**Programming Homework Assignment #4**
Due Date: Friday, May 26, 11:55 PM
Upload the source files (.java files) with output (copied and pasted to the end of the main file) ZIPPED into one .zip file (submit under Week 7)

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

Complete the BinaryTree class member methods given in the HW4_JavaCode file:

- public writeIndentedTree (described in the code file)
- protected writeTree (described in the code file)
- protected inorder method (similar to preorder) with the BinaryNode parameter as well as Visitor parameter
- protected postorder (similar to preorder) with the BinaryNode parameter as well as Visitor parameter

Change part of the Binary Search Tree (BST) class as indicated in the HW4_JavaCodefile (note that the class heading is now: **public class** BST<E> **extends** BinaryTree<E>, but the class has a **PRIVATE Comparator variable**):

- ADD to the default constructor (AND the constructor with an array parameter) a Comparator<E> parameter and assign the parameter to the Comparator<E> instance variable
- every method that compares data MUST REPLACE the *compareTo* calls with the **compare** method that has 2 E parameters and returns an int (examples will be given in a separate file)
- complete/fix these methods:
  - getEntry (algorithm given in the code file)
  - findNode (algorithm given in the code file)
  - getFirst() and getLast() (see details in the code file)

Use the separate class given in the HW4-JavaCodeFile to be used for the data in the list (the same Date class as in HW#3, but MAKE SURE THE compareTo is CORRECT).

Define 2 Comparator classes which implement Comparator<Date> (hints given in the Comparators file on Catalyst):

- called YearComparator that overrides one method: int compare (Date left, Date right), returns the result of calling compareTo for the left parameter, passing the right parameter, BUT MAKE SURE THE Date's compareTo is correct (I'll give feedback about your compareTo before this is due)
- called MonthComparator that overrides one method: int compare (Date left, Date right), which returns difference of the left's month - right's month, BUT if the months are the same, the difference of the left's day - right's day, BUT if the days are the same, too, the difference between left's year - right's year

Define 2 Visitor classes which implement Visitor<Date>:

- called YearVisitor that overrides one method: void visit(Date dt) that writes the Date in the format: year-month-day (use the accessors)
- call MonthVisitor that overrides one method: void visit(Date dt) that writes the toString of the parameter to System.out

Write main IN A SEPARATE CLASS AND FILE so it has 2 BST<Date> variables. Assign to one of the variables a new BST with a new YearComparator passed as an argument. Assign to the 2nd variable a new BST with a new MonthComparator 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)

- open an input file using the openInputFile() method given in the HW4_JavaCodeFile (DON'T CHANGE THE openInputFile METHOD)! If it doesn't open, quit the program!
- read from the file, which has 3 ints per line (for month, day and year, IN THAT ORDER). BEFORE the input loop, instantiate a new default Date and assign to a local Date variable. In a loop, read 3 ints, then call the setDate() method for the local Date passing the 3 ints. If setDate returns true, instantiate a NEW Date object (passing the input month, day, and year) and insert that same Date object to BOTH BSTs (one at a time) (otherwise, display a message that the input is invalid and display the invalid month, day and year)
- after the input loop, close in the input file
- call each BST's **inorder** method (one at a time), passing a YearVisitor for the Year-ordered BST and a MonthVisitor for the Month-ordered BST (please display good headings so we know what is being displayed)
- call **writeIndentedTree**() passing System.out, for EACH BST (one at a time), with appropriate headings
- call for both trees (ONE AT A TIME) a static method that you write that tests the getFirst, getLast, insert, getEntry methods from a BST<Date> (parameter) in ONE method so it does the following for the BST parameter:
  - get the first and last Date from the tree (using the BST's getFirst and getLast methods), save into local variables, and display them
  - make a copy of the first Date (create a new Date with the same month, day and year as the first Date
  - call getEntry passing the copy of the first Date, and display the return value (with labels as shown in the test runs)
  - make a copy of the last Date, but CHANGING it (create a new Date with the same month, day, but (year+1) as the last Date), then display it
  - call getEntry passing the almost copy of the last Date, and display the return value (with labels as shown in the test runs)
  - try to insert the almost copy of the last Date and display a message showing what the insert method returned

- call **postorder** for the Year-ordered BST passing a YearVisitor (cont'd-->)

- call **writeIndentedTree**() passing System.out, for the Month-ordered BST
- call the **testMore()** method given in the HW4_JavaCode file for EACH BST (call a total of 2 times)

See test runs on Catalyst. Use the input files given on Catalyst.

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