Week 07: Binary Search Trees

Lab Tutor: For this lab-tutorial session, please discuss about the solution for each question in the lab. You may allocate about 30 minutes for each question. No need to discuss about the assignment questions and practice questions.

Questions:

1. Write a function `insertBSTNode()` that adds an item to a Binary Search Tree.

   ```
   void insertBSTNode(BTNode **cur, int value);
   ```

   BST nodes should be created dynamically using a `malloc()` call. Please try to write a non-recursive solution.
   Note: Make sure that your code is able to correctly add a node into an empty BST.

   A sample input and its output is given below:

```
Enter a list of numbers for a Binary Tree, terminated by any
non-digit character:
31 93 14 70 94 99 23 7 67 93 7 25 a
Duplicated Item: 93
Duplicated Item: 7
The Binary Search Tree:
31
|---14
|       |---7
|       |___23
|       |       |___25
|___93
|       |---70
|       |       |---67
|       |___94
|       |       |___99
```

2. Write a function `removeBSTNode()` that removes a given item from a Binary Search Tree. The function should return 1 if the item was found and successfully removed and 0 otherwise. The prototype of the function is given below.

   ```
   int removeBSTNode(BTNode **nodePtr, int item);
   ```

   A sample input and its output is given below:

```
Enter a list of numbers for a Binary Tree, terminated by any
non-digit character:
31 93 14 70 94 99 23 7 67 25 71 66 -20 0 88 a
The Binary Search Tree:
31
|---14
|        |---7
|        |        |----20
|        |        |        |___0
|        |___23
|        |        |___25
|___93
|        |---70
|        |        |---67
|        |        |        |---66
|        |        |___71
|        |        |        |___88
|        |___94
|        |        |___99
Enter an integer to be removed from the tree:
93
93 was removed
The Binary Search Tree:
31
|---14
|        |---7
|        |        |----20
|        |        |        |___0
|        |___23
|        |        |___25
|___88
|        |---70
|        |        |---67
|        |        |        |---66
|        |        |___71
|        |___94
|        |        |___99
```

3. Write a function `isBST()` that determines whether a given Binary Tree is also a Binary Search Tree. The function should return 1 if the BT is a BST, and 0 otherwise.

```
int isBST(BTNode *root);
```

Sample inputs and outputs are given below:

```
Enter a list of numbers for a Binary Tree, terminated by any
non-digit character:
```

```
21 15 78 a
The binary tree is a BST:
21
|---15
|___78

The binary tree is not a BST:
21
|---15
|       |___78

Enter a list of numbers for a Binary Tree, terminated by any
non-digit character:
31 93 14 70 94 99 23 7 67 25 a
The binary tree is a BST:
31
|---14
|       |---7
|       |___23
|       |       |___25
|___93
|       |---70
|       |       |---67
|       |___94
|       |       |___99

The binary tree is not a BST:
31
|---93
|       |---7
|       |       |---67
|       |___14
|       |       |---25
|       |       |___94
|___70
|       |---23
|       |___99
```

4. Write a function rotateRNode() that rotates right about a given node. The binary tree will be updated after the function is called. The prototype of the function is given below.

```
void rotateRNode(BTNode **node)
```

A sample input and its output is given below:

```
Enter a list of numbers for a Binary Tree, terminated by any
non-digit character:
70 14 93 7 31 73 99 10 23 45 97 95 a
70
|---14
|       |---7
|       |       |___10
|       |___31
|       |       |---23
|       |       |___45
|___93
|       |---73
|       |___99
|       |       |---97
|       |       |       |---95
Enter an integer to be searched in the tree:
93
Found
about its left child (0) or right child (1): 1
70
|---14
|       |---7
|       |       |___10
|       |___31
|       |       |---23
|       |       |___45
|___93
|       |---73
|       |___97
|       |       |---95
|       |       |___99
```

You may also implement the left rotation about the given node.