και να παράγει ως αποτέλεσμα μια νέα συμβολοσειρά με το αποτέλεσμα της πράξης.

Προσέξτε ότι λόγω του μεγάλου εύρους των αριθμών, η πράξη αυτή πρέπει να γίνει από δεξιά προς τα αριστερά, ψηφίο-ψηφίο, ή εναλλακτικά σε ομάδες k ψηφίων. Αν οι δύο αριθμοί βρίσκονται αποθηκευμένοι στους πίνακες συμβολοσειρών K και L, και το αποτέλεσμα θα τοποθετηθεί σε έναν τρίτο πίνακα M, τότε η εναλλακτική υλοποίηση πρόσθεσης και αφαίρεσης μπορεί να ακολουθεί τον εξής αλγόριθμο:

1. Δημιούργησε τρεις δείκτες προς τα δεξιά άκρα των K, L και M.
2. Αρχικοποίησε μια μεταβλητή κρατουμένου C σε 0.

3. Όσο υπάρχουν επόμενα προς τα αριστερά τμήματα k ψηφίων στους δύο αριθμούς:
   3.1. Μετακίνησε τους δείκτες κατά k ψηφία προς τα αριστερά, ώστε να δείχνουν στην αρχή του κάθε τμήματος k ψηφίων.
   3.2. Μετάτρεψε τα τμήματα k ψηφίων των K, L σε ακέραιους αριθμούς.
   3.3. Εκτέλεσε την πράξη πρόσθεσης ή αφαίρεσης για εύρος k ψηφίων, λαμβάνοντας υπόψη το κρατούμενο C, παράγοντας ένα αποτέλεσμα και ένα νέο κρατούμενο.
   3.4. Μετάτρεψε το αποτέλεσμα σε συμβολοσειρά, και αποθήκευσε αυτήν στα αντίστοιχα k ψηφία του M.

Η τιμή του k θα πρέπει να είναι τέτοια, ώστε η πράξη του βήματος 3.3 να υλοποιείται με απλή αριθμητική εντολή. Δεδομένου ότι το σύνηθες μέγεθος ενός ακέραιου είναι μέχρι 2^{31} κατά απόλυτη τιμή, δηλαδή λίγο παραπάνω από 2000000000, μπορείτε να επιλέξετε $k \leq 9$ (για βάση $B = 10$), ώστε να είστε σίγουροι ότι η πράξη μπορεί να γίνει σωστά.

Για τη μετατροπή συμβολοσειράς σε ακέραιο, χρησιμοποιήστε τη συνάρτηση *atoi()*, η οποία με παράμετρο μια συμβολοσειρά επιστρέφει τον ακέραιο που αυτή παριστάνει, ενώ για την αντίστροφη μετατροπή χρησιμοποιήστε τη συνάρτηση *sprintf()*.

Για την περίπτωση της πρόσθεσης, αν η τελευταία επιμέρους πράξη οδηγήσει σε μη μηδενικό κρατούμενο εξόδου, θα πρέπει αυτό να τοποθετηθεί στην επόμενη θέση του πίνακα M. Για την περίπτωση της αφαίρεσης, αντίστοιχα, ένα μη μηδενικό κρατούμενο εξόδου, θα σημαίνει ότι το αποτέλεσμα είναι αρνητικό. Επειδή όμως στον αλγόριθμο του Karatsuba οι αφαιρέσεις δεν δίνουν ποτέ αρνητικό αποτέλεσμα, δεν θα προκύψει ποτέ μη μηδενικό κρατούμενο εξόδου σε πράξη αφαίρεσης.

Προχωρήστε τώρα με την υλοποίηση του αλγορίθμου του Karatsuba για $B = 10$, μέσα από μια αναδρομική συνάρτηση. Η συνάρτηση αυτή θα δέχεται ως παράμετρο έναν αριθμό μεγάλου εύρους ως συμβολοσειρά, θα τον χωρίζει σε δύο μέρη του μισού περίπου εύρους, θα καλεί αναδρομικά τον εαυτό της τρεις φορές για τον υπολογισμό των Z_2, Z_1 και Z_0, και στο τέλος θα εκτελεί τις προσθέσεις και αφαιρέσεις που απαιτούνται για τον υπολογισμό του γινομένου. Η διαίρεση θα συνεχίζεται μέχρι να φτάσετε σε κάποιο ελάχιστο εύρος m. Το μέγιστο τέτοιο εύρος είναι εκείνο για το οποίο οι εμπλεκόμενες πράξεις πολλαπλασιασμού, πρόσθεσης και αφαίρεσης υλοποιούνται με απλή αριθμητική εντολή, και σχετίζεται με το μέγιστο k που προαναφέρθηκε. Έτσι, από τη στιγμή που ένας πολλαπλασιασμός δίνει αποτέλεσμα εύρους μέχρι το άθροισμα εύρους των δύο αριθμών που πολλαπλασιάζονται, το εύρος αυτό είναι το ακέραιο μισό του εύρους του κάθε όρου πρόσθεσης ή αφαίρεσης. Επειδή δε ο υπολογισμός του Z_1 απαιτεί πρόσθεση πριν τον πολλαπλασιασμό, το ελάχιστο αυτό εύρος μειώνεται κατά 1, από τη στιγμή που μια πρόσθεση δίνει αποτέλεσμα εύρους μέχρι το εύρος του μεγαλύτερου αριθμού συν 1. Επομένως το μέγιστο m που μπορούμε να έχουμε για συνήθεις ακέραιους αριθμούς είναι 3, εφόσον το ακέραιο μισό του 9 – το οποίο είναι το μέγιστο k – είναι 4.

Το τελικό πρόγραμμα θα πρέπει να δέχεται ως είσοδο δύο συμβολοσειρές, και να βγάζει ως έξοδο τη συμβολοσειρά του αποτελέσματος. Αν θέλετε, προσθέστε έλεγχο για ορθότητα των συμβολοσειρών εισόδου – δηλαδή να είναι αριθμητικές.

Μεταφράστε, επαληθεύστε και αποθηκεύστε το πρόγραμμά σας.

Μην ξεχάσετε να υποβάλετε τις ασκήσεις σας.