On my honor, I pledge that I have neither received nor provided improper assistance in the completion of this assignment.
서명: ***강동인*** 학번: 21500002___

Operations: Insert (or grow)

- grow(node, k) - Insert a node with k
  - **Step 1**: If the tree is empty, return a new node(k).
  - **Step 2**: Pretending to search for k in BST, until locating a nullptr.
  - **Step 3**: create a new node(k) and link it.

```
tree grow(tree node, int key) {
  if (node == nullptr)
    return new tree(key);

  if (key < node->key)
    node->left = grow(node->left, key);
  else if (key > node->key)
    node->right = grow(node->right, key);
  return node;
}
```

- Q1: Explain the differences between the binary tree and binary search tree in this operation.

  Since the values of the left and right subtrees of BST are always smaller or larger than the root, you only need to compare one subtree by comparing the key to be inserted with the root values of the left subtree and right subtree. However, in the case of BT, since there is no relationship between subtrees, all subtrees must be searched.

- Q2: To complete inserting **7**, how many times was **`grow()`** called?

  4 times

- Q3: How many times "**`if (key < node->key)`**" called during this process or inserting 7?

  1 time

- Q4: At the end of this whole process, which **`return`** will be executed and what is the key value of the node?

  return new tree(key), key = 7