OLD Quiz 6 Version C

Due No due date	**Points** 8	**Questions** 3	**Available** after Nov 18 at 1pm
Time Limit None	**Allowed Attempts** Unlimited		

Instructions

Quiz 6: Binary Search Trees

7 points required to pass

Take the Quiz Again

Attempt History

	Attempt	Time	Score
LATEST	Attempt 1	less than 1 minute	0 out of 8 *

* Some questions not yet graded

Score for this attempt: **0** out of 8 *
Submitted Dec 5 at 2:09pm
This attempt took less than 1 minute.

Question 1

Not yet graded / 3 pts

Suppose we have this structure definition for a node in a binary search tree (BST) of **int**:

struct bt_node { int value; bt_node *left; bt_node *right; };

The header node in a calling function is of type pointer-to **bt_node**. A header node of **nullptr** (or **0**) indicates an empty tree. The ordering rule for the BST is that all values less than the current node's value are to the left, and all values greater are to the right. Duplicate values are ***not allowed***.

Write the implementation of this function that returns the number of values in a binary search tree that are greater than or equal to **N**:

```
size_t binary_tree_count_ge_N(const bt_node *top, int N)
{
    // this is the code you have to write
}
```

Your Answer:

fdas

```
if (top == nullptr)
   return 0;  // no values in this tree
if (top->value < N)
   return binary_tree_count_ge_N(top->right, N);  // nothing to left is >= N
// top->value >= N
return 1 + binary_tree_count_ge_N(top->left, N)
         + binary_tree_count_ge_N(top->right, N);
```

Question 2

Not yet graded / 2 pts

Suppose we have this structure definition for a node in a binary search tree (BST) of **int**:

struct bt_node { int value; bt_node *left; bt_node *right };

The header node in a calling function is of type pointer-to **bt_node**. A header node of **nullptr** (or **0**) indicates an empty tree. The ordering rule for the BST is that all values less than the current node's value are to the left, and all values greater are to the right. Duplicate values are ***not allowed***.

Write the implementation of this function that creates an ***empty*** binary tree, and returns the appropriate value for the header node:

```
bt_node *mk_empty_binary_tree()

{

    // this is the code you have to write

}
```

Your Answer:

```
return nullptr;
```

Question 3

Not yet graded / 3 pts

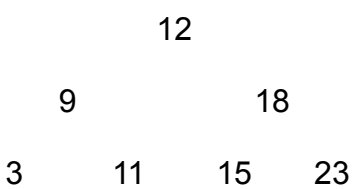
Suppose we have this structure definition for a node in a binary search tree (BST) of **int**:

struct bt_node { int value; bt_node *left; bt_node *right };

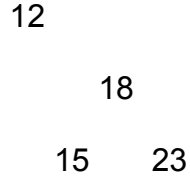
The header node in a calling function is of type pointer-to **bt_node**. A header node of **nullptr** (or **0**) indicates an empty tree. The ordering rule for the BST is that all values less than the current node's value are to the left, and all values greater are to the right. Duplicate values are ***not allowed***.

Write the implementation of this function that ***deletes the left subtree*** of the

argument tree. For example, if the tree originally contained:



then after this function returns the tree should contain:



If the tree is empty, or has no left subtree, the function should do nothing.

You may find it useful to define and call a helper function.

```
void binary_tree_delete_left_subtree(bt_node **ptop)
{
    // this is the code you have to write
}
```

Your Answer:

```
// body of binary_tree_delete_left_subtree:
if (*ptop == nullptr || (*ptop)->left == nullptr)
    return;
binary_tree_delete(&(*ptop)->left);

// binary_tree_delete function:
void binary_tree_delete(bt_node **ptop)
{
    if (*ptop) {
        binary_tree_delete(&(*ptop)->left);
        binary_tree_delete(&(*ptop)->right);
        delete *ptop;
        *ptop = nullptr;
    }
}
```

Quiz Score: **0** out of 8