Programming Assignment 5

Due by 10월 11일 저녁 9시

1. Implement `add()`, `delete()`, `print_inorder()`, `height()` and submit "backend-bst.c". DO NOT CHANGE ANYTHING ELSE.

2. For this programming assignment, we implement the address book with a binary search tree. As always, `data` is the pointer to the binary search tree that stores the data for the address book.

3. Note that we have a new command "`H`" and "`h`" that prints the height of the BST for the Address Book. By definition, the height of an empty binary tree is -1!

4. Note that we adopt "$\leq$ or $>$" rule when we `add` a key. That is, when we `add` a key that is already in the BST, the new key is `add`ed to the left subtree of the node of the already existing key.

5. We implement our own memory management almost the same way as before. The only difference is that the nodes have a different structure and size; a node has two links. As before, `new_node()` returns `NULL` when the pool is empty. As in HW3, we provide it only in compiled form as "memory.o". **Since it is linux-binary, it works only in linux.**

6. The function `delete()` is the most difficult part of this assignment. There can be many ways to implement it, but you should stick to "replace with successor" rule.