Κ08 Δομές Δεδομένων και Τεχνικές Προγραμματισμού (Τμήμα Φοιτητών/τριών με Άρτιο ΑΜ)

Διδάσκων: Μανόλης Κουμπαράκης

Εαρινό Εξάμηνο 2023-2024

Εργασία 3 (ανακοινώθηκε στις 13 Μαΐου 2022, προθεσμία παράδοσης: 14 Ιουλίου, ώρα 23:59)

16% του συνολικού βαθμού στο μάθημα. Άριστα=310 μονάδες (υπάρχουν και 100 μονάδες μπόνους).

Προσοχή: Πριν διαβάσετε παρακάτω, διαβάστε παρακαλώ προσεκτικά τις οδηγίες υποβολής των ασκήσεων που βρίσκονται στην ιστοσελίδα http://cgi.di.uoa.gr/~k08/homework.html, ειδικά ότι αναφέρεται στο github και τα σχετικά αρχεία.

Απορίες: Αν έχετε απορίες σχετικά με την εργασία, ρωτήστε στο piazza και όχι στέλνοντας e-mail στον διδάσκοντα. Τέτοια e-mails ΔΕΝ θα λαμβάνουν απάντηση.

Κώδικας: Ο κώδικας που παρουσιάσαμε στις διαλέξεις του μαθήματος (Ενότητες 16-19) βρίσκεται στο παρακάτω private repository του classroom του μαθήματος: https://github.com/artioi-k08/2024-ergasia3.

Για να συνδεθείτε σε αυτό το repository, θα πρέπει να χρησιμοποιήσετε το link https://classroom.github.com/a/NuW8LdBj. Μπορείτε να χρησιμοποιήσετε αυτόν τον κώδικα στην εργασία αυτή, όμως θα πρέπει να το γράψετε ξεκάθαρα στα σχόλια του προγράμματός σας.

Όταν συνδεθείτε θα μπορείτε να δείτε το προσωπικό σας repository https://github.com/artioi-k08/2024-ergasia3-<github-your-username> στο οποίο θα δουλέψετε για την Εργασία 3.

Μετά τη σύνδεσή σας, στο προσωπικό σας repository, θα βρείτε τον κώδικα που καλύπτει την Ενότητα 16 του μαθήματος οργανωμένο σε κατάλληλο φάκελο. Θα βρείτε επίσης και ένα φάκελο solutions-ergasia3 με υπο-φακέλους question1, question2, …, question8, question9 στους οποίους θα πρέπει να γράψετε τον

κώδικα και τις απαντήσεις σας για τα 9 θέματα της εργασίας που θα βρείτε παρακάτω. Παρακαλώ τηρείστε ευλαβικά αυτήν την οργάνωση αλλιώς θα χάσετε 20% του συνολικού βαθμού κατά την βαθμολόγηση.

Κύριο πρόγραμμα: Σε όλες τις παρακάτω προγραμματιστικές ασκήσεις θα πρέπει να υλοποιήσετε και ένα κύριο πρόγραμμα (συνάρτηση `main`) το οποίο θα διαβάζει τα δεδομένα εισόδου, θα επιδεικνύει τη λειτουργικότητα της συνάρτησής σας για κατάλληλα επιλεγμένες εισόδους, και θα πείθει τον βαθμολογητή ώστε να σας βαθμολογήσει με τον υψηλότερο δυνατό βαθμό.

1. Θεωρήστε τον κώδικα της Ενότητας 16 για κατευθυνόμενους γράφους με χρήση λιστών γειτνίασης. Στην άσκηση αυτή θα οργανώσουμε και θα εμπλουτίσουμε τον κώδικα αυτό ώστε να ορίζει και να υλοποιεί ένα αφαιρετικό τύπο δεδομένων `DirectedGraph` με τις εξής λειτουργίες που υλοποιούνται από κατάλληλες συναρτήσεις της C:
    a. `Initialize`. Η συνάρτηση αυτή αρχικοποιεί το γράφο που παίρνει σαν όρισμα.
    b. `InsertEdge`. Η συνάρτηση αυτή εισάγει μια ακμή σε ένα γράφο. Η ακμή και ο γράφος είναι ορίσματα της συνάρτησης.
    c. `ShowGraph`. Η συνάρτηση αυτή εκτυπώνει ένα γράφο που δίνεται σαν όρισμα χρησιμοποιώντας την αναπαράσταση των λιστών γειτνίασης όπως κάνουμε στις διαφάνειες της Ενότητας 16 (π.χ., στη διαφάνεια 68).
    d. `DepthFirst`. Η συνάρτηση αυτή κάνει μια πρώτα κατά βάθος διάσχιση του γράφου και εκτυπώνει τις κορυφές που επισκέφθηκε. Επιπλέον εκτυπώνει όλες τις ακμές που διασχίζει και τις ταξινομεί σε ακμές δένδρου, ακμές οπισθοχώρησης, ακμές προώθησης και εγκάρσιες ακμές. Η υλοποίηση αυτής της συνάρτησης θα πρέπει να επεκτείνει τον σχετικό κώδικα από τις διαφάνειες της Ενότητας 16.
    e. `BreadthTopSort`. Η συνάρτηση αυτή υπολογίζει μια τοπολογική ταξινόμηση του γράφου και την εκτυπώνει. Η υλοποίηση της συνάρτησης αυτής θα χρησιμοποιεί τον σχετικό κώδικα από τις διαφάνειες της Ενότητας 16.

f. `GraphReverse`. Η συνάρτηση αυτή παίρνει σαν όρισμα ένα γράφο που παριστάνεται με λίστες γειτνίασης και επιστρέφει ένα νέο γράφο που έχει τις ίδιες κορυφές με τον πρώτο αλλά η κατεύθυνση κάθε ακμής στο νέο γράφο έχει αντιστραφεί.
g. `StrongComponents`. Η λειτουργία αυτή υπολογίζει τις ισχυρά συνεκτικές συνιστώσες του γράφου χρησιμοποιώντας τον αλγόριθμο του Kosaraju. Προτείνουμε να χρησιμοποιήσετε ένα πίνακα `sc` για να κωδικοποιείτε σε ποια συνεκτική συνιστώσα ανήκει κάθε κορυφή ως εξής: αν `sc[v]=i` τότε η κορυφή `v` ανήκει στην συνιστώσα `i`.

Εκτός από τις παραπάνω συναρτήσεις, θα πρέπει να γράψετε και μια συνάρτηση `main` η οποία θα διαβάζει ένα γράφο από την είσοδο και θα εκτελεί τις διάφορες λειτουργίες που περιγράψαμε. Μπορείτε να υποθέσετε ότι ο γράφος δίνεται σε ένα αρχείο εισόδου το οποίο περιέχει στην πρώτη γραμμή του τον αριθμό κόμβων του γράφου, και σε κάθε επόμενη γραμμή την αναπαράσταση μιας ακμής (x,y) με την απλούστερη μορφή `x-y`.

Το πρόγραμμα που θα παραδώσετε θα πρέπει να έχει οργανωθεί σαν ένα module της C που υλοποιεί τον αφηρημένο τύπο δεδομένων. Θα πρέπει να υπάρχει και το αντίστοιχο makefile.

(10+10+10+20+5+10+50+20=135 μονάδες)

2. Δώστε ένα παράδειγμα κατευθυνόμενου γράφου και 2 διαφορετικά δένδρα επικάλυψης του που προκύπτουν από 2 διαφορετικές εκτελέσεις του αλγόριθμου DFS. Εξηγήστε ποιες είναι οι ακμές δένδρου, οπισθοχώρησης, προώθησης και εγκάρσιες στα δένδρα αυτά. Το παράδειγμα σας πρέπει να είναι το μικρότερο δυνατό.

(5 μονάδες)

3. Δίδεται ένας άκυκλος κατευθυνόμενος γράφος με θετικά βάρη στις ακμές. Να υλοποιήσετε ένα αλγόριθμο εύρεσης συντομότερων μονοπατιών από μια αφετηρία προς όλους τους άλλους κόμβους του γράφου. Να γράψετε και μια `main` η οποία θα επιδεικνύει τη συμπεριφορά αυτής της συνάρτησης. Υπόδειξη: Χρησιμοποιείστε τον αλγόριθμο τοπολογικής ταξινόμησης.

(30 μονάδες)

4. Έστω $G = (V, E, W)$ συνεκτικός, μη κατευθυνόμενος και πλήρης γράφος όπου $W: E \to \mathbb{R}$. Θεωρούμε $C_1, C_2, \dots, C_k \subseteq V$ με $C_i \neq \emptyset$ για όλα τα i. Τα σύνολα κορυφών C_i είναι ξένα μεταξύ τους ανά δύο. Να σχεδιάσετε και να υλοποιήσετε πολυωνυμικό αλγόριθμο (δίνοντας και την ακριβή χρονική πολυπλοκότητα χειρίστης περίπτωσης) ο οποίος να βρίσκει ένα μονοπάτι ελάχιστου μήκους k κορυφών της μορφής $c_1 \to c_2 \to \cdots \to c\_k$ όπου $c_i \in C_i$ για κάθε i (δηλαδή το πρώτο στοιχείο να ανήκει στο C_1, το δεύτερο στο C_2 κλπ.). Να γράψετε και μια `main` η οποία να επιδεικνύει τη συμπεριφορά αυτού του αλγόριθμου. Υπόδειξη: Θα χρειαστεί να φτιάξετε ένα καινούριο γράφο και να χρησιμοποιήσετε τον αλγόριθμο του θέματος 3.

(30 μονάδες)

5. Θεωρήστε τον κώδικα της Ενότητας 16 για μη κατευθυνόμενους γράφους με χρήση λιστών γειτνίασης. Στην άσκηση αυτή θα οργανώσουμε και θα εμπλουτίσουμε τον κώδικα αυτό ώστε να ορίζει και να υλοποιεί ένα αφαιρετικό τύπο δεδομένων `UndirectedGraph` με τις εξής λειτουργίες που υλοποιούνται από κατάλληλες συναρτήσεις:
    a. `Initialize`, `InsertEdge` και `ShowGraph` με την ίδια λειτουργικότητα του Θέματος 1.
    b. `SimplePathCheck`. Η συνάρτηση αυτή παίρνει σαν είσοδο ένα μη κατευθυνόμενο γράφο και δύο κορυφές διαφορετικές μεταξύ τους και επιστρέφει ένα απλό μονοπάτι που συνδέει αυτές τις κορυφές ή μας πληροφορεί ότι τέτοιο μονοπάτι δεν υπάρχει. Ένα μονοπάτι λέγεται **απλό** εάν όλες οι κορυφές από τις οποίες αποτελείται είναι διαφορετικές μεταξύ τους.

Γράψτε μια κατάλληλη `main` για την επίδειξη των δυνατοτήτων του προγράμματος σας. Ο γράφος θα δίνεται σε ένα αρχείο χρησιμοποιώντας τις συμβάσεις του προηγούμενου ερωτήματος. Το πρόγραμμα που θα παραδώσετε θα πρέπει να είναι οργανωμένο όπως και το πρόγραμμα του προηγούμενου ερωτήματος.

(10+20=30 μονάδες)

6. Δώστε ένα παράδειγμα μη κατευθυνόμενου γράφου και 2 διαφορετικά δένδρα επικάλυψης του που προκύπτουν από 2 διαφορετικές εκτελέσεις του αλγόριθμου DFS. Εξηγήστε ποιες είναι οι ακμές δένδρου και ποιες οι ακμές οπισθοχώρησης στα δένδρα αυτά. Το παράδειγμα σας πρέπει να είναι το μικρότερο δυνατό.

(5 μονάδες)

7. Στην άσκηση αυτή θα ορίσουμε τον αφαιρετικό τύπο δεδομένων `WeightedUndirectedGraph` με τις εξής λειτουργίες που υλοποιούνται από κατάλληλες συναρτήσεις:
   a. `Initialize`, `InsertEdge` και `ShowGraph` με αντίστοιχη λειτουργικότητα με τις συναρτήσεις του προηγούμενου ερωτήματος.
   b. `MinimumSpanningTree`. Η συνάρτηση αυτή υλοποιεί τον αλγόριθμο των Prim-Jarnik για τον υπολογισμό του ελάχιστου δέντρου επικάλυψης ενός δοσμένου μη κατευθυνόμενου γράφου.

   Εκτός από τις παραπάνω συναρτήσεις, θα πρέπει να γράψετε και μια συνάρτηση `main` η οποία θα διαβάζει ένα μη κατευθυνόμενο γράφο με βάρη από την είσοδο και θα εκτελεί τις διάφορες λειτουργίες που περιγράψαμε. Μπορείτε να υποθέσετε ότι ο γράφος δίνεται σε ένα αρχείο εισόδου το οποίο περιέχει στην πρώτη γραμμή του τον αριθμό κόμβων του γράφου, και σε κάθε επόμενη γραμμή την αναπαράσταση μιας ακμής (x, y, w) (όπου x και y είναι κορυφές και w είναι το βάρος της ακμής (x, y)) με την απλούστερη μορφή `x-y-w`.

   Το πρόγραμμα που θα παραδώσετε θα πρέπει να έχει οργανωθεί σαν ένα module της C που υλοποιεί τον αφηρημένο τύπο δεδομένων. Θα πρέπει να υπάρχει και το αντίστοιχο makefile.

(10+50=60 μονάδες)

8. Να υπολογίσετε την υπολογιστική πολυπλοκότητα χειρίστης περίπτωσης του αλγόριθμου Prim-Jarnik που υλοποιήσατε στο προηγούμενο ερώτημα (δηλ. θα

υπολογίσετε $O(\dots)$ με παραμέτρους e τον αριθμό των ακμών και n τον αριθμό των κορυφών του γράφου).

(15 μονάδες)

9. Να υλοποιήσετε την τεχνική του γραμμικού κατακερματισμού (linear hashing) η οποία παρουσιάζεται στην ιστοσελίδα https://en.wikipedia.org/wiki/Linear_hashing και στα σχετικά άρθρα που αναφέρονται σε αυτήν. Θα πρέπει να υλοποιηθεί μια συνάρτηση η οποία να εισάγει στοιχεία στον πίνακα κατακερματισμού, και μια η οποία αναζητεί ένα στοιχείο με δοσμένο κλειδί. Δεν χρειάζεται να υλοποιήσετε συνάρτηση για τη διαγραφή στοιχείων.

(100 μονάδες μπόνους)

Καλή Επιτυχία!!!