CS 1083

Module 11 Assignment

By Ngoc Phuong Anh Nguyen - 3712361

August 04th 2021

Source Code:

BinaryTreeNode.java:

```
public class BinaryTreeNode
{
    private int ticketId;
    private BinaryTreeNode left, right;

    public BinaryTreeNode(int ticketIdIn)
    {
        ticketId  = ticketIdIn;
        left  = null;
        right = null;
    }

    public void setTicketId(int ticketIdIn)
    {
        ticketId = ticketIdIn;
    }

    public int getTicketId()
    {
        return ticketId;
    }

    public BinaryTreeNode getLeft()
```

```
{
     return left;
}

public BinaryTreeNode getRight()
{
     return right;
}

/**
 * Display the subtree rooted at this
 * @param indent The indentation of the output.
*/
public void displayPreOrder(String indent)
{
     System.out.println(indent + ticketId);
     if (left != null)
          left.displayPreOrder("    " + indent);
     if (right != null)
          right.displayPreOrder("    " + indent);
}

/**
 * Inserts a given String into the subtree rooted at this. Maintains the
 * basic property of a binary tree, which is that all smaller values go
 * into the left subtree, and all larger values go into the right subtree.
```

```
      * @param ticketIdIn The id of the ticket.
      */
     public void insert(int ticketIdIn)
     {
          if (ticketIdIn < ticketId)
          {
               if (left == null)
                    left = new BinaryTreeNode(ticketIdIn);
               else
                    left.insert(ticketIdIn);
          }
          else if(ticketIdIn > ticketId)
          {
               if (right == null)
                    right = new BinaryTreeNode(ticketIdIn);
               else
                    right.insert(ticketIdIn);
          }
     } // end insert method
} // end class
```

BinaryTree.java:

```
public class BinaryTree
{
     private BinaryTreeNode root;
```

```
public BinaryTree()
{
     root = null;
}

/**
 * Insert the root of the binary tree.
 * @param ticketIdIn The ticket id.
 * @throws IllegalArgumentException
 */
public void insert(int ticketIdIn) throws IllegalArgumentException
{
     if (root == null)
     {
          root = new BinaryTreeNode(ticketIdIn);
     }
     else
     {
          root.insert(ticketIdIn);
     }
}

public void displayPreOrder()
{
     System.out.println ("*** Beginning of preorder display ***");
```

```
          if (root != null)
               root.displayPreOrder("");
          System.out.println ("****** End of preorder display ******");
     }
}
```

DuplicateNumbers.java:

```
/**
 * This is the driver program.
 * @author Ngoc Phuong Anh Nguyen - 3712361
 */

public class DuplicateNumbers
{
    public static void main(String[] args)
    {
        int[] array = {500000, 200000, 100000, 300000, 700000, 600000, 800000};
        try
        {
            BinaryTree binaryTree = new BinaryTree();
            for(int i = 0; i < array.length; i++)
            {
                binaryTree.insert(array[i]);
            }
            binaryTree.displayPreOrder();
```

```
}
catch(Exception e)
{
    System.err.println(e);
}

System.out.println("\n***** Array of 20 numbers (100000-999999) *****\n");

int[] a = new int[20];

for(int i = 0; i < a.length; i++)
{
    a[i] = (int)(Math.random()*(999999 - 100000 + 1) + 100000);
}

a[2] = a[6];
a[12] = a[6];
a[18] = a[6];

int count = 0;
for(int i = 0; i < a.length; i++)
{
    System.out.print(a[i] + "\t");
    count++;
    if(count % 5 == 0)
    {
        System.out.println();
    }
}
count = 0;
System.out.println("\n***************** End of Array *****************\n");
System.out.println("Inserting Values in the Tree:");
BinaryTree binary = new BinaryTree();
for(int i = 0; i < a.length; i++)
{
    binary.insert(a[i]);
    int same = 0;
    for(int j = 0; j < i; j++)
    {
        if(a[i] == a[j])
        {
            System.out.println("Duplicate found: Number " + a[i]
                    + " is already in the tree.");
            same++;
            break;
        }
    }
    if(same == 0)
    {
        count++;
    }
}
```

```
        System.out.println("A total of " + count + " numbers were added.\n");
        binary.displayPreOrder();
    }
}
```

Output:

Case 1:

```
*** Beginning of preorder display ***
500000
    200000
        100000
        300000
    700000
        600000
        800000
****** End of preorder display ******

***** Array of 20 numbers (100000-999999) *****

947858    189796    693198    395376    303891
584297    693198    345842    328538    332868
155017    142562    693198    528055    873351
567417    775504    118898    693198    149141

**************** End of Array ****************

Inserting Values in the Tree:
Duplicate found: Number 693198 is already in the tree.
Duplicate found: Number 693198 is already in the tree.
Duplicate found: Number 693198 is already in the tree.
```

```
A total of 17 numbers were added.

*** Beginning of preorder display ***
947858
    189796
        155017
            142562
                118898
                149141
        693198
            395376
                303891
                    345842
                        328538
                            332868
                584297
                    528055
                        567417
            873351
                775504
****** End of preorder display ******
```

Case 2:

```
*** Beginning of preorder display ***
500000
    200000
        100000
        300000
    700000
        600000
        800000
****** End of preorder display ******

***** Array of 20 numbers (100000-999999) *****

220236    804487    544541    526065    692234
102969    544541    899358    629682    401025
312187    320420    544541    915414    982616
543421    584576    566893    544541    280412

***************** End of Array *****************

Inserting Values in the Tree:
Duplicate found: Number 544541 is already in the tree.
Duplicate found: Number 544541 is already in the tree.
Duplicate found: Number 544541 is already in the tree.
A total of 17 numbers were added.
```

```
*** Beginning of preorder display ***
220236
    102969
    804487
        544541
            526065
                401025
                    312187
                        280412
                        320420
                543421
            692234
                629682
                    584576
                        566893
        899358
            915414
                982616
****** End of preorder display ******
```