Advanced Data Structures -Project Report

Programming language –Java

Compiler –javac

Instructions to compile and execute the program:

1. Run make command to compile the program. Name of the executable is bbst.
2. For test_100000000.txt file please provide jvm heap size parameter
   Eg:- java –Xmx8000m bbst test_100000000.txt <commands.txt >out_100000000.txt

Function Prototypes and Program Structure

Class bbst contains the following methods. Class bbst contains a private class Node.

Class bbst contains the following methods. Below are function prototypes with a short description for each.

Method summary:

1. public Node(int key, int val, boolean color)
   This method is present inside the private class Node. The method is a Node constructor that creates a node with key, value and color.

2. private boolean checkIfRed(Node a)
   The method checks if the color of a node is RED. It returns true if color is RED and false it is BLACK.

3. public boolean isEmpty()
   Method checks if root is equal to null. Returns true if root is null and false otherwise.

4. public int compare(int a, int b)
   Compares int a and int b. Returns 0 if they are equal. 1 if a >b and -1 if a<b.

5. public int get(int key)
   The method returns value if key is present in the tree.

6. private int get(Node a, int key)
   The method is called from public int get(int key) method. It returns the value present at the key if the key is present in the tree.

7. public int update(int key, int value)
   The method updates the value present at the key if the key is present in the tree.

8. private int update(Node a, int key,int value)
   The method updates the value present at the key if the key is present in the tree.

9. public boolean contains (int key)
   The method finds if a node is present in the tree and returns true if found .If not found it returns false.
10. public void add(int key, int val)
    The method adds a node to the red black tree.

11. private Node add(Node a, int key, int value)
    This method adds a node to the tree.It is called from public void add(int key, int val) method. New node created is red. Color flips and rotation restore the Red black tree property.

12. private Node rotateRight(Node a)
    Private method for right rotation. Callled during adding or deleting a node from the tree.

13. private Node rotateLeft(Node a)
    Private method to rotate left. Callled during adding or deleting a node from the tree.

14. private void flipColor(Node a)
    Private method to change color of node and left and right child.

15. public void delete (int key)
    Method to delete node from the tree if key is present in the tree.

16. private Node delete(Node a, int key)
    Method to delete a node from the tree if key is present in the tree. This method is called from public void delete (int key) method.

17. private Node turnRedLeft(Node a)
    This method implements 2 rotations and a color flip to rotate right-leaning red links on the search path.

18. private Node turnRedRight(Node a)
    This method implements 2 rotations and a color flip to rotate left-leaning red links on the search path.

19. private Node restore(Node a)
    This method Restores red black BST properties.

20. public int minimum()
    This method returns the minimum node in the tree.

21. private Node minimum(Node temp)
    This method returns the minimum node in the tree.

22. public void deleteMinimum()
    This method deletes the minimum element from the tree.

23. private Node deleteMinimum (Node a)
    This method deletes the minimum element from the tree.

24. public Iterable<Integer> keys(int lo, int hi)
    Returns all nodes with key in the range lo to hi

25. private void keys(Node a, Queue<Integer> queue, int lo, int hi)
    Returns all nodes with key in the range lo to hi in the subtree rooted at Node a.

26. public int Previous(int key)
    This method returns the value at the node with greatest key less than the given key.

27. private Node Previous(Node a, int key)
    This method returns the value at the node with greatest key less than the given key in the subtree rooted at Node a.

28. public int Next(int key)
    This method returns the value at the node with smallest key greater than the given key.

29. private Node Next(Node a, int key)
    This method returns the value at the node with smallest key greater than the given key in the subtree rooted at Node a.

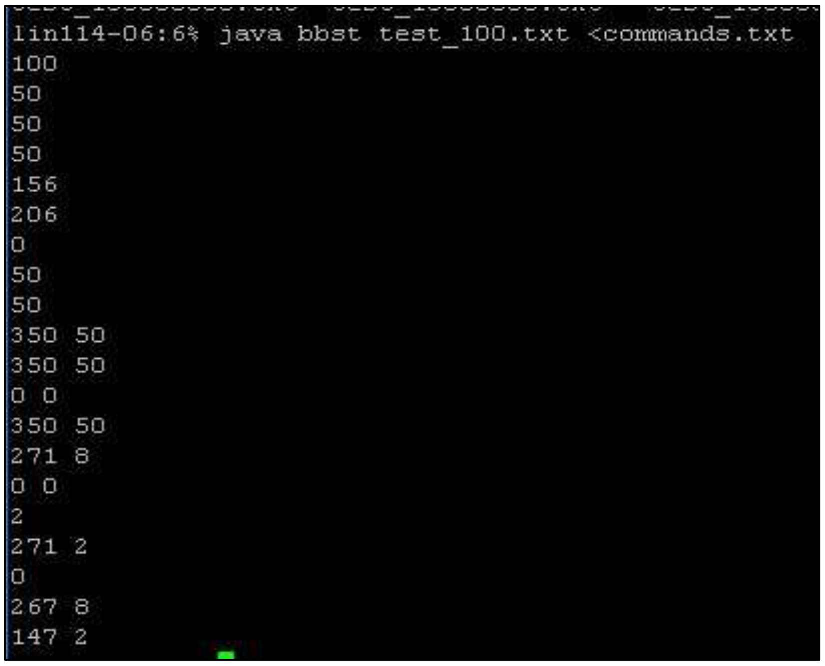
30. public void BuildTree(int EventIds[], int CountIds[], int EvenT)
    This method builds Redblack BST from a sorted array.

31. private Node BuildTree(int EventIds[], int CountIds[], int start, int end, double n)
    This method builds Redblack BST from a sorted array. All nodes except leaves are colored BLACK. Leaves are colored RED.

32. public static void main(String[] args)
    Program reads file name from command line arguments and scanner reads commands from stdin and executes and displays the output to console.

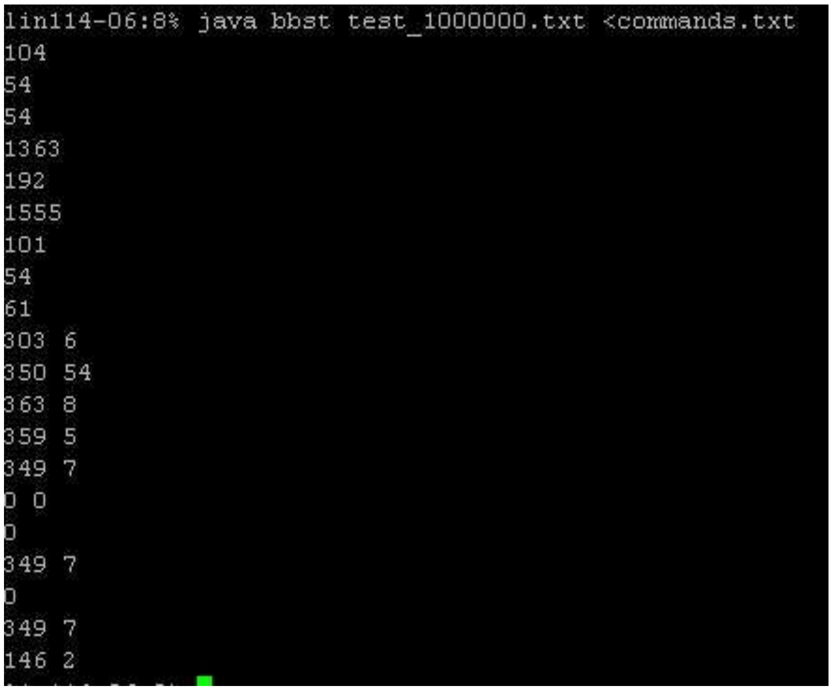
Output for test files

1. test_100.txt



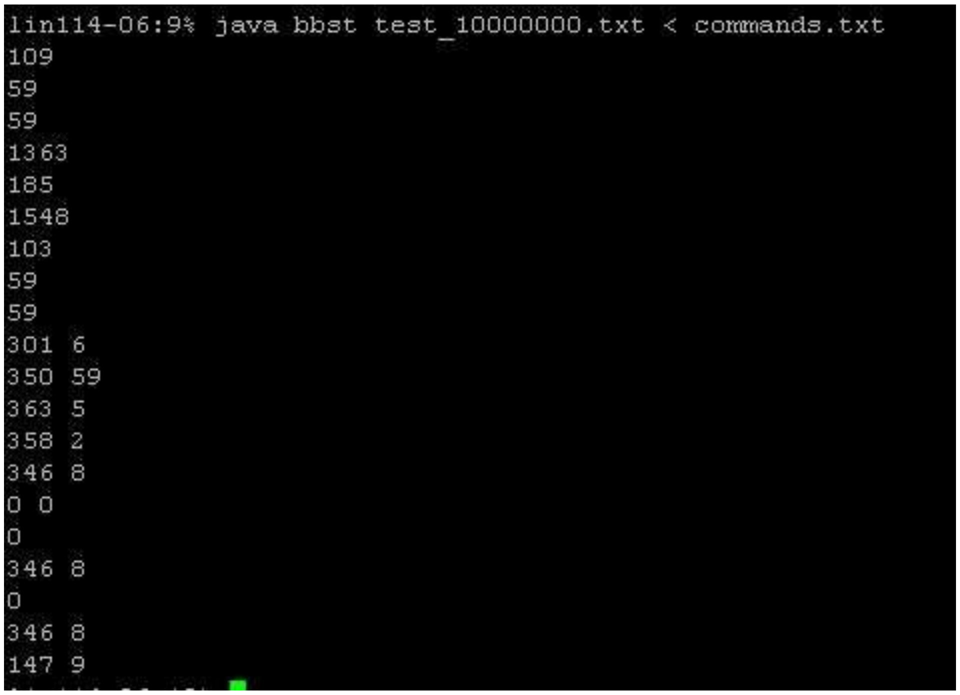
```
lin114-06:6% java bbst test_100.txt <commands.txt
100
50
50
50
156
206
0
50
50
350 50
350 50
0 0
350 50
271 8
0 0
2
271 2
0
267 8
147 2
```

2. test_1000000.txt

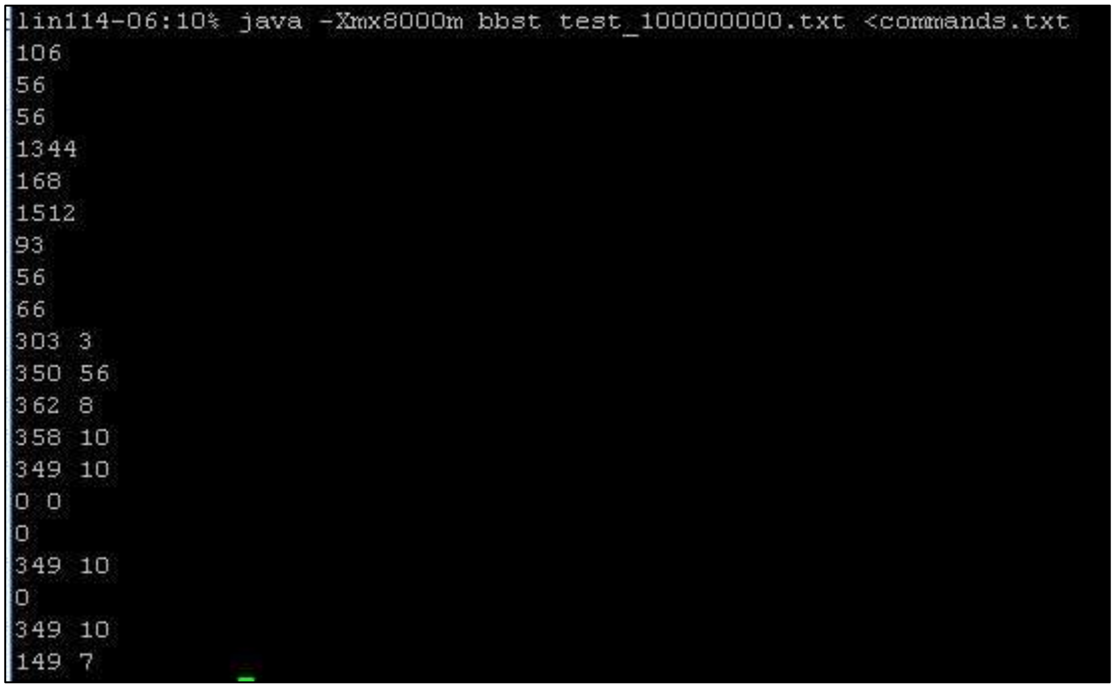


```
lin114-06:8% java bbst test_1000000.txt <commands.txt
104
54
54
1363
192
1555
101
54
61
303 6
350 54
363 8
359 5
349 7
0 0
0
349 7
0
349 7
146 2
```

3. test_10000000.txt


```
lin114-06:9% java bbst test_10000000.txt < commands.txt
109
59
59
1363
185
1548
103
59
59
301 6
350 59
363 5
358 2
346 8
0 0
0
346 8
0
346 8
147 9
```

4. test_100000000.txt


```
lin114-06:10% java -Xmx8000m bbst test_100000000.txt <commands.txt
106
56
56
1344
168
1512
93
56
66
303 3
350 56
362 8
358 10
349 10
0 0
0
349 10
0
349 10
149 7
```

References: -

Algorithms, 4Th Edition –Robert Sedgewick, Kevin Wayne. (http://algs4.cs.princeton.edu/home/).

Left Leaning Red Black BST – Robert Sedgewick
(https://www.cs.princeton.edu/~rs/talks/LLRB/RedBlack.pdf)