My Name: Lin Jia
Uid: u1091732

My Partner Name: Pingchuan Ma
Uid: u0805309

When you are satisfied that your program is correct, write a brief analysis document. The analysis document is 25% of your Assignment 8 grade. Ensure that your analysis document addresses the following.

1. Who is your programming partner? Which of you submitted the source code of your program?
Pingchuan Ma is my partner.

2. Evaluate your programming partner. Do you plan to work with this person again?
She is perfect. She does a lot for the code.

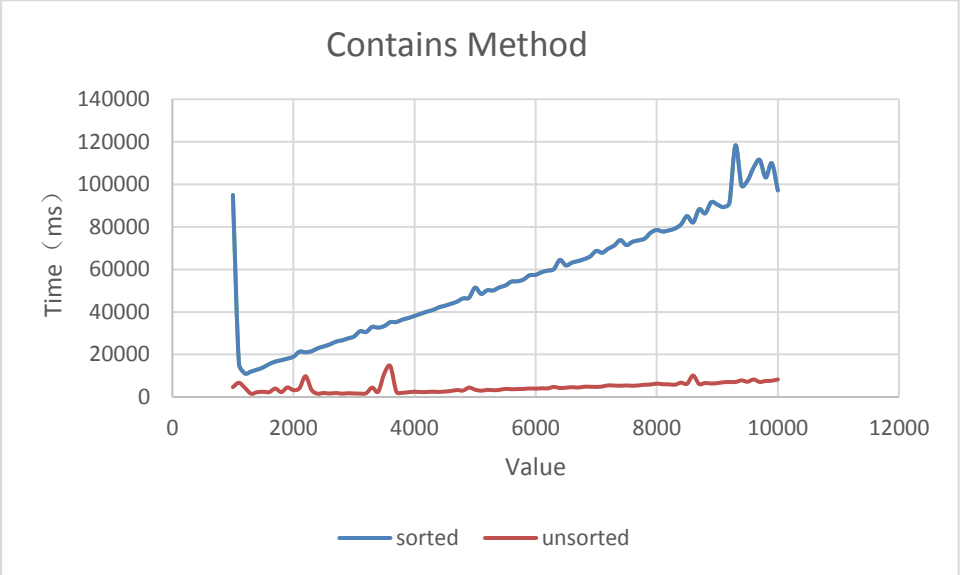
3. Design and conduct an experiment to illustrate the effect of building an N-item BST by inserting the N items in sorted order versus inserting the N items in a random order. Carefully describe your experiment, so that anyone reading this document could replicate your results. Submit any code required to conduct your experiment with the rest of your program and make sure that the code is well-commented. Plot the results of your experiment. Since the organization of your plot(s) is not specified here, the labels and titles of your plots(s), as well as, your interpretation of the plots is critical.

My experiment is like this:

1.Add N items to a BST in sorted order, then record the time required to invoke the contains method for the biggest item in the BST. for each N in the range [1000, 10000] stepping by 100.
Because I add N items by sorted order, the BST likes a linked list. I got O(n).

2.Add the same N items to a new BST in a random order, then record the time required to invoke the contains method for the biggest item in the new BST. for each N in the range [1000, 10000] stepping by 100.
Because I add N items by random order, the BST is an unbalanced tree. When I call contains, it will search from root one side. I got O(logn).

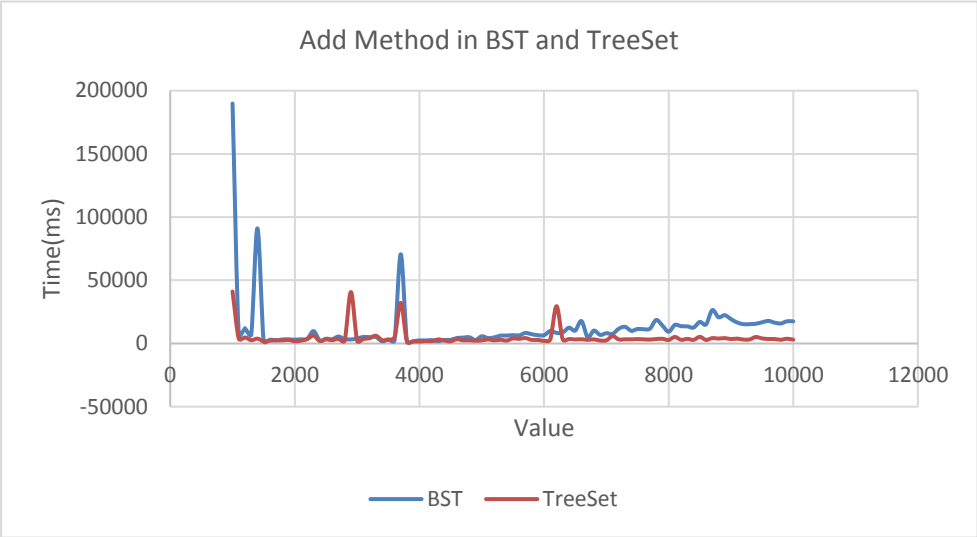
So according to the graph and analysis, building a BST by inserting random order is more efficient than in sorted order.



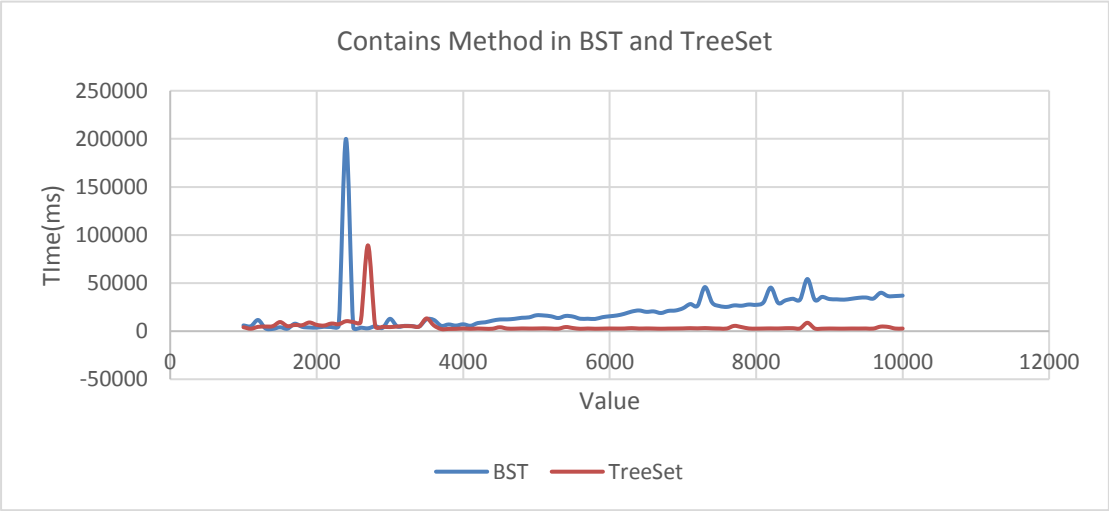
4. Design and conduct an experiment to illustrate the differing performance in a BST with a balance requirement and a BST that is allowed to be unbalanced. Use Java's TreeSet (http://docs.oracle.com/javase/7/docs/api/java/util/TreeSet.html) as an example of the former and your BinarySearchTree as an example of the latter. Java's TreeSet is an implementation of a BST which automatically re-balances I tself when necessary. Your BinarSearchTree class is not required to do this. Carefully describe your experiment, so that anyone reading this document could replicate your results. Submit any code required to conduct your experiment with the rest of your program and make sure that the code is well-commented. Plot the results of your experiment. Since the organization of your plot(s) is not specified here, the labels and titles of your plots(s), as well as, your interpretation of the plots is critical.

My experiment is like this:

1. Add N items to a TreeSet in a random order. Add the same N items (in the same random order) to a BinarySearchTree. For each N in the range [1000, 10000] stepping by 100.And add the same N+1 item into TreeSet as well as BST and then record the time separately.
   TreeSet is an implementation of BST which can re- balance itself automatically, so it costs more time than BST for add method. They are both O(logn).



2. Add N items to a TreeSet in a random order. Add the same N items (in the same random order) to a BinarySearchTree. For each N in the range [1000, 10000] stepping by 100. Record the time required to invoke the contains method for each item in the TreeSet. Record the time required to invoke the contains method for each item in the BinarySearchTree.
   TreeSet costs less time than BST for contains method because its re-balance itself automatically. They are both O(logn).



5. Many dictionaries are in alphabetical order. What problem will it create for a dictionary BST if it is constructed by inserting words in alphabetical order? Explain what you could do to fix the problem.

Dictionaries is usually in alphabetical order. But when we insert word into a dictionary BST by alphabetical order, it will get a right heavy unbalanced tree. For the searching, big-O will be O(n). So I will add the words from the middle one as the root, and then also add the middle one from the two separated side until the add all words. We will get a balance dictionary BST. That will easy to search. The big-O for searching will be O(logn).

6. How many hours did you spend on this assignment?