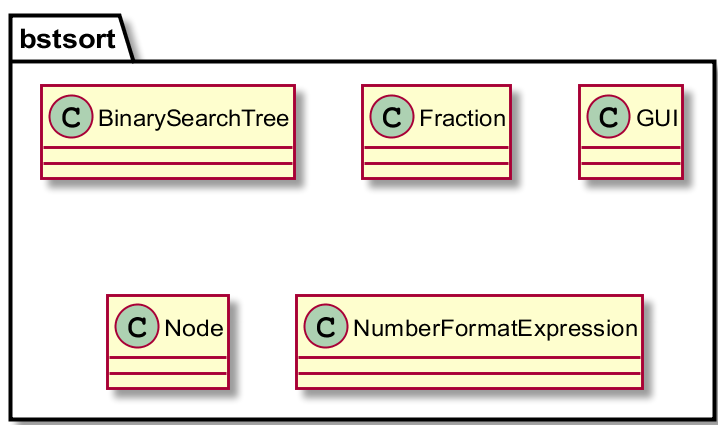
CMSC 350 – Project 3: Binary Search Tree Sort

UML for written classes: 1

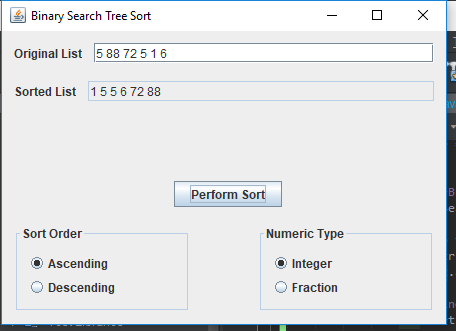
Test Plan: 2 – 8

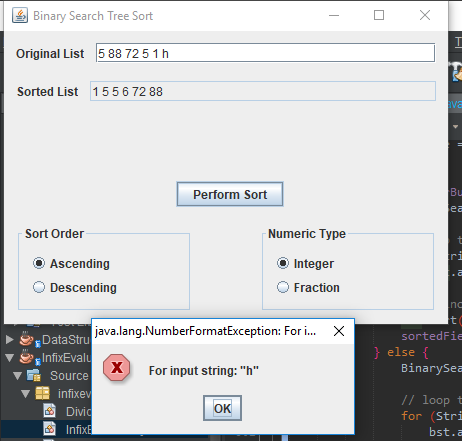
Lessons Learned: 9

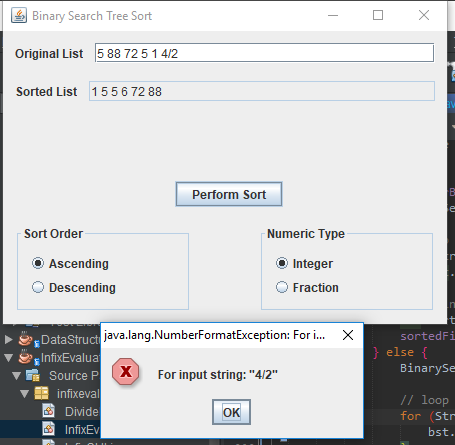
1. **UML**



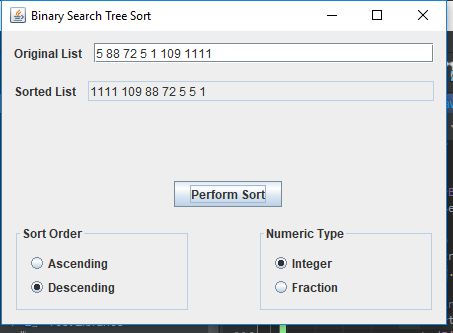
1. **Test Plan**
2. **Ascending Integers:**

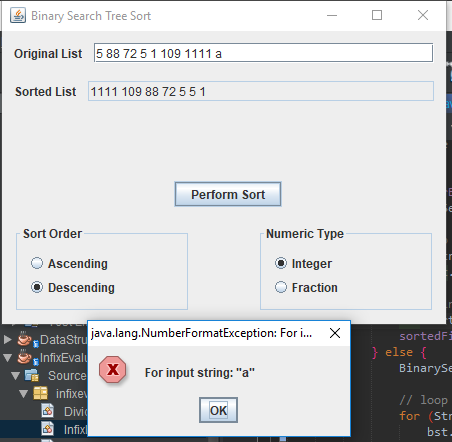




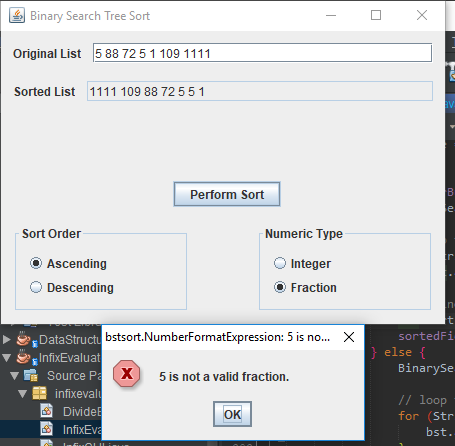


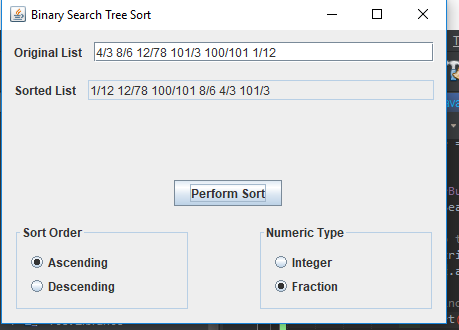
1. **Descending Integers:**



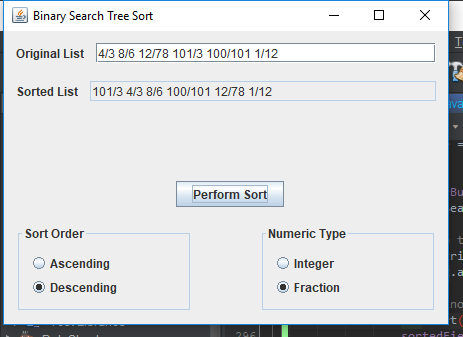


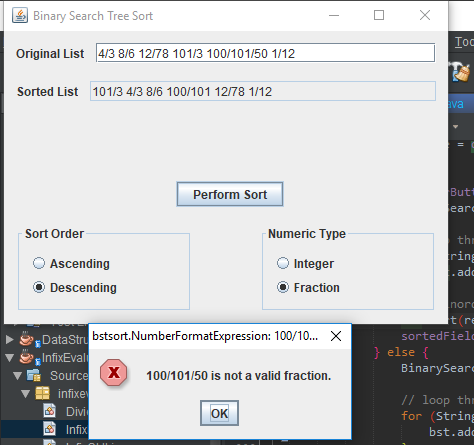
1. **Ascending Fractions:**

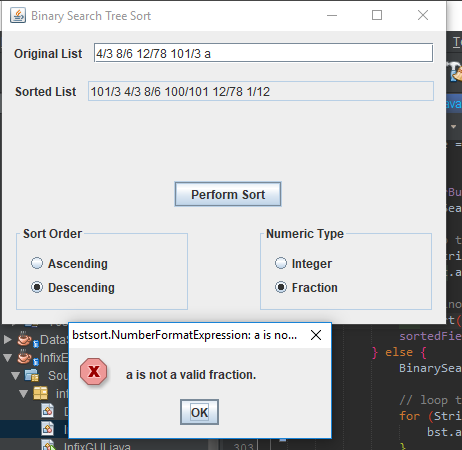


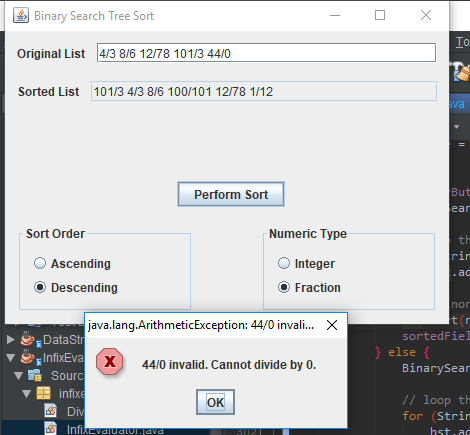


1. **Descending Fractions:**









1. **Lessons Learned**

The lesson learned with this project was central to the Binary Search Tree data structure itself. This really challenged my mastery of applying recursion which is used both for the add/insert method and the traversal methods which are used for sorting. The traversal was similar to node traversal done previously. However, the add method was a bit more of a challenge for me (as you can probably tell from the detailed comments I left for it.) In short, it compares passed ‘value’ to the passed ‘node’ value and sets the passed ‘node’ left/right based on the comparison. This continues until the base case of the passed node being null is reached – which leads to a new node being created with the value passed and the node being returned.

Additionally, this was good practice using generics. Especially since we had to make our own fraction class. Being able to use the Binary Search Tree on both integers and fractions was a positive learning experience both in the knowledge gained and in seeing the benefit of using generics in practice.