Offline on Binary Search Tree

Implement a binary search tree of integers having the following functions:

- *insertItem* : This function inserts a new item in the binary search tree.
- *searchItem* : This function searches for an item in the binary search tree. If the item, say 52, is found, the function should print "52 has been found"; otherwise the function should print "52 has not been found".
- *getInOrderSuccessor* : This function returns the in-order successor of an item in the binary search tree. When there is no successor, return an invalid number to indicate it.
- *getInOrderPredecessor* : This function returns the in-order predecessor of an item in the binary search tree. When there is no predecessor, return an invalid number to indicate it.
- *deleteItem* : The function will delete an existing item from the binary search tree. You must re-structure the tree after deletion according to the idea given in class. In case of deletion of a node where both of its children exist, you must choose the next largest node in the tree for replacement. If the item is not in the tree, do nothing.
- *getItemDepth* : This function returns the depth of an item in the binary search tree. Assume that the root has a depth of 0, the nodes at the next level have a depth of 1 and so on. If the item is not in the tree, return an invalid number to indicate it.
- *getMaxItem* : This function finds and returns the maximum item of the binary search tree. You cannot write recursive function for this task.
- *getMinItem* : This function finds and returns the minimum item of the binary search tree. You cannot write recursive function for this task.
- *getHeight* : This function returns the height of the binary search tree.
- *printInOrder* : This function prints the in-order traversal of the binary search tree.
- *printPreOrder* : This function prints the pre-order traversal of the binary search tree.
- *printPostOrder* : This function prints the post-order traversal of the binary search tree.
- *getSize* : This function returns the current size of the binary search tree. The size of a tree is the number of nodes in the tree. You should write a recursive function for this task.

Except the last 5 functions, all the other functions should run in O(h) time, where h is the height of the binary search tree. Here "item" means an integer.

You can assume that all the items in the tree are distinct, i.e. there is no repetition. Each node of the binary search tree should have two nodes- one for left child and another for right child.

Write a menu in an infinite loop in main function. The menu should have option to perform any of the mentioned functionalities at a time, like this:

1. Insert Item 2. Search Item 3. Get In Order Successor ….

Take input from the user in the infinite loop and perform the desired operation. Don't do any hard-code. In case of the functions that returned a value, print the corresponding result in the main function.

Special Instructions:

Write readable, re-usable, well-structured, quality code. This includes but is not limited to writing appropriate functions for implementation of the required algorithms, meaningful naming of the variables, suitable comments where required, proper indentation etc.

Please DO NOT COPY solutions from anywhere (your friends, seniors, internet etc.). Any form of plagiarism (irrespective of source or destination), will result in getting -100% marks in the offline. Also, be informed that for repeated offence of plagiarism, the departmental policies suggest stricter measures.

Submission Guideline:

1. Create a directory with your 7 digit student id as its name.

2. Put the source files only into the directory created in step 1.

3. Zip the directory (compress in .zip format; .rar, .7z or any other format is not acceptable).

4. Upload the .zip file on Moodle.

For example, if your student id is 1805xxx, create a directory named 1805xxx. Put only your source files(.c, .cpp, .java, .h, etc.) into 1805xxx. Compress 1805xxx into 1805xxx.zip and upload the 1805xxx.zip on Moodle.

Failure to follow the above-mentioned submission guideline may result in upto 10% penalty.

Submission Deadline: April 2, 2021 11:55 PM