Binary Search Tree Using Java

Homework #7

By:

Pratyusha Thundena

CS 303L – L1A

October 17, 2018

### Problem Specification

The objectives included inserting elements in a Binary Search Tree (BST), utilizing in-order traversal of the BST to print elements in a sorted order, searching an element in BST, and using BST in a real application; such as, UPC.csv and input.dat.

### Program Design

This program required the following: UPC.csv, input.dat, Driver program(Lab7.java), io.File class, io.FileNotFoundException class, ArrayList class, and Scanner class.

The following steps were required to develop the program:

1. create insert () method, create inOrder () method, create Binary () method, create BST () method
2. create Binary Search () method
3. use the driver to read through the UPC.csv and input.dat file
4. utilize the search tree created to find the description linked to a given set of UPC keys
5. test the program for the given search keys
6. print out the description associated with the search keys
7. print out the total time it takes to complete the search

The following constructors and methods were defined within the class:

1. tree ()

Constructor that instantiates the object tree.

b) UPC ()

Constructor that parses a stringed argument as a signed decimal long and returns a long value.

c) ArrayList<Binary>searchList ()

Constructor that finds an element within a list.

d) Binary (long keys, String words)

Basic method that defines key as keys and data as words.

d) BST ()

Basic method that defines the root as null.

e) insert (Binary z)

Basic method that inserts a node having the value of key equals data in BST.

f) inOrder (Binary x)

Basic method that prints the elements of the BST in an in-order format.

g) Binary search (Binary x, long k)

Basic method that searches for a node having the value of key equals data in BST.

The println method of the System.out object was used to display the inputs and results for the driver program.

### Testing Plan

The test plan involved using UPC.csv and input.dat in order to search the program for the description associated with the given search keys and then recording the total time (in nanoseconds) it took to discover the descriptions.

|  |  |
| --- | --- |
| Total Time (in nanoseconds): | 242502 |
| Input | **Output** |
| 79,,INDIANA LOTTO 93,,treo 700w 123,,Wrsi Riversound cafe cd 161,,Dillons/Kroger Employee Coupon ($1.25 credit) 2140000070,,Rhinestone Watch 2140118461,,"""V"": Breakout/The Deception  VHS Tape" 2144209103,VHS,Tintorera - Tiger Shark 2144622711,,Taxi : The Collector's Edition VHS 2147483647,,Toshiba 2805 DVD player 2158242769,288/1.12Z,GREEN SUGAR COOKIES4276 2158561631,,HOT COCOA W/BKMK 2158769549,njhjhn,gjfhjbgkj 2160500567,2.25 oz (64)g,Dollar Bar Rich Raspberry 2172307284,,Mixed seasonal flower bouquet 2177000074,,4 way 13 AMP Extension Lead (Wilkinson UK) 2184000098,21 oz,Christopher's Assorted Fruit Jellies 2187682888,,fairway | **INDIANA LOTTO treo 700w Wrsi Riversound cafe cd Dillons/Kroger Employee Coupon ($1.25 credit) Rhinestone Watch """V"": Breakout/The Deception VHS Tape" Tintorera - Tiger Shark Taxi : The Collector's Edition VHS Toshiba 2805 DVD player GREEN SUGAR COOKIES4276 HOT COCOA W/BKMK**  **gjfhjbgkj Dollar Bar Rich Raspberry Mixed seasonal flower bouquet 4 way 13 AMP Extension Lead (Wilkinson UK) Christopher's Assorted Fruit Jellies fairway** |

### Results

### Analysis and Conclusions

In this program, the input file UPC.csv provided a vast list of keys with corresponding descriptions. The comma-separated input.dat file contained the various search keys that needed to be found. The results section displayed the output of the description for each key mentioned in the input.dat file. A typical binary search tree will be used for three major operations including the following: inserting elements, deleting elements, looking up elements. The average-case time complexity for the search operation was O (log n). The worst-case time complexity for the search operation happened to be O(n); specifically, when the tree was unbalanced. For the insertion operation, the root is usually examined and the new node is inserted recursively on either the left or right subtree. The time complexity for the average-case insertion was O (log n) and the time complexity for the worst-case insertion was O(n). In-order transversal method involved elements being retrieved by recursively reversing from left and right subtrees. The average time complexity for in-order traversal was O(n) because every node was visited.

### References

The parameters and input files (UPC.csv and input.dat) was provided in the homework assignment (by Dr. Bangalore) and Introduction to Algorithms (3rd ed.) was used to do the lab report.