

Hands-On: Binary Search Trees

This activity focuses on the fundamental mechanics of adding new values to a binary search tree. You should study the [instructional resources](#) on binary search trees before attempting this activity.

You will need the following source code files to complete this activity.

- [BinarySearchTree.java](#)
- [BstClient.java](#)

Note: This activity utilizes [jGRASP Viewers](#), which are available in [jGRASP](#), [IntelliJ](#), and [Eclipse](#).

Adding values, self-check

1. Open `BstClient.java` in jGRASP and compile it.
2. Run this program, observe the output, and make sure you understand what it is doing.
3. Set a breakpoint on the statement `bst.add(value)` in the `main` method.
4. Start the debugger and wait until execution is paused at the breakpoint.
5. Raise a viewer on the `bst` object. Use the viewer settings menu to customize the viewer as you like. For example, you may prefer to have the nodes rendered as circles, and you will most likely prefer that the root be at the top with descendants below.
6. With pen and paper, add the current `value` to a binary search tree that you draw yourself.
7. Step over the call to `add` to confirm your answer.
8. Repeat this process for each element in the `values` array.
9. Make sure you understand how to build a binary search tree from a given sequence of values.

Adding values, iterative `add` method

1. Open `BstClient.java` in jGRASP and compile it.
2. Run this program, observe the output, and make sure you understand what it is doing.
3. Set a breakpoint on the statement `bst.add(value)` in the `main` method.
4. Start the debugger and wait until execution is paused at the breakpoint.
5. Open a new Canvas window.
6. Add viewers for `values`, `value`, and `bst` to the canvas window.
7. Step in to the call to `add`.
8. Step over (single-step) the statements of the `add` method, observing their effect in the canvas window.

9. Repeat this process for each element in the `values` array.
10. Make sure you understand how the iterative `add` method works.

Adding values, recursive `put` method

1. Open `BstClient.java` in jGRASP and compile it.
2. Run this program, observe the output, and make sure you understand what it is doing.
3. Modify the statement `bst.add(value)` in the `main` method so that it uses the `put` method instead.
4. Set a breakpoint on this statement.
5. Start the debugger and wait until execution is paused at the breakpoint.
6. Open a new Canvas window.
7. Add viewers for `values`, `value`, and `bst` to the canvas window.
8. Step in to the call to `put`.
9. Using step-over and step-in as needed, explore the execution of both `put` methods, observing their effect in the canvas window.
10. Repeat this process for each element in the `values` array.
11. Make sure you understand how the recursive `put` method works.

Submission

The submission page for this activity asks you to apply your understanding of binary search trees to a problem and then submit it for a grade.