## 

## Project Description

In this program, I created an implementation of a Binary Search Tree and a command line interface to interact with it.

## Project Specifications

The requirement for this project was to create a program that would store integer values in a search tree, as well as add, delete, and traverse the values.

## Testing Methodology

To create this program, I used many interfaces together. This allowed me to implement very complex behavior as a series of simple, easily testable steps. The BinarySeachTree class was the most advanced, but used many methods from the BinaryTree class it extended to simplify common operations. I also implemented features one by one in the order they were needed, for example I implemented the add() method of the BST first because you have to add values to the tree before you can delete or traverse them. This helped me focus on smaller tasks and not get slowed down from bugs from unnecessary code that didn’t even work yet.

## Lessons Learned

From this assignment, I learned how to use several interfaces and subclasses together to create an advanced piece of software. I was able to add, delete, and traverse trees. However, I wasn’t able to get my code working for finding predecessors. I understand in theory how to find the correct node before another given node, but my recursive implementation of the method unfortunately doesn’t work yet. This taught me that just knowing something in theory isn’t enough to get your assignment done - you have to actually know how to implement it! The last lesson I learned was how to use one-off helper classes like NodePair to return a set of values rather than a single value. Before this, to return two nodes in a pair, I might have created a BinaryNode array of size 2, but the helper class is easier to understand and debug.