

CS 211 Data Structures and Algorithms Lab

Aug -- Nov, 2019

Assignment 2

Total Marks: 10

Due on 31st August

The objective of this assignment is to implement Binary Search Tree (BST).

Command-line argument:

Your program should receive a file (input file) as a command line argument.

Input file

The input file will be a text file where each line will be of any of the following format: insert <number>, inorder, preorder, postorder, search <number>, minimum, maximum, successor <number>, predecessor <number>, delete <number>, where <number> represents any non-negative integer. The input will be given in such a way that, at any point in time, the BST contains only distinct numbers.

The output must be in a file named 'bst.txt'. Every line in the input file must have a corresponding output line in bst.txt. The details are given below.

| Command | Meaning | Output |
|-----------------|--------------------------------------|---|
| insert <number> | Insert <number> to the BST | <number> inserted |
| inorder | Do an inorder traversal of the BST | Sequence of numbers (separated by a white space) obtained by doing inorder traversal / <empty-line> (if BST is empty) |
| preorder | Do a preorder traversal of the BST | Sequence of numbers (separated by a white space) obtained by doing preorder traversal / <empty-line> (if BST is empty) |
| postorder | Do a post-order traversal of the BST | Sequence of numbers (separated by a white space) obtained by doing postorder traversal / <empty-line> (if BST is empty) |
| search <number> | Search <number> in the BST | <number> found / <number> not found |
| minimum | Obtain the minimum number in the BST | <minimum-number> / <empty-line> (if BST is empty) |
| maximum | Obtain the | <maximum-number> / <empty-line> |

| | | |
|----------------------|--|---|
| | maximum number in the BST | (if BST is empty) |
| successor <number> | Obtain the successor of <number> in the BST | <successor> / <number> does not exist / successor of <number> does not exist (if <number> is the maximum number) |
| predecessor <number> | Obtain the predecessor of <number> in the BST | <predecessor> / <number> does not exist / predecessor of <number> does not exist (if <number> is the minimum number) |
| delete <number> | Delete (if exists) <number> from the BST | <number> deleted / <number> does not exist |

The content of an example input file (on the left side) and the corresponding output file (on the right side) are given below.

| | |
|----------------|---------------------------------|
| insert 6 | 6 inserted |
| insert 9 | 9 inserted |
| insert 3 | 3 inserted |
| insert 19 | 19 inserted |
| insert 0 | 0 inserted |
| insert 5 | 5 inserted |
| inorder | 0 3 5 6 9 19 |
| preorder | 6 3 0 5 9 19 |
| search 6 | 6 found |
| search 8 | 8 not found |
| delete 10 | 10 does not exist |
| delete 9 | 9 deleted |
| insert 21 | 21 inserted |
| insert 13 | 13 inserted |
| maximum | 21 |
| minimum | 0 |
| postorder | 0 5 3 13 21 19 6 |
| successor 4 | 4 does not exist |
| predecessor 19 | 13 |
| predecessor 0 | predecessor of 0 does not exist |
| delete 0 | 0 deleted |
| delete 3 | 3 deleted |
| delete 5 | 5 deleted |
| delete 6 | 6 deleted |
| delete 19 | 19 deleted |
| delete 13 | 13 deleted |

| | |
|-----------|------------------|
| delete 21 | 21 deleted |
| delete 1 | 1 does not exist |
| inorder | |
| insert 1 | 1 inserted |
| inorder | 1 |

You can follow your own pseudocode for implementing these functions. But the 'effect' should be the same as that discussed in the class. For example, we know that a node can potentially be inserted at many places in a BST. But for this assignment, it is required that the node should be inserted at the leaf, as discussed in the class.

Submission

- The program you submit should output bst.txt when run.
- The main file of your program should be named as <roll no>.<extension>, where roll no. specifies your roll no. and the extension depends on the language you choose (Usage of C/C++ is mandatory for this assignment). Ex: 180040001.c
- Test well before submission. We have some hidden inputs with us to test your program. The mark you obtain is purely based on whether your program correctly gives outputs for the hidden inputs.
- If your program has only a single source file, please submit the file as it is. If your program has multiple source files, please submit your code as a zip file where the name of the zip file should be your roll number. It is important that you follow the input/output conventions exactly (including the naming scheme) as we may be doing an automated evaluation. There will be a penalty of 10% (on the mark you deserve otherwise) if you do not follow the naming conventions exactly.
- Follow some coding style uniformly. Provide proper comments in your code.
- Submit only through moodle. Submit well in advance. Any hiccups in the moodle/internet at the last minute is never acceptable as an excuse for late submission. Submissions through email or any other means will be ignored.
- Acknowledge the people (other than the instructor and TA) who helped you to solve this assignment. The details of the help you received and the names of the people who helped you (including internet sources, if applicable) should come in the beginning of the main file as a comment. Copying others' programs is a serious offence and deserving penalty will be imposed if found.

Evaluation

- To consider for first evaluation without penalty, you have to submit your program by 31st August (11:59 pm). If you submit after 31st August but on or before 7th September (11:59 pm), there will be a penalty of 10% on the marks you deserve otherwise.
- If you do not submit by 7th September, your program will not be considered for the first evaluation.
- We will do the first evaluation after 7th September. The marks you obtain will be proportional to the number of correct lines in the output files. We will use the 'diff' program to check the differences between the correct output file and the output file generated by your program. So, you may verify the correctness of output file by using

diff program with sample output file before submission. (See the man page of diff for more info).

- After the first evaluation, you will get a chance to improve your program. For this, after modification, you can submit your code for second evaluation. It comes with a 20% penalty. The due date for the second evaluation will be announced after the first evaluation. Those who submit their code after 7th September and before the due date for second evaluation will also be considered for the second evaluation. Submission done after the due date of the second evaluation will be ignored.