

Homework Assignment 5

Trees and Binary Search Trees

Release Date	Due Date
March 26, 2020	April 12, 2020

Objectives

- To understand trees data structures
- To practice programming using binary search trees
- Practice developing high-performance solutions.
- Analyze what are the advantages and the disadvantages of various techniques to manipulate trees.

Problem Specification

In this assignment we will get practice in manipulating trees.

PART 1 (conceptual, no programming):

1. Derive the relationships among height h , total number of nodes n , number of leaves L , and the number of edges m in a fully-complete quad tree T (*a fully-complete tree is a complete tree in which all possible nodes at every level in the tree are present including the last level*). In particular, given h , what is n ? Given n , what is h ? Given m , what is h ? Given m , what is n ? Given L , what is h ? Given h , what is L ? Derive exact expressions first and then express them using big-theta/big-oh notation. Show all your work otherwise no credit.
2. Repeat question 1 if T is a complete quad tree.
3. Describe an implicit representation of a quad-tree T using arrays. Give the declaration of the array that stores the data of the nodes of T . Assume that the data consists of three fields: `ssid`, `name` (*in the format `firstname:lastname`*), `phoneNumber`. Given an arbitrary node v of T at index i , what are the indices of the four children of v and the parent of v ?
4. Let T be a (unbalanced) binary search tree:
 - a. Insert the sequence 65, 70, 60, 72, 87, 40, 35, 90, 75, 63, 68, 69, 61, 20, 25, 28, 37 (of integer keys) into T . Show the tree after each insert operation.
 - b. Show the output of Preorder, Inorder and Postorder traversals of T after last insertion (i.e., after key 37 is inserted).

- c. Delete the sequence 28, 72, 65, 35, 63, 87, 70 from T. Show T after each deletion. Note that to be consistent in deletion, follow the same algorithm when you have to “borrow” a key to fill a hole.

PART 2 (programming):

5. Design, develop and implement an efficient object-oriented application to build an (unbalanced) binary search tree T using an array based implicit representation of T and starting with an empty tree T. The object representing T should support at least the following operations: *insert(mydata x)*, *delete (mydata x)*, *search (mydata x)*, *preorderTraversal()*, *inorderTraversal()*, and *postorderTraversal()*. ‘mydata’ is a record containing following three fields: stuName (a string of at most 15 characters, first and last name separated by a colon ‘:’), courseNumber(an integer), grade (a char). Search key in T is based on stuName. Use recursion to implement the three traversals. Use dynamic allocation of array elements whenever possible (hint: c# and Java users can use arraylist).

One preliminary object-oriented design would be to define at least two classes: one mydata and another called binaryStreeImplicit with appropriate data members and methods, a skeleton is shown below. (Feel free to extend to make it better.)

```

Class Mydata { String stuName; int courseNumber; char grade;
                ... // properties, constructors and methods }
Class BinaryStreeImplicit {
    Mydata[ ] tree; // holds all the nodes of the binary search tree
    int treeSize; // number of nodes in the binary search tree
    int lastIndexUsed; // the tree may not be full,
                        // so specifies the array bound
    ... // properties and constructors
    // some methods that may be helpful
    int root(); //returns the index of the root of the tree or -1 if
tree is empty
    int leftchild(int i); int rightchild(int i); int parent(int i);
        // return the indices of the children and
        // parent of a node at index i
    void inorderTraversal(); void preorderTraversal();
void postorderTraversal();
        //obvious meanings; prints data of nodes one line at a
time; prints only the nodes where actual data exists (i.e., no holes)
    int insert (Mydata x); // inserts x into the tree; returns the
index of the array where inserted, or -1 if unsuccessful
    int delete (Mydata x); // deletes x from the tree – note this
may create “hole” in the array storing the tree nodes but BST is
maintained; returns -1 if x doesn’t exist, otherwise the index where x was
    int search (Mydata x); // returns the index where x exists,
otherwise -1
    ... // add any other private / public methods that may help
manipulation of BST
}

```

[Sample input demo text file](#). Format of the demo file: a number of lines, each line containing an operation to be executed on the BST. For example,

```
Insert: John:Doe, 3310, A
Insert: Jane:Dane, 1120, B
Delete: June:Doe
Delete: John:Doe
Insert: Aj:Gup, 3310, B
Insert: Foo:Done, 2240, C
Inorder
Preorder
Insert: Aj:Gupt, 3310, D
Postorder
Search: John:Doe
Search: Jane:Dane
Inorder
...
```

Read in the name of the input file.

Output execution of the BST operations specified in the input file to the file named “hw5output_yourname.txt” (replace yourname to yours).

Design Requirements

Code Documentation

For this assignment, you must include documentation for your code. This include how to compile and run your program. Also, you **must** give outputs of your program using the supplied input file hw5cs3310Sp2020data.txt. If you don't provide outputs for this input instance, it means your program does not execute properly.

You must also include documentation for your code as generated by Javadoc (and similar tools if using some other language). You should have Javadoc comments for every class, constructor, and method. By default, Javadoc should output html documentation to a subfolder within your project (/dist/javadoc). Make sure this folder is included when you zip your files for submission. You do not need to submit a hard copy of this documentation.

Hint: <http://stackoverflow.com/questions/4468669/how-to-generate-javadoc-html-in-eclipse>

Coding Conventions and Programming Standards

You must adhere to all conventions in the CS 3310 Java coding standard. This includes the use of white spaces for readability and the use of comments to explain the meaning of various methods and attributes. Be sure to follow the conventions for naming files, classes, variables, method parameters and methods. Read the material linked from our class web-pages (in case you can't recall programming styles and conventions from your CS1 and CS2 courses).

Testing

Make sure you test your application with several different values capturing different cases, to make sure it works. That is test your program not just on the inputs we specified but other representative test data that covers your program's logic as well.

Assignment Submission

- Generate a .zip file that contains all your files, including:
 - Signed [Plagiarism Declaration](#)
 - Source code files
 - Include any input or output files
 - Documentation of your code – e.g. using Javadoc if using Java
 - A brief report (in a pdf file) on your observations of comparing theoretical vs empirically observed time complexities. Note this report will include (a) a brief description of problem statement(s), (b) algorithms descriptions (if these are standard, commonly known algorithms, then just mention their names along with customization to your specific solution(s), otherwise give the pseudo-code of your algorithms, (c) theoretically derived complexities of the algorithms used in your code, (d) table(s) of the observed time complexities, and (e) plots comparing theoretical vs. empirical along with your observations (e.g. do theoretical agree with your implementation, why? Why not?).
 - Beginning of the first file (i.e., readme file) should clearly identify you, the class, submission date, and the main goals of the homework / programming assignment. Also **add credit to others if you had to resort to looking up the solution elsewhere**. Finally add one of the sentences: " I give permission to the instructor to share my solution(s) with the class." or " I do NOT give permission to the instructor to share my solution(s) with the class."
- Don't forget to follow the naming convention specified for submitting assignments