ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ
ΣΧΟΛΗ ΗΛΕΚΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΔΟΜΕΣ ΔΕΔΟΜΕΝΩΝ ΚΑΙ ΑΡΧΕΙΩΝ

2η άσκηση

Ημερομηνία παράδοσης: 25 Απριλίου 2021
Η άσκηση είναι ατομική

Σε ένα απλό Δυαδικό Δένδρο Έρευνας (ΔΔΕ) η διάσχιση (traversal) «σε τάξη» (inorder traversal) επισκέπτεται όλα τα κλειδιά σε αύξουσα σειρά. Μια ιδιαίτερη εφαρμογή της διάσχισης inorder είναι η επεξεργασία ερωτημάτων «εύρους τιμών» (range queries)[1] του τύπου «βρες όλα τα κλειδιά με τιμές στο διάστημα [K1, K2]». Μία διάσχιση μπορεί να γίνει με ένα αναδρομικό αλγόριθμο ή με ένα επαναληπτικό αλγόριθμο και την χρήση «στοίβας» (stack).

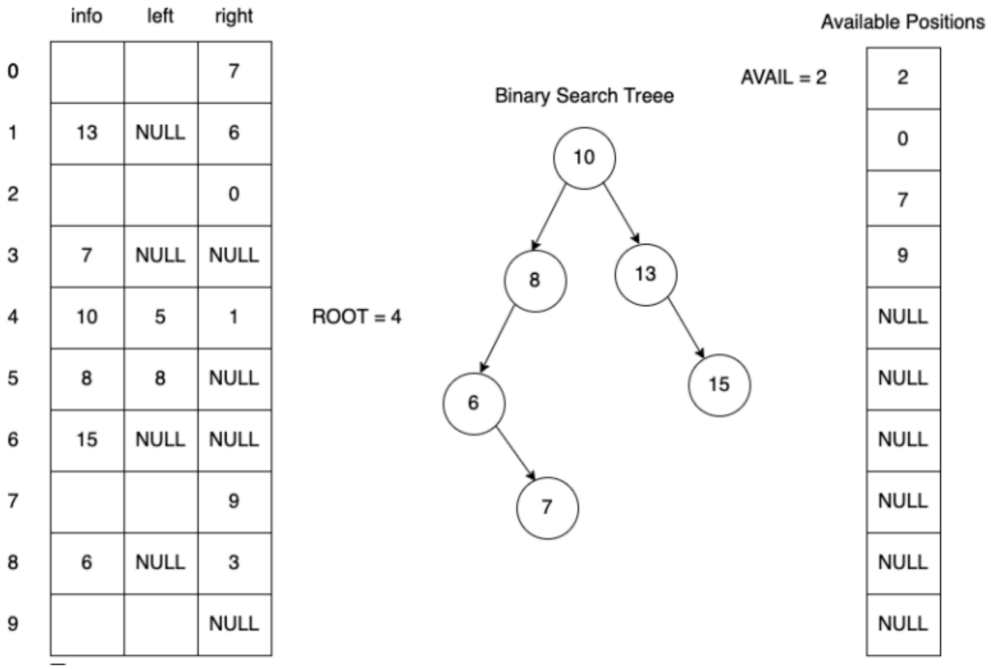
Α Δυαδικό Δένδρο Έρευνας (2 μονάδες)

Κατασκευάστε την κλάση «Δυαδικό Δένδρο Έρευνας» (ΔΔΕ) με χρήση πεδίου αριθμών (array) μεγέθους Nx3. Εκτός από τις βασικές πράξεις info(tree_pointer), left(tree_pointer), right(tree_pointer) που ισχύουν γενικά για ΔΔΕ, η κλάση υποστηρίζει τις πράξεις:

- Εισαγωγή τυχαίου κλειδιού
- Αναζήτηση τυχαίου κλειδιού
- Αναζήτηση εύρους τιμών[2].

Όπως συμβαίνει και στην περίπτωση συνδεδεμένης λίστας, η υλοποίηση μπορεί να γίνει είτε με δυναμική παραχώρηση μνήμης είτε με πεδίο (array). Για την υλοποίηση του ΔΔΕ χρησιμοποιείστε ένα πεδίο Nx3 ακεραίων αριθμών (N είναι ο αριθμός των στοιχείων). Κάθε θέση του πίνακα παριστάνει έναν κόμβο του δένδρου και σε κάθε μία από τις 3 στήλες αποθηκεύονται τα πεδία info, left, right του κόμβου. Ένα πεδίο μεγέθους Nx3 στοιχείων μπορεί να παραστήσει ένα ΔΔΕ με N κλειδιά το πολύ.

[1] https://www.geeksforgeeks.org/print-bst-keys-in-the-given-range/



Το παραπάνω σχήμα (αριστερά) δείχνει της υλοποίηση του ΔΔΕ (στο κέντρο). Η ρίζα του δένδρου βρίσκεται στην θέση 4 (ROOT= 4). Η υλοποίηση είναι αντίστοιχη με την υλοποίηση συνδεδεμένης λίστας με array[3]. Το πεδίο left έχει τιμή που δηλώνει την θέση του αριστερού υποδένδρου. Το πεδίο right έχει διπλό ρόλο: (α) δηλώνει την θέση του δεξιού υποδένδρου και (β) χρησιμοποιείται επίσης για να υλοποιήσει την στοίβα (stack) με τις διαθέσιμες θέσεις του array. Στο σχήμα (δεξιά), η θέση AVAIL=2 δηλώνει ότι θέση 2 είναι η επόμενη διαθέσιμη θέση που θα χρησιμοποιηθεί στην επόμενη εισαγωγή κόμβου. Αν γίνει διαγραφή κόμβου, τότε αυτή θα προστεθεί στην κορυφή της στοίβας (και η θέση AVAIL θα πάρει τιμή την θέση του κόμβου που διαγράφτηκε). Επομένως, για να υλοποιήσετε τις μεθόδους εισαγωγής και διαγραφής στοιχείων από τον ΔΔΕ θα χρειαστείτε πιο πριν να έχετε υλοποιήσει της μεθόδους getnode(tree_pointer) και free_node(tree_pointer) στην στοίβα των ελεύθερων θέσεων του array[4]. Στο παραπάνω σχήμα (δεξιά) φαίνεται η μορφή της στοίβας που υλοποιεί η στήλη right του array. Προσέξετε ότι δεν είναι ανάγκη να φτιάξετε την στοίβα με ξεχωριστή δομή δεδομένων έξω από το array (η στοίβα στα δεξιά του σχήματος υπάρχει ήδη στο πεδίο right).

Νηματοειδές Δυαδικό Δένδρο Έρευνας (6 μονάδες)

Το «νηματοειδές ΔΔΕ» (Threaded BST)[5] μπορεί να πετύχει διάσχιση με επανάληψη χωρίς στοίβα (χωρίς αναδρομή). Αυτό που κάνει ένα ΔΔΕ να είναι νηματοειδές είναι η διαφορετική χρήση των αριστερών και δεξιών δεικτών σε κάθε κόμβο. Η μόνη διαφορά είναι σε δείκτες που δεν έχουν αριστερό ή δεξιό παιδί (υποδένδρο). Πιο συγκεκριμένα, αυτοί οι δείκτες μπορούν να παίρνουν τιμές που δεν είναι NULL ακόμα και στην περίπτωση που δεν υπάρχει αριστερό ή δεξιό υποδένδρο. Αν ένας κόμβος δεν έχει αριστερό υποδένδρο, αντί να έχει τιμή NULL, τον κάνουμε να δείχνει στον κόμβο με την αμέσως μικρότερη τιμή κλειδιού. Ομοίως, αν ένας δείκτης δεν έχει δεξιό υποδένδρο, αντί να έχει τιμή NULL, το κάνουμε να δείχνει στον κόμβο με την αμέσως μεγαλύτερη τιμή κλειδιού. Δεδομένου ότι ο δεξιός δείκτης χρησιμοποιείται για δύο σκοπούς, προσθέτουμε

[3] Συμβουλευτείτε την υλοποίηση της σελ. 16 της διάλεξης για συνδεδεμένες λίστες.
[4] Η υλοποίηση της στοίβας γίνεται όπως και στην περίπτωση της συνδεδεμένης λίστας.
[5] https://www.geeksforgeeks.org/threaded-binary-tree/

στον κόμβο την δυαδική (boolean) μεταβλητή rightThread που χρησιμοποιείται για να δηλώσει εάν ο δεξιός δείκτης δείχνει στο δεξί παιδί ή στο κόμβο με την αμέσως μεγαλύτερη τιμή κλειδιού. Ομοίως, μπορούμε να προσθέσουμε την δυαδική μεταβλητή leftThread για να δηλώσει αν ο αριστερός δείκτης δείχνει στο αριστερό παιδί ή στον κόμβο με την αμέσως μικρότερη τιμή κλειδιού.

Με αυτόν τον τρόπο, μια inorder διάσχιση δένδρου, ξεκινά από τον πιο αριστερό κόμβο (που είναι ο κόμβος με το μικρότερο κλειδί) και καταλήγει σε μια διαδοχική διάσχιση λίστας συνδεδεμένων κόμβων. Αντίστοιχα, σε ένα ερώτημα εύρους τιμών [K1, K2] όπου K1 < K2, βρίσκουμε πρώτα το κλειδί K1 (ή το αμέσως μεγαλύτερό του αν δεν υπάρχει) και συνεχίζουμε με διάσχιση μέχρι να βρεθεί το κλειδί K2 (ή το αμέσως μικρότερο του αν δεν υπάρχει). Με παρόμοιο τρόπο θα μπορούσατε να απαντήσετε σε ερωτήματα εύρους τιμών [K2, K1].

Τροποποιείστε την υλοποίηση του ερωτήματος Α ώστε το ΔΔΕ να γίνει Νηματοειδές ΔΔΕ και υλοποιήστε όλες τις πράξεις όπως και στο ερώτημα Α. Οι τιμές των δεικτών που είναι NULL θα αλλάξουν (εφόσον υπάρχει μικρότερο ή μεγαλύτερο κλειδί) και θα χρειαστεί να προσθέσετε στον πίνακα τα πεδία leftThread και leftThread (δηλαδή ο πίνακας θα γίνει Nx5). Στο Web (στην διεύθυνση που δόθηκε) θα βρείτε την υλοποίηση με δυναμική παραχώρηση μνήμης (δηλαδή με χρήση της εντολής new). Θα πρέπει να την τροποποιήσετε ώστε να λειτουργεί με χρήση του πίνακα. Εκτός των άλλων θα βρείτε και αλγόριθμο που μετατρέπει ένα απλό ΔΔΕ σε Νηματοειδές (στην άσκηση δεν ζητείται αυτή η υλοποίηση).

Απόδοση και Τεκμηρίωση Αποτελεσμάτων (2 μονάδες)

Χρησιμοποιήστε ένα πεδίο και κάντε εισαγωγή n =10^5 κλειδιών με τιμές 1 έως 10^6 (δεν έχει σημασία ποια είναι η μέγιστη τιμή κλειδιού αρκεί τα κλειδιά να μην επαναλαμβάνονται) σε τυχαία σειρά (n < N) και με τις δύο υλοποιήσεις και:

1. Μετρήστε το μέσο αριθμό συγκρίσεων ανά εισαγωγή. Δηλαδή για κάθε εισαγωγή νέου στοιχείου μετρήστε τον αριθμό συγκρίσεων και στο τέλος υπολογίστε τον μέσο όρο.
2. Κάντε 100 αναζητήσεις τυχαίων αριθμών κλειδιών με τιμές στο διάστημα 1 έως 10^6 και μετρήστε τον μέσο αριθμό συγκρίσεων ανά αναζήτηση.
3. Κάντε 100 τυχαίες αναζητήσεις εύρους τιμών και μετρήστε τον μέσο αριθμό συγκρίσεων ανά αναζήτηση. Το εύρος ορίζεται με την αρχική τυχαία τιμή + K (όπου το K παίρνει τιμές 100 και 1000)

Κατασκευάστε ένα ταξινομημένο πίνακα n στοιχείων με τιμές στο διάστημα 1 έως 10^6 και πάρτε τις ίδιες μετρήσεις πίνακα εφαρμόζοντας δυαδική αναζήτηση (binary search) εκτός από την περίπτωση 1 (μέσο αριθμό συγκρίσεων ανά εισαγωγή). Συμπληρώστε τις τιμές στο παρακάτω πίνακα:

Μέθοδος	**Μέσος αριθμός συγκρίσεων / εισαγωγή**	**Μέσος αριθμός συγκρίσεων / τυχαία αναζήτηση**	**Μέσος αριθμός συγκρίσεων / αναζήτηση εύρους (K=100)**	**Μέσος αριθμός συγκρίσεων / αναζήτηση εύρους (K=1000)**
ΔΔΕ Α				
Νηματοειδές ΔΔΕ Β				
Ταξινομημένο πεδίο				

Τεκμηρίωση των Αποτελεσμάτων (2 μονάδες)

Σχολιάστε (ξεχωριστό αρχείο κειμένου) την απόδοση όλων των μεθόδων και προσπαθήστε να δικαιολογήστε την απόδοση κάθε μεθόδου. Προτείνετε πιθανές βελτιώσεις (δεν ζητείται να τις υλοποιήσετε). **Δώστε ιδιαίτερη βαρύτητα στην ερμηνεία των αποτελεσμάτων και μην δείξετε μόνο τις τιμές.**

Συστάσεις:

- Μην προχωρήσετε στην υλοποίηση πριν καταλάβετε καλά την υλοποίηση συνδεδεμένης λίστας με array που συζητήθηκε στο μάθημα.
- Δώστε προσοχή στην αρχικοποίηση της λίστας διαθέσιμων θέσεων. Θα πρέπει όλες οι θέσεις του πεδίου να βρίσκονται στην στοίβα των διαθέσιμων θέσεων. Η αρχικοποίηση μπορεί να γίνει με τον ίδιο ακριβώς τρόπο όπως στην συνδεδεμένη λίστα. Με τον ίδιο τρόπο επίσης υλοποιούνται οι μέθοδοι getnode και freenode (αντί για next χρησιμοποιείστε τον δείκτη right).
- Το πρόγραμμά σας πρέπει να δουλέψει για την είσοδο κλειδιών σε αρχείο που θα δώσουμε εμείς κατά την διόρθωση της άσκησης.
- Η άσκηση πρέπει να τυπώνει στην οθόνη τις τιμές του παραπάνω πίνακα για οποιοδήποτε αρχείο κλειδιών που θα δίνεται στην είσοδο.

Παραδοτέα: Ένα συμπιεσμένο zip αρχείο που περιέχει:

- Μία έκθεση 2 σελίδες το πολύ με τα αποτελέσματα που σας ζητούνται δηλαδή. **Δώστε ιδιαίτερη βαρύτητα στην ερμηνεία των αποτελεσμάτων** και μην δείξετε μόνο τις τιμές.
- Μία έκθεση που περιγράφει σε 1-2 σελίδες πώς φτιάχτηκε ο κώδικας (δηλ. για κάθε ερώτημα ποια είναι η γενική ιδέα της λύσης σε 3-4 προτάσεις), υπάρχουν σαφείς οδηγίες μετάφρασης από compiler και εκτέλεσης, τι λάθη έχει (αν έχει, περιπτώσεις που δεν δουλεύει το πρόγραμμα, ή περιπτώσεις που κάνει περισσότερα από όσα σας ζητεί η άσκηση, τι χρησιμοποιήσατε από έτοιμα προγράμματα ή πηγές πληροφόρησης. Υποδείξετε ακόμα και πηγές στο WWW όπως Wikipedia ή Stackoverflow (πλήρης διεύθυνση σχετικών σελίδων).
- Ο κώδικας περιέχει συνοπτικά σχόλια που εξηγούν την υλοποίηση. Προσθέστε σχόλια σε μορφή javadoc στην αρχή της κάθε κλάσης και κάθε μεθόδου. Επίσης javadoc σχόλια πριν από κάθε member variable των κλάσεων. Και όπου απαιτείται μέσα στον κώδικά σας.
- Εκτός των παραπάνω, οι ασκήσεις βαθμολογούνται με άριστα εφόσον:
  - Το zip είναι πλήρες.
  - Οι κώδικες περνούν από compiler και εκτελούνται κανονικά και σωστά σε windows ή Linux περιβάλλον (Προσοχή: Θα πρέπει να κάνετε χρήση σχετικών directory paths και όχι απόλυτων, τα οποία ισχύουν μόνο για τους υπολογιστές σας).
  - Ο κώδικάς σας δουλεύει για οποιαδήποτε τιμή παραμέτρων και αρχείο εισόδου με N αριθμούς (μορφή αρχείου text ή binary).
- Οι ασκήσεις υποβάλλονται ηλεκτρονικά στον ιστοχώρο του μαθήματος και όχι με e-mail.
- Οι αντιγραφές (ακόμη και μέρους της υλοποίησης) μηδενίζονται.