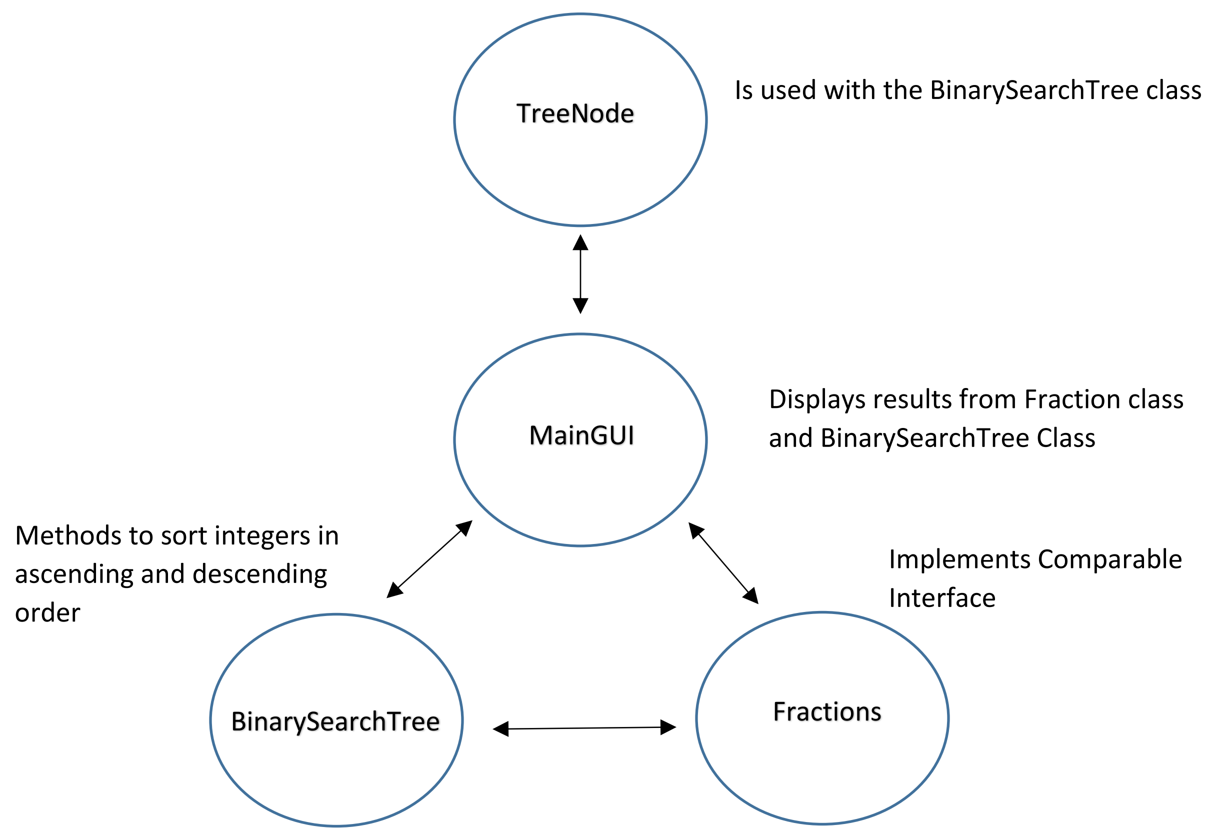
Anthony Borza

The below UML Diagram represents the java classes used throughout project 3:

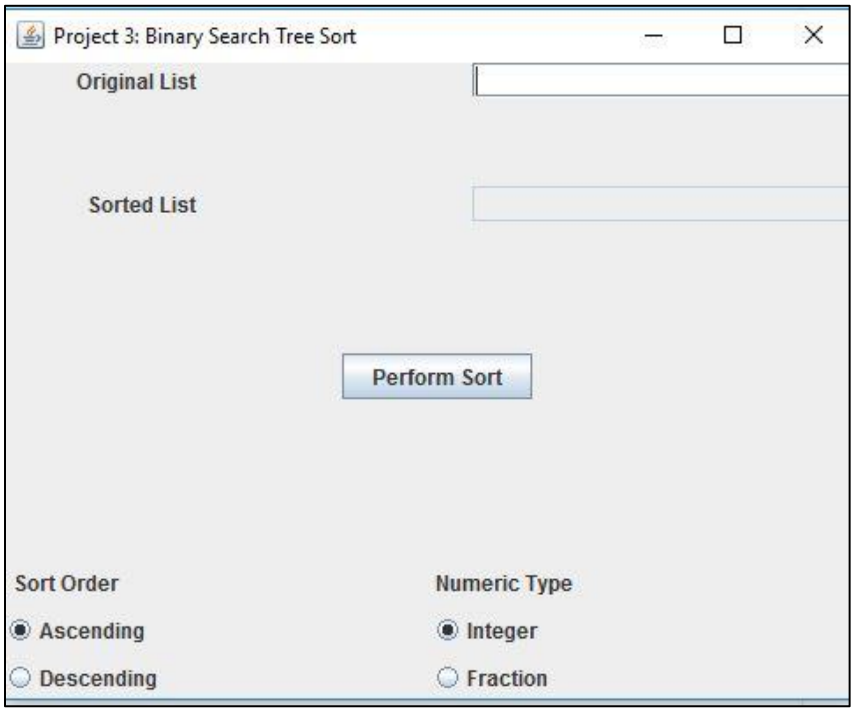
- TreeNode class
- MainGUI class
- BinarySearchTree class
- Fractions class

Project 3: Binary Search Tree Sort UML Diagram:


TreeNode
Is used with the BinarySearchTree class
MainGUI
Displays results from Fraction class and BinarySearchTree Class
Methods to sort integers in ascending and descending order
Implements Comparable Interface
BinarySearchTree
Fractions

Test Plans for Project 3:

GUI Display:


Project 3: Binary Search Tree Sort

Original List

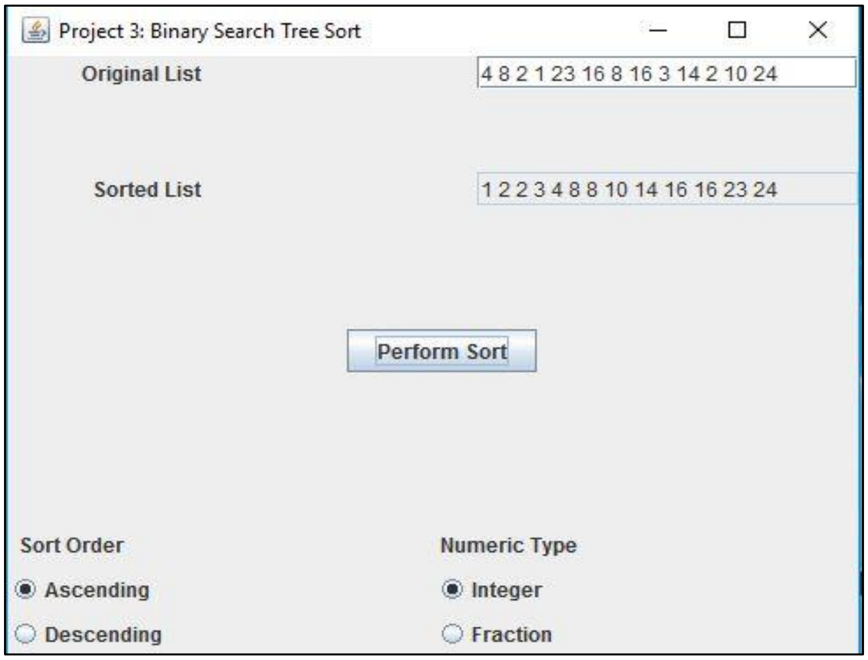
Sorted List

Sort Order: [● Ascending] ○ Descending

Numeric Type: [● Integer] ○ Fraction

Integers:

Test Case 1:

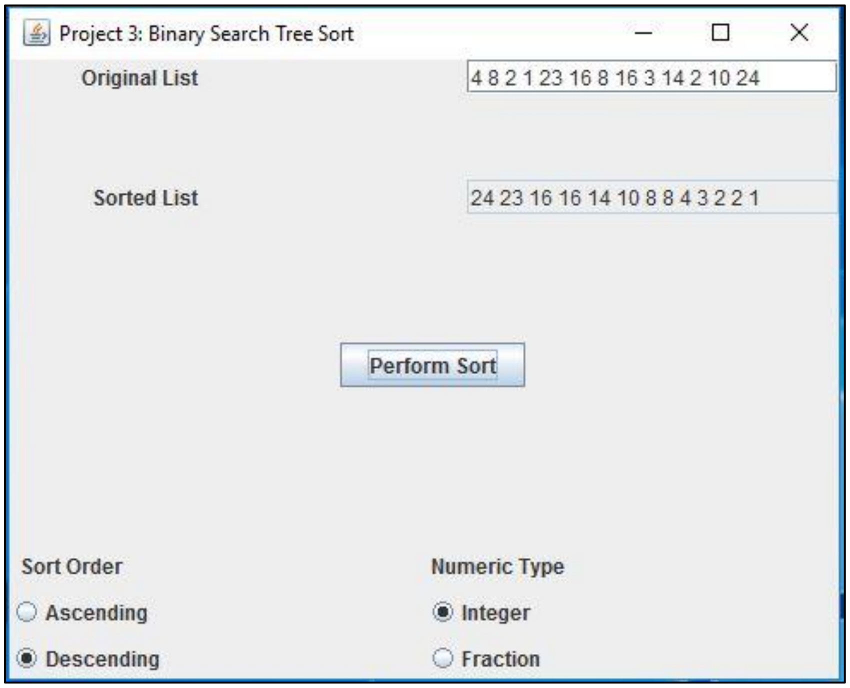

Project 3: Binary Search Tree Sort

Original List: 4 8 2 1 23 16 8 16 3 14 2 10 24

Sorted List: 1 2 2 3 4 8 8 10 14 16 16 23 24

Sort Order: [● Ascending] ○ Descending

Numeric Type: [● Integer] ○ Fraction


Project 3: Binary Search Tree Sort

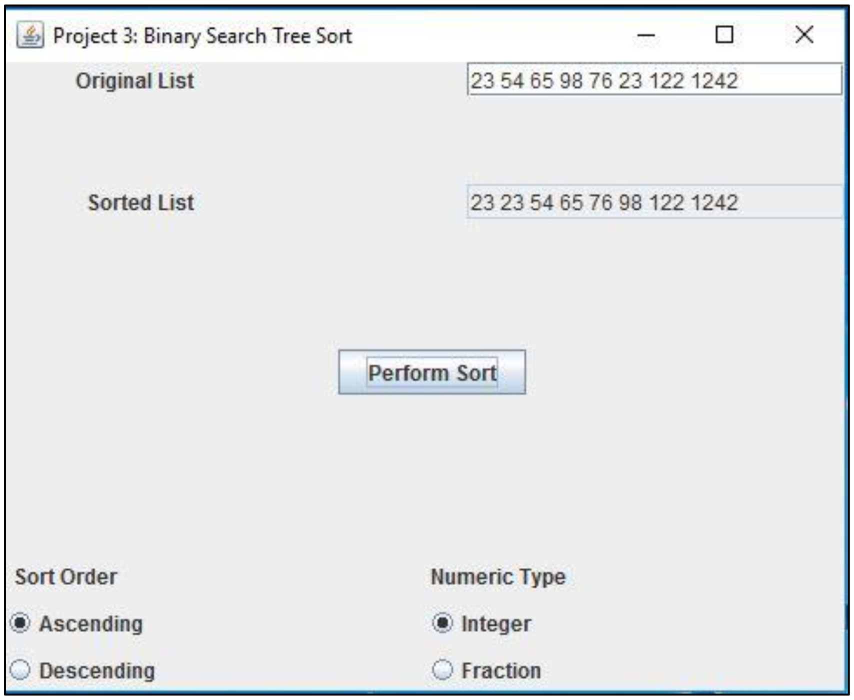
Original List: 4 8 2 1 23 16 8 16 3 14 2 10 24

Sorted List: 24 23 16 16 14 10 8 8 4 3 2 2 1

Sort Order: Ascending [✓ Descending]

Numeric Type: [✓ Integer] Fraction

Test Case 2:

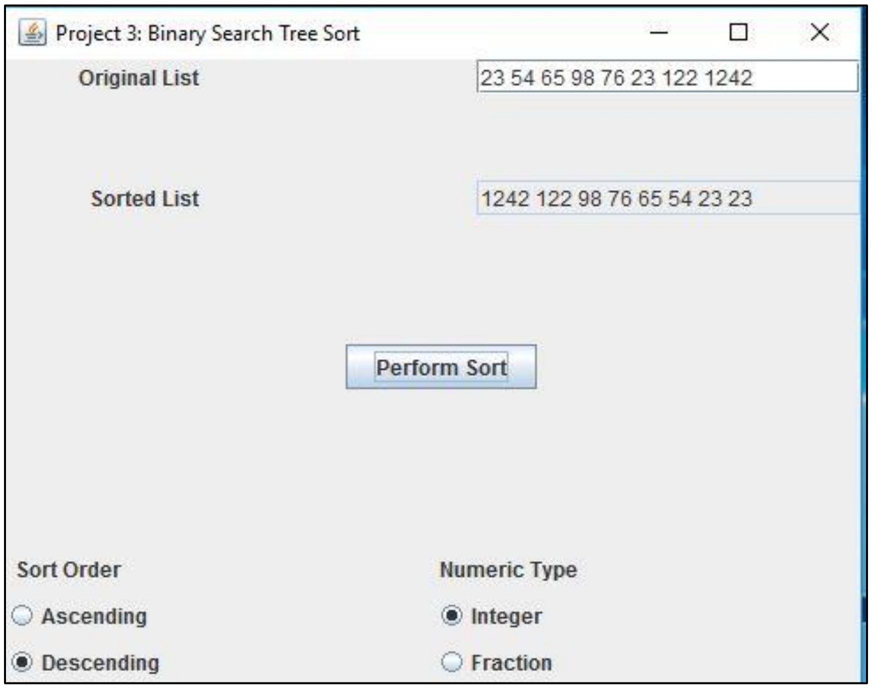

Project 3: Binary Search Tree Sort

Original List: 23 54 65 98 76 23 122 1242

Sorted List: 23 23 54 65 76 98 122 1242

Sort Order: [✓ Ascending] Descending

Numeric Type: [✓ Integer] Fraction


Project 3: Binary Search Tree Sort

Original List: 23 54 65 98 76 23 122 1242

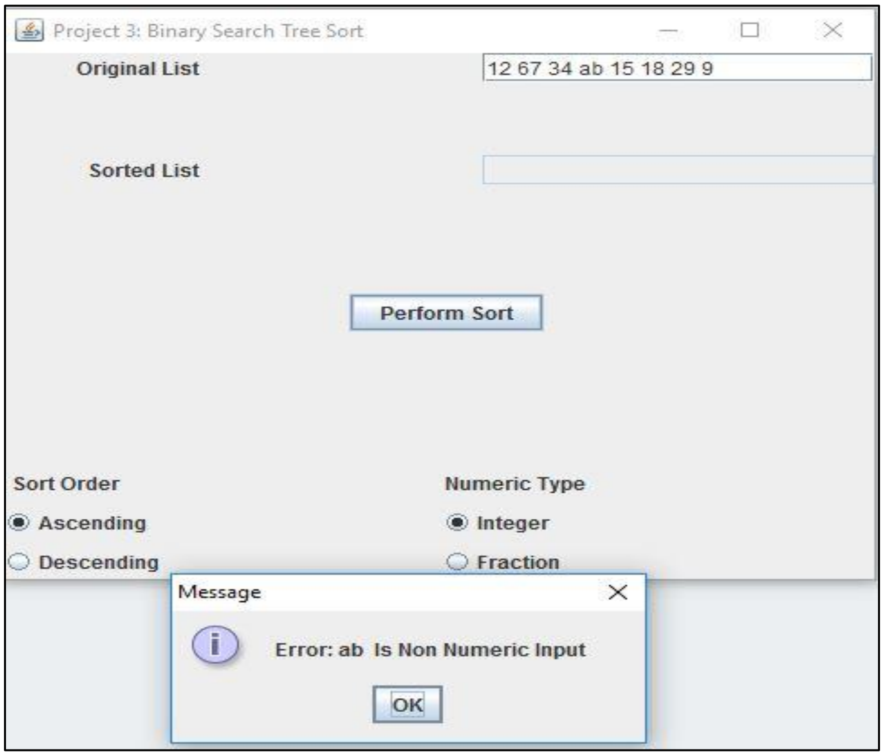
Sorted List: 1242 122 98 76 65 54 23 23

Perform Sort

Sort Order
( ) Ascending
(●) Descending

Numeric Type
(●) Integer
( ) Fraction

Error Checking for Integers:


Project 3: Binary Search Tree Sort

Original List: 12 67 34 ab 15 18 29 9

Sorted List:

Perform Sort

Sort Order
(●) Ascending
( ) Descending

Numeric Type
(●) Integer
( ) Fraction

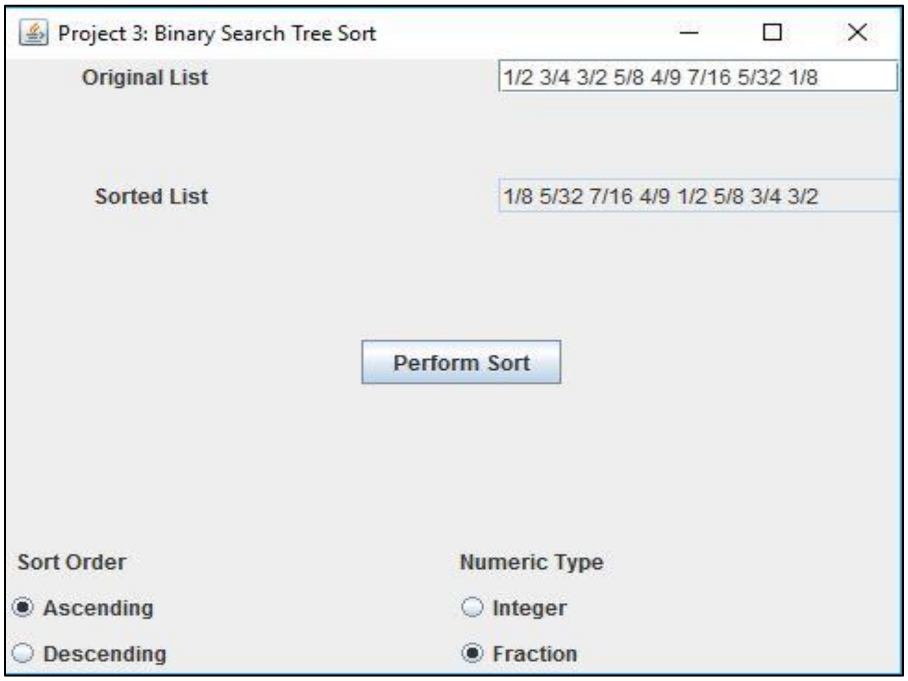
Message

Error: ab Is Non Numeric Input

OK

Fractions:

Test Case 1:



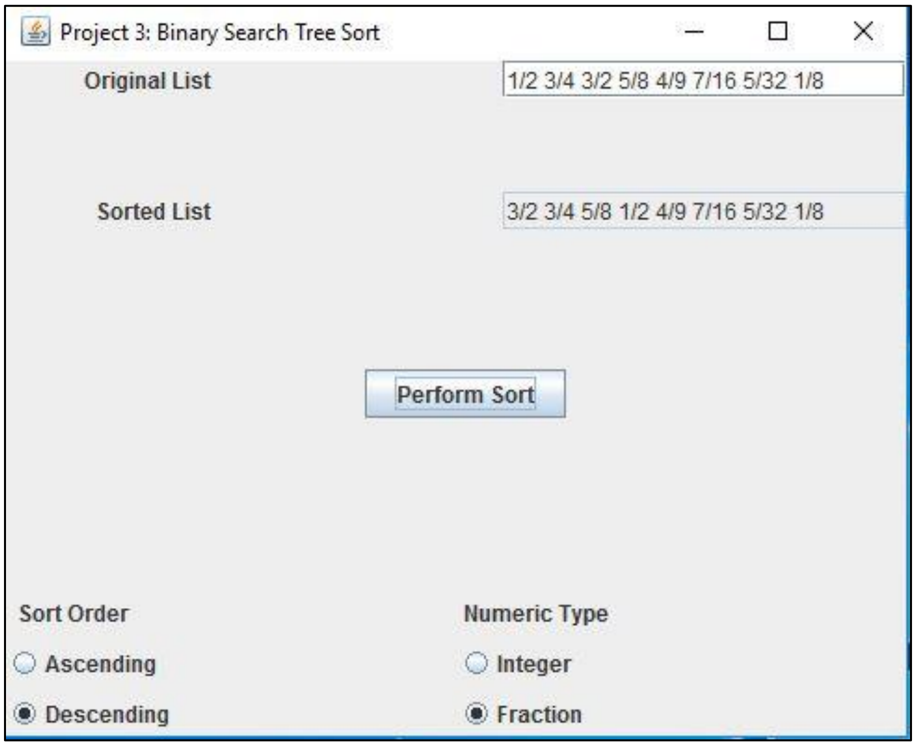
Project 3: Binary Search Tree Sort

Original List: 1/2 3/4 3/2 5/8 4/9 7/16 5/32 1/8

Sorted List: 1/8 5/32 7/16 4/9 1/2 5/8 3/4 3/2

Sort Order: [✓ Ascending] Descending

Numeric Type: Integer [✓ Fraction]



Project 3: Binary Search Tree Sort

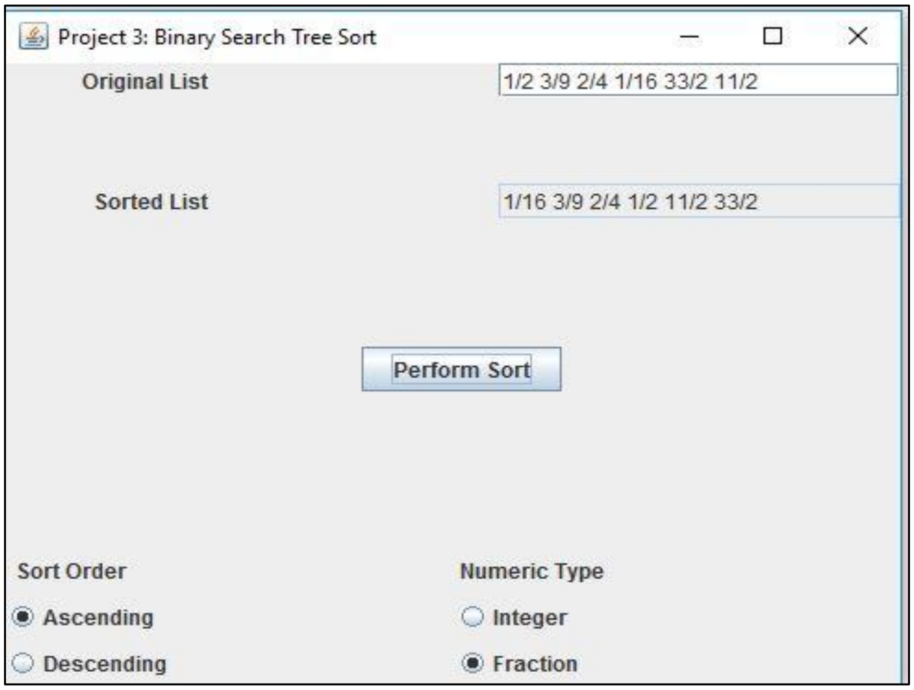
Original List: 1/2 3/4 3/2 5/8 4/9 7/16 5/32 1/8

Sorted List: 3/2 3/4 5/8 1/2 4/9 7/16 5/32 1/8

Sort Order: Ascending [✓ Descending]

Numeric Type: Integer [✓ Fraction]

Test Case 2:

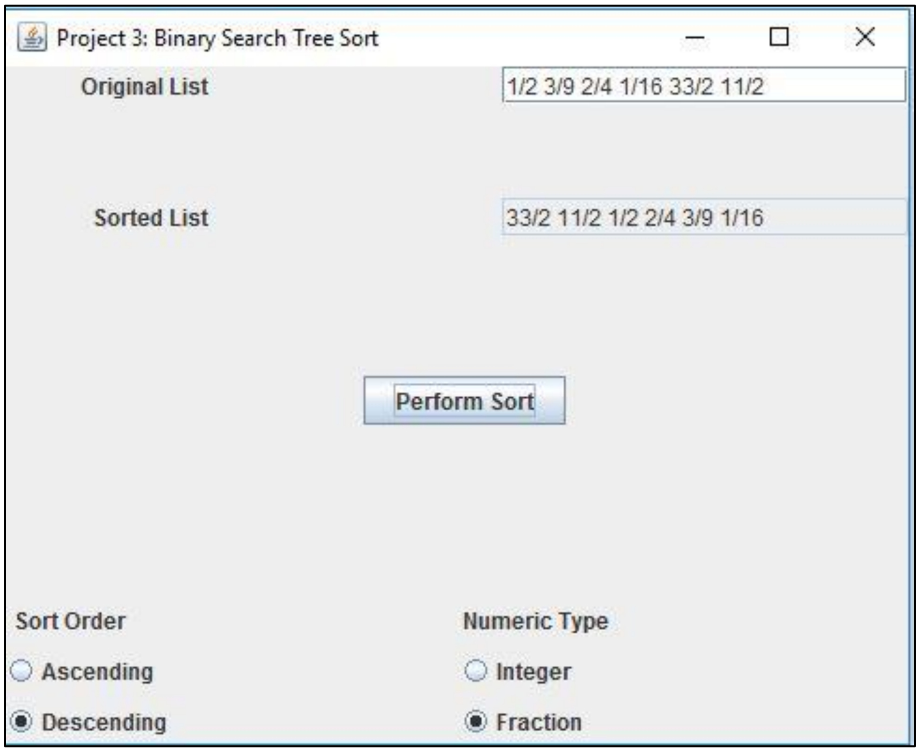

Project 3: Binary Search Tree Sort

Original List: 1/2 3/9 2/4 1/16 33/2 11/2

Sorted List: 1/16 3/9 2/4 1/2 11/2 33/2

Sort Order: [✓ Ascending] Descending

Numeric Type: Integer [✓ Fraction]


Project 3: Binary Search Tree Sort

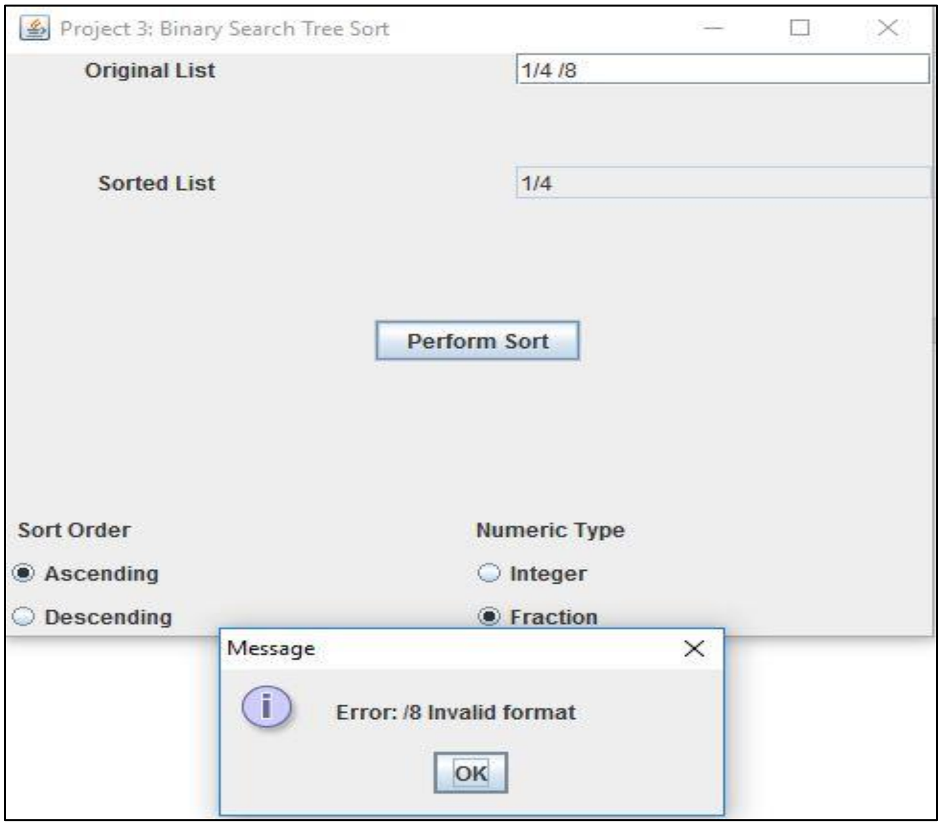
Original List: 1/2 3/9 2/4 1/16 33/2 11/2

Sorted List: 33/2 11/2 1/2 2/4 3/9 1/16

Sort Order: Ascending [✓ Descending]

Numeric Type: Integer [✓ Fraction]

Error Checking for Fractions:



Original List 1/4 /8

Sorted List 1/4

Sort Order
● Ascending
○ Descending

Numeric Type
○ Integer
● Fraction

Message

Error: /8 Invalid format

<u>**Lessons Learned:**</u>

After completing this project there are many things that were learned.   realize how important Binary Search Trees are to data structures. I learned how the tree insert method works, how the inorder method works, and how the descending order method works for sorting integers in ascending and descending order. I also learned how to sort fractions in ascending and descending order by using the comparable interface java has to offer. What is most important, like in previous projects, understanding what the problem is asking, and how to implement a solution to solve the problem only comes through trial and error. The hardest part of this project was getting the fractions to sort in ascending and descending order. It took me about a day and a half to figure out how to do it. The easiest part once everything else was working properly was developing the GUI and calling the classes. All in all, I spent a good portion of the week working on this project.