CS 211 Data Structures and Algorithms Lab
Autumn, 2021

Assignment no.	4
Objective	**To implement BST**
Total marks	**10**
Due date (without penalty)	**4th October (Monday) 11:59 pm**
Cut-off date (with penalty - 5%)	**9th October 11:59 pm**
Penalty for violating naming convention(s)	**5%**

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>, 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 in the BST	<maximum-number> / <empty-line> (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)

You can follow your own pseudocode for implementing these functions. 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._

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 is mandatory for this assignment). Ex: 200010001.c.
- Test well before submission. You may use the attached sample input file for testing. The corresponding output file is also attached. 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 5% (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 a deserving penalty will be imposed if found.

Evaluation
- To consider for the first evaluation without penalty, you have to submit your program by the due date. If you submit after the **due date** but on or before the **cut-off date**, **there will be a penalty of 5% on the marks you deserve otherwise**.
- If you do not submit by the **cut-off date,** your program will not be considered for the first evaluation.
- We will do the first evaluation after the cut-off date. 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 the output file by using the diff program with a sample output file before submission. (See the man page of diff for more info).
- We will do the second evaluation later. This is for those who want to improve their marks obtained in the first evaluation or who do not submit for the first evaluation. There will be **a penalty of 20%** for those who are submitting for the second evaluation. The details of the second evaluation will be shared later.