1. True/False

T/F: The destruction (clearing) of a tree should be implemented as a pre-order traversal.

Answer: False, it is a post-order traversal, children must be destroyed before parents. (Page 462)

1. Multiple Choice

Given a tree constructed from the following data (or items with the following keys), in the order they are presented: ‘a’, ‘b’, ‘f’,‘d’, ‘e’. Which of the following items would be contained right-most, bottom-most (in that order) after the insertion of ‘c’?

1. ‘z’
2. ‘a’
3. ‘f’
4. ‘d’
5. ‘c’

Answer: C. ‘f’ (Page 464)

1. Fill in the Blank

When removing a node with two children from a tree, instead of removing the node with the data to be removed, the data of that node is often replaced with a different node’s data and that other node is actually removed (unless that node also has two children). This “other” node is often known as \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

Answer: The in-order predecessor (successor is also acceptable). (Page 476)

1. Short Answer or Code

Implement the overloaded assignment operator for a binary search tree that only contains characters.

Answer:

CharBST& CharBST::operator=( const CharBST& other )

{

if( this != &other )

{

clear();

if( !other.isEmpty )

{

clone\_sub( other.root );

}

}

return \*this;

}

void CharBST::clone\_sub( Node\*& current,

const Node\* otherCurrent )

{

if( otherCurrent != NULL )

{

current = new Node( otherCurrent->data, NULL, NULL );

clone\_sub( current->left, otherCurrent->left );

clone\_sub( current->right, otherCurrent->right );

}

}

CharBST::Node::Node( const char newCh, const Node\* newLeft,

Const Node\* newRight )

{

data = newCh;

left = newLeft;

right = newRight;

}

(Page 463)