**PROBLEM DESCRIPTION:**
Write a program that uses random integers to build a random Binary Tree and Binary Search Tree.

**PURPOSE:**
- Use structured programming, to improve readability and flexibility.
- Learn to use Inheritance.
- Learn to use virtual functions.
- Learn to define and implement Generic Binary Tree and Binary Search Tree Data Structures.
- Learn to use different user-defined data structures in a program.

**REQUIREMENTS:** Strictly adhere to all the requirements. Any submission that does not follow any one of them will be either rejected summarily or penalized heavily.

**DATA STRUCTURES:**
1) Use a linked list to define (**Stack.h**) and implement (**Stack.cpp**) Generic Stack Data Structure.
2) Use a static array to define (**Queue.h**) and implement (**Queue.cpp**) Generic Queue Data Structure.
3) Define and implement the following Generic Data Structures:
   a) **NodeType** (**NodeType.h & NodeType.cpp**).
   b) **BTree** (**BTree.h & BTree.cpp**) as a base class.
   c) **BSTree** (**BSTree.h & BSTree.cpp**) as a derived class of **BTree**.

**MEMBER FUNCTIONS:**
4) Include the indicated functions as members:

<table>
<thead>
<tr>
<th>NodeType</th>
<th>BTree</th>
<th>BSTree</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 Constructors</td>
<td>Constructor, Destructor</td>
<td>Constructor, Destructor</td>
</tr>
<tr>
<td></td>
<td>PreOrder, InOrder, PostOrder</td>
<td>Insert (as a virtual function)</td>
</tr>
<tr>
<td></td>
<td>BFS, DFS</td>
<td>Delete</td>
</tr>
<tr>
<td></td>
<td>Height, NumberOfNodes</td>
<td>Search</td>
</tr>
<tr>
<td></td>
<td>Clear</td>
<td></td>
</tr>
</tbody>
</table>

5) All five functions **PreOrder, InOrder, PostOrder, BFS** and **DFS** must print the value in a node as well as the level of the node within parentheses.

6) **BFS** and **DFS** must be implemented using the user-defined **Stack** Data Structure and **Queue** Data structure.

7) **Insert** must be defined as a virtual function in **BTree** and **BSTree**. It must set the level of the new node as it inserts a value.

8) When **Delete** function deletes a node from a BST, the level of all the nodes in the left and right branches of that node, will be off by one. You do not have to fix that, but if you do, you will get extra credit points.

9) It is not required but as a good programming practice, you should write a main function in the implementation file (**.cpp**) of each data structure to test the correctness of your implementation. But once verified, each main function must be commented out (do not delete it). Remember a program must have only one main function.
**MAIN FUNCTION:**

10) The main program must do the following at least 5 times:

   a) Randomly decide the number of nodes in a tree. Since you need to draw the trees manually, get a value between 15 and 20: NumOfNodes = 15 + rand()%6;

   b) Generate the required number of random integers. Using the same set of integers, build a random Binary Tree and a Binary Search Tree.

   c) For each tree, print:
      i) The type of tree (BT or BST)
      ii) The number of nodes
      iii) The height of the tree
      iv) Values in the nodes of the trees using all 5 traversal functions.
      v) To verify the results, manually draw the tree.

   d) For each BST, verify the two functions: Search and Delete. Print the tree after each deletion.

**STYLE:**

11) Carefully read the guidelines given in “Requirements for all Projects”. You will lose at least 5 points for every violation of the rules indicated in the document. A link to this file is on my website: http://jjcweb.jjay.cuny.edu/mmanth/ListOfClasses.html. Go to the course page and click on “Projects”.

---

**SUBMIT IN AN ENVELOPE:**

1) **Part A: [15 points]** Due on Nov 24, 2009. Submit hard copies of the following, printed using a C++ editor:

   a) [5] NodeType.h
   b) [10] BTree.h
   c) [5] BSTree.h

2) **Part B: [65 points]** Due on Dec 1, 2009. Submit hard copies of the following, printed using a C++ editor:

   a) [5] NodeType.cpp
   b) [30] BTree.cpp
   c) [15] BSTree.cpp
   d) [15] Queue.h & Queue.cpp

3) **Part C: [55 points]** Due on Dec 10, 2009. Submit hard copies of the following, printed using a C++ editor:

   a) [20] BuildBTree.cpp
   b) [15] Output File with at least 10 trees (5 sets of BT & BST)
   c) [20] Manual drawing of all trees

4) **Part C: [5 points]** Due on Dec 10, 2009. A one-page commentary on your experience from completing this project, detailing what you learned, the difficulties you encountered and your deficiencies in basic knowledge required to implement the program in C++. Acknowledge also the help you received. Use Microsoft Word (or any word processor) to prepare this document.

   **Read the Rules on any Submission in the Course Policy.**

11/22/2009  Mantharam