ΔΕΥΤΕΡΗ ΑΣΚΗΣΗ

Γκαραγκάνη Αλεξάνδρα,
ΑΜ:2019030020

Α Δυαδικό Δένδρο Έρευνας

- **public ArrayBST(int N)**: Constructor που αρχικοποιεί τις μεταβλητές του δέντρου και έχει ως όρισμα το μέγεθός του.
- **private int getNode()**: Μέθοδος που επιστρέφει την επόμενη ελεύθερη θέση της στοίβας για εισαγωγή. Επιστρέφει την θέση αυτή.
- **public void insert(int key)**: Μέθοδος που έχει γίνει Overwrite από το Interface BinarySearchTree , έχει ως όρισμα το κλειδί που θα εισαχθεί και αναθέτει στο root την τιμή που επιστρέφει η συνάρτηση insertKey.
- **public int insertKey(int root, int key)**: Η μέθοδος αυτή εξετάζει όλα τα πιθανά σενάρια για εισαγωγή. Αν το array είναι άδειο τοποθετεί το κλειδί στο root του, αλλιώς κάνει traverse .Αφού βρίσκει το σημείο που πρέπει να τοποθετηθεί το κλειδί , ενώ έχει πρώτα βρει την επόμενη ελεύθερη θέση απ΄ την στοίβα, κάνει την εισαγωγή.
- **public int find(int root, int key)**: Μέθοδος που έχει γίνει Overwrite από το Interface BinarySearchTree , έχει ως όρισμα το κλειδί που θα αναζητηθεί μέσα στο δέντρο. Αν το κλειδί δεν είναι στη ρίζα ή το δέντρο είναι άδειο κάνει traverse για να το βρεί.
- **public void printRange(int root,int low, int high)**: Μέθοδος που έχει γίνει Overwrite από το Interface BinarySearchTree , έχει ως όρισμα την ρίζα του δέντρου , το κάτω και το πάνω άκρο του range. Αυτή ανάλογα με το που βρίσκεται το κλειδί προς επεξεργασία μέσω της αναδρομής κάνει traverse στο δέντρο και εκτυπώνει προαιρετικά τα κλειδιά που βρίσκει ότι είναι εντός του range.
- **public void searchRandom(int searches)**: Μέθοδος που εκτελεί 100 αναζητήσεις τυχαίων κλειδιών ,καλώντας την μέθοδο find. Εκτυπώνει τον μέσο αριθμό συγκρίσεων ανά τυχαία αναζήτηση.
- **public void searchingRange(int range, int searches)**: Μέθοδος που εκτελεί 100/1000 αναζητήσεις τυχαίου εύρους(που δημιουργείται μέσα σε αυτή) και εκτυπώνει τον μέσο αριθμό συγκρίσεων ανά τυχαία αναζήτηση εύρους.
- **public void insertInfo(int[] info)**: Μέθοδος που εισάγει 100000 τυχαίους ακεραίους μέσα στο δέντρο και εκτυπώνει τον μέσο αριθμό συγκρίσεων ανά εισαγωγή.

Νηματοειδές Δυαδικό Δένδρο Έρευνας

- **public ThreadedBST (int size)**: Constructor που αρχικοποιεί τις μεταβλητές του δέντρου και έχει ως όρισμα το μέγεθός του.
- **private int getNode()**: Ίδια ακριβώς διαδικασία με το Δυαδικό Δέντρο Έρευνας.

- **private int findNext(int current)**: Μέθοδος η οποία τσεκάροντας τα boolean fields του array καταφέρνει να κάνει in order διάσχιση και επιστρέφει το **αμέσως** μεγαλύτερο κλειδί του current.
- **private int findPrev(int current)**: Μέθοδος η οποία τσεκάροντας τα boolean fields του array καταφέρνει να κάνει in order διάσχιση και επιστρέφει το αμέσως μικρότερο κλειδί του current.
- **public void insert(int key)**: Ίδια ακριβώς διαδικασία με το Δυαδικό Δέντρο Έρευνας.
- **public int insertKey(int root, int key)**: Η μέθοδος αυτή εξετάζει όλα τα πιθανά σενάρια για εισαγωγή. Αν το array είναι άδειο τοποθετεί το κλειδί στο root του, αλλιώς κάνει traverse .Αφού βρίσκει το σημείο που πρέπει να τοποθετηθεί το κλειδί , ενώ έχει πρώτα βρει την επόμενη ελεύθερη θέση απ΄ την στοίβα, κάνει την εισαγωγή.Η διαφορά με την υλοποίηση στο Δυαδικό Δέντρο Έρευνας, είναι ότι χρησιμοποιήθηκαν στις συγκρίσεις τα boolean fields του array για ταχύτερη διαδικασία.
- **public int find(int root, int key)**: Μέθοδος που έχει γίνει Overwrite από το Interface BinarySearchTree , έχει ως όρισμα το κλειδί που θα αναζητηθεί μέσα στο δέντρο. Αν το κλειδί δεν είναι στη ρίζα ή το δέντρο είναι άδειο κάνει traverse για να το βρεί. Η διαφορά με την υλοποίηση στο Δυαδικό Δέντρο Έρευνας, είναι ότι χρησιμοποιήθηκαν στις συγκρίσεις τα boolean fields του array για ταχύτερη διαδικασία.
- **public void printRange(int root,int low, int high)**: Μέθοδος που έχει γίνει Overwrite από το Interface BinarySearchTree , έχει ως ορίσματα την ρίζα του δέντρου , το κάτω και το πάνω άκρο του range. Ανάλογα με το που βρίσκεται το κλειδί προς επεξεργασία μέσω των συναρτήσεων **findNext** και **findPrev ,** αλλά και συγκρίσεων με τα άκρα του range γινεται treaverse στο δέντρο και εκτυπώνει προαιρετικά τα κλειδιά που βρίσκει ότι είναι εντός του range.
- **public void searchRandom(int searches)**: Ίδια ακριβώς διαδικασία με το Δυαδικό Δέντρο Έρευνας.
- **public void searchingRange(int range, int searches)**: Ίδια ακριβώς διαδικασία με το Δυαδικό Δέντρο Έρευνας.
- **public void insertInfo(int[] info)**: Ίδια ακριβώς διαδικασία με το Δυαδικό Δέντρο Έρευνας.

Ταξινομημένο Πεδίο

- **public BinarySearch(int newData[])**: Constructor που πα'ιρνει ως όρισμα ένα integer array.
- **private int binarySearch(int left, int right, int key)**: Μέθοδος που έχει ως ορίσματα τα άκρα του array και το κλειδί προς αναζήτηση. Στην συγκεκριμένη περίπτωση η αναζήτηση του κλειδιού γίνεται μέσω του binarySearch κώδικας που έχει διεξαχθεί στο εργαστήριο. Αν το κλειδι δε βρεθεί επιστρέφει Integer.MIN_VALUE.

- **public void binaryRange(int left,int right,int min,int max)**: Μέθοδος που έχει ως ορίσματα τα άκρα του array και του range. Εκτελώντας έναν κώδικα πολύόμοι με του BinarySearch βρισκουμε και εκτυπώνουμε (πρεραιτικα), όλα τα κλειδιά εντώς του range.
- **public void randomBinarySearch(int searches)**: Μέθοδος που εκτελεί 100 αναζητήσεις τυχαίων κλειδιών ,καλώντας την μέθοδο **binarySearch**. Αν η επισρεφόμενη τιμή είναι διάφορη του Integer.MIN_VALUE και ισούται με το κλειδί εκτυπώνει το κλειδί. Επίσης εκτυπώνει τον μέσο αριθμό συγκρίσεων ανά τυχαία αναζήτηση.
- **public void printRange(int range,int searches)**: Μέθοδος που εκτελεί 100/1000 αναζητήσεις τυχαίου εύρους(που δημιουργείται μέσα σε αυτή) και εκτυπώνει τον μέσο αριθμό συγκρίσεων ανά τυχαία αναζήτηση εύρους.

Links

https://www.geeksforgeeks.org/threaded-binary-tree-insertion/

Print BST keys in the given range - GeeksforGeeks

Threaded Binary Tree - GeeksforGeeks

https://www.geeksforgeeks.org/print-bst-keys-in-given-range-o1-space/

https://www.geeksforgeeks.org/arrays-sort-in-java-with-examples/