

CE 242 - Data Structures and Algorithms  
Spring 2009 - Ahmet Ardal  
Lab Assignment 4

1. Using the reference binary search tree implementation, write code for the implementation of `BTreeRotateRight()` and `BTreeRotateLeft()` functions. The code for those functions' declarations and definitions are provided in the `BTree.h` and `BTree.cpp` source files and what you are asked to do is just filling in the function definitions. Further explanation is given as inline code comment in the `BTree.h` and `BTree.cpp` source files. For detailed information about tree rotations check this: [http://en.wikipedia.org/wiki/Tree\\_rotation](http://en.wikipedia.org/wiki/Tree_rotation)

2. Using the reference binary search tree implementation, write code for the implementation of `BTreeInsertRoot()` function. The code for this function's declaration and definition is provided in the `BTree.h` and `BTree.cpp` source files and what you are asked to do is just filling in the function definition. Further explanation is given as inline code comment in the `BTree.h` and `BTree.cpp` source files.

3. Using the reference binary search tree implementation, write an alternative version of the `BTreeRemove()` function that is current implementation replaces the removed node with its in-order successor (the left-most child of the right subtree), your implementation should replace the removed node with its in-order predecessor (the right-most child of the left subtree). Again, the code for this function's declaration and definition is provided in the `BTree.h` and `BTree.cpp` source files and what you are asked to do is just filling in the function definition. Further explanation is given as inline code comment in the `BTree.h` and `BTree.cpp` source files.

Note:

- Test code for this homework is included in the file `testBTree.cpp`.
- Output of the test code for `BTreeInsertRoot()` function:

```
5, 10, 12
10, 11, 12
-----
5
10
11
12
-----
12
11
10
5
-----
```

- Output of the test code for other `BTree` module functions:

```
root: 10
root: 1
root: 1
root: 10
1
5
6
```

```
7
8
9
10
11
13
14
-----
14
13
11
10
9
8
7
6
5
1
-----
data found: 14
-----
removed...
-----
1
5
6
7
8
9
11
13
14
-----
```