Programming Homework Assignment #4
Due Card: Thur., Nov. 5, 11:59 PM
Upload the source files (.h and .cpp files) with output (copied and pasted to the end of the main file) ZIPPED into one .zip file (submit under Week 8)

Problem:
Write a C++ program which changes and completes the Binary Tree and Binary Search Tree (BST) classes and practices using BST objects.

Complete the Binary Tree class member functions given in the HW4_Code file (I'm calling "code file" below):

- copy constructor (assign copyTree(passing parameter's rootPtr) to rootPtr) (hint: see operator= in this BinaryTree class for how to call copyTree correctly)
- destructor (algorithm given in the code file)
- copyTree (algorithm given in the code file is PROTECTED)
- destroyTree (algorithm given in the code file)
- _inorder (similar to _preorder)
- _postorder (similar to _preorder)

Change the Binary Search Tree class so it has a PRIVATE pointer to function member variable (returns an int, has 2 const ItemType& parameters) and changes the following:

- ADD a constructor that has a compare function parameter and assign the parameter to the pointer to function member variable
- ADD a COPY constructor that assigns the parameter's pointer to function member variable to this' pointer to function member variable, then calls the BinaryTree's copyTree
- every function that uses < or == or > MUST be changed to use the compare function that returns an int (examples will be given in a separate file)
- complete or CHANGE these functions:
    - getEntry (algorithm given in the code file)
    - findNode (algorithm given in the code file)
    - getFirst() and getLast() (see code file)

Use the Card class given in the HW4_CodeFile.

In your main file, typedef the Card* (I'm calling it PTR_CARD) AFTER you #include "Card.h". Also, define the following standalone (non-member) functions:

- int comparePips(const PTR_CARD &left, const PTR_CARD &right), is similar to operator<, but returns 0 if both members are the same, -1 if operator < returns true, or 1 otherwise
- int compareSuits(const PTR_CARD &left, const PTR_CARD &right), which return 0 if both members are the same, -1 if left’s suit < right’s suit

OR (left’s suit equals right’s suit AND left’s pips < right’s pips), otherwise, return 1

- void displayPTR_CARD(PTR_CARD &ptrCard), which writes the "dereferenced" parameter to cout (calling its operator<<), then cout<<endl;
- void displaySuitPTR_CARD(PTR_CARD &ptrCard), which writes to cout the members of the Card in the format: suitName, pipsName (use the accessors)

Write main so it has 2 POINTER to BinarySearchTree<PTR_CARD> variables. Assign to one of the pointer variables a new BinarySearchTree with the comparePips passed as an argument. Assign to the 2nd pointer variable a new BinarySearchTree with compareSuits as an argument. Then do the following (for which most should be a function or points may be deducted if main is too long): (MUST BE IN THIS ORDER)

- call a function (that you write) to call srand(time(0)), then fill both BinarySearchTrees in a loop for 25 times (don't worry about duplicates):
  - o get a random int from 1 to 13, inclusive (assign to a local int for the pips)
  - o get a random int from 0 to 3, inclusive, assign to another local int for the suit number
  - o DYNAMICALLY ALLOCATE a Card (passing the local random ints)
  - o **insert** the <u>same</u> Card to each BinarySearchTree (through its pointer), one at a time
- call each tree's **inOrder** function (one at a time), passing displayPTR_CARD for the pips-ordered BST and displaySuitPTR_CARD for the suit-ordered BST
- call the standalone **testBST** function (given in the code file) for EACH tree (one at a time)
- call a <u>function that you write</u> that tests deleting from BOTH BinarySearchTrees (reference or pointer parameters) in ONE function so it does the following:
  - o Do the following for each tree:
    - ▪ get the first and last Card from the tree
    - ▪ try to remove the first, then the last, check if successful and display a message indicating if removed
- call the standalone **testCopyAndAssign** function (given in the code file)
- call **postOrder** for ONE OF THE sorted BinarySearchTree passing deletePTR_CARD (function), then delete EACH tree (one at a time)

See test runs on Catalyst. DO NOT USE ANY EXTERNAL VARIABLES (variables declared outside of main or outside of any function)!! (External const declarations are OK.)

Extra Credit Problems (due the last day of the quarter!): Textbook (Data Abstraction & problem Solving by Carrano) pp. 490-491 #19 (for our BinarySearchTree), and TEST in a program.