COP3503 Computer Science 2
Dr. Andrew Steinberg
Fall 2021

Programming Assignment 2
Address Book Assignment
Due: 10/31/2021 at 11:59pm

Objective: Students will apply concepts of Java Generics and Red-Black Trees in this programming assignment.

Assignment Description: In this class, we have learned a couple of advanced data structures related to trees. In this assignment, you will manage an address book of professors with the abstract data type (ADT) red-black tree data structure. You will create your own Red-Black Trees using Java Generics This programming assignment must be completed in the Java programming language. Any other language used will result in score of 0 on the assignment.

For this assignment, you must follow these **requirements**.

1. You will create an ADT Red-Black Tree class called `AddressBookTree` with Java Generics (type T and type U). The attribute for the class must contain:
    a. root of Node
    b. You may create additional helper attributes that can assist with maintaining the red-black tree.
2. You will use two unique type variables T and U. T represents the name and U represents office.
3. You will create a `Node` class that will represent the nodes in a red-black tree using generics. The attributes needed for the class are
    a. Name of type T (Generics)
    b. Office of type U (Generics)
    c. Reference to parent node
    d. Reference to left child node
    e. Reference to right child node
    f. An int that represents color (0 for black and 1 for red)
4. The `AddressBookTree` class must have the following methods. Most of the methods have been provided to you in the pseudocode on the slides.
    a. Default constructor
        i. Set everything to default values.
    b. Insert
        i. Parameter takes two arguments: T name, U office
    c. Insert-Fix (applying rules for insertion)
        i. Parameter takes one argument: Node<T, U>
    d. Delete
        i. Parameter takes one argument: T name
    e. Delete-Fix (applying rules for deletion)
        i. Parameter takes one argument: Node<T, U>

   f. Rotations (both left and right)
      i. Parameter for both takes one argument: Node<T, U>
   g. RB-Transplant
      i. Parameter takes two arguments: Node<T, U> and Node<T,U>
   h. Display (inorder traversal)
      i. No arguments needed.
   i. countBlack – method to count black nodes in tree
      i. Parameter takes Node to count from (the root)
   j. countRed – method to count red nodes in tree
      i. Parameter takes Node to count from (the root)
   k. You may also create additional helper methods as well.
5. Make all methods public and class attribute private. It's good practice!

A runner file (`AddressBookTreeRunner.java`) has been provided for you to show you how the methods are called along with 3 test cases based on maintaining the tree after certain operations. A text file is also provided for you that will help fill the tree in the initial run. The text file itself must be in the same directory as the runner file.

What to submit: Submit a file called `AddressBookTree.java` to webcourses. You are not required to submit the runner file as that will be provided for the graders to test your code. Please make sure the runner file provided works for your code. Any name changes may cause your program not to work when graded, which will result in a lower score on the assignment and would not be changed.

Important Note for running Eustis: Many of you are probably using IDEs like Netbeans and Eclipse to build your Java Solutions. Please note you will need to separate the two files. In other words, do not put the runner file in the same package as `AddressBookTree.java`. If you put both files in the same package, the command line will have issues with compiling. The workaround this is putting your `AddressBookTree.java` file in a package called "`addressbook`" and have the runner file import it (see the runner file for clarification). **Please make sure to name the package `addressbook`. Check out the runner file!!!**