University of Victoria
Faculty of Engineering: Computer Science

CSC 115: Fundamentals of Programming II

Spring 2016

Assignment 4: Patient Location

Objectives: Upon completion of this assignment you need to be able to:

- Become familiar with object *inheritance*.
- Become more familiar with use of generics.
- Use a BinarySearchTree to store and manipulate comparable items.
- Traverse the tree to produce an ordered list.
- Continue to apply good programming practice in
  - designing programs
  - proper documentation,
  - testing and debugging problems with code.

Resources:

- CSC 115 Java Coding Conventions.pdf (found on conneX, under Resources)
- Textbook Chapter 11
- The specification pages provided.

Introduction:

The local hospital Admissions Department needs to keep a record of current patients and room numbers. During the day, the admission staff will update, insert, delete and retrieve patient information from one of the workstations in the Admissions Department. Visitors make enquiries at the desk to locate their loved ones in the hospital. In the evening, when the Admissions Desk staff go home, two volunteers take over the job of locating patients for the evening and night visitors. However, the volunteers to do not have access to a workstation and therefore work from a printout of ordered patient names and room numbers. This list is printed daily for them by one of the Admission staff just before she goes home.

We will provide a BinarySearchTree data structure to handle the PatientLocation information, including inserting new patients, removing patients and updating information. We will also add the functionality to allow the Admissions staff to create a daily ordered printout for the volunteers.

Quick Start:

1) If you haven't done so already, download all the necessary documents for this assignment from conneX into a specific folder for CSC115 assignment three, `assn4` is a recommended name. You may want to separate the documentation files into a separate directory.
2) All of the specification documentation for public class and public methods can be found in the appropriate `*.html` files.
3) The following classes are complete: `TreeNode.java`, `PatientLocation.java`, and `DrawableBTree.java`.
4) The `BinaryTree` class needs to be completed.
5) The `BinarySearchTree` class needs to be completed.

Detailed Instructions:

Note that the Textbook has detailed instructions on programming the BinaryTree and BinarySearchTree. This assignment follows most of the authors' definitions and instructions.

The BinaryTree class:

The `BinaryTree` class is a very basic tree, with no order imposed on its elements and very little functionality. Implement the methods as instructed, following the specifications. Note that there is no opportunity to add elements to this tree so testing in the main method is not possible. The testing for this tree must be done in any class that `extends` this class.

The BinarySearchTree class:

The `BinarySearchTree` class `extends BinaryTree` by specifically storing elements with an internal order imposed on them. We enforce this ordeing by only allowing elements that `implement Comparable`, in other words, the element class implements a `compareTo` method that determines which of two elements come earlier in an ordered list.
You are to implement all the the methods as per the specification document. Most methods are described in the textbook; the authors have chosen to use recursion, but you are free to use whatever algorithms make sense to you.

The DrawableBTree class:

This class is simply a tool to visualize the structure of the `BinaryTree` during internal testing.You do not need to understand or alter any of the code. To invoke a rendering, you only need two lines of code in the `BinarySearchTree` main method; they are provided. Once you are able to test the insert methods, uncomment them.
At the point of the statement to `showFrame()`, a frame becomes visible on the console. You can resize the frame as desired and the tree will expand to fit the frame. The program continues on but does not completely quit until the frame is closed.

Submission

Submit the following completed files to the Assignment folder on conneX:

- `BinaryTree.java`
- `BinarySearchTree.java`

Please make sure that conneX has sent you a confirmation email. Do not send [`.class`] (the byte code) file, or any another file for this assignment. Also make sure you *submit* your assignment, not just save a draft. All draft copies are not available to the instructors, so we cannot mark them.

Note about Academic Integrity: It is OK to talk about your assignment with your classmates, and you are encouraged to design solutions together, but each student must implement (code) their own solution.
We will be using plagiarism detection software on your assignment submissions.

Grading

All submitted java files must compile.
The marker will be looking for:

- Proper programming style as per the code conventions on CSC115 conneX Resources.
- Source code follows the specifications and instructions.
- Source code takes full advantage of *inheritance*.
- Good modularity: well-defined helper methods.
- Internal testing, using the main method as a test harness.