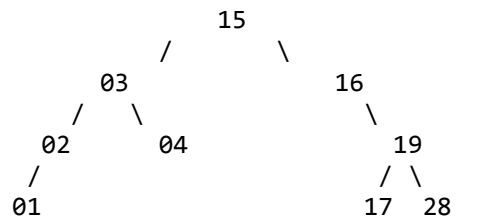
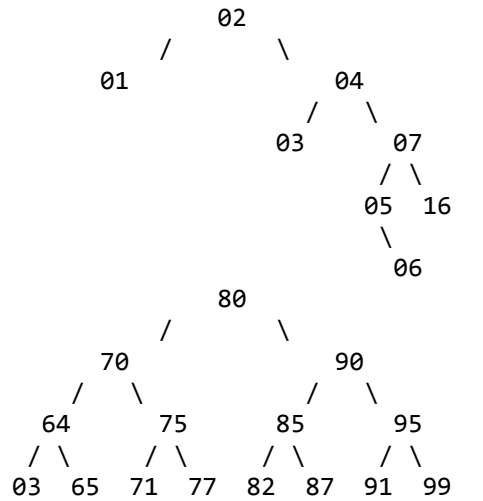
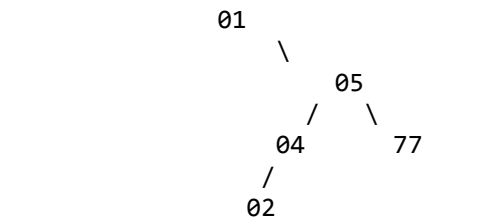
List of Classes:

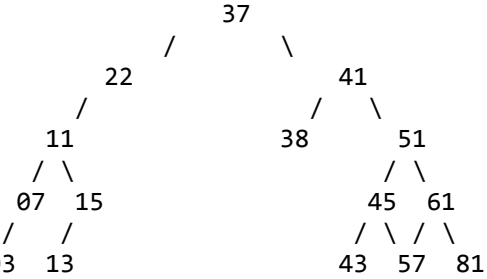
- BinaryNode.java
- BinaryTree.java
- BinaryTreeException.java
- Project2.java

Output:










```
package p2_Submit;
```

```java
/**
 * Project 2 - Plot the Binary Search Tree.
 */
public class BinaryNode{
	int element; //friendly data accessible by any class in the same package.
	BinaryNode left, right;
	/**
	 * Two Constructors.
	 */
	BinaryNode(int e){this(e, null, null);}

	BinaryNode(int e, BinaryNode ln, BinaryNode rn){
		element = e;
		left = ln;
		right = rn;
	}
}//class BinaryNode
```

```java
package p2_Submit;
/**
 * Project 2 - Plot the Binary Search Tree.
 */
public class BinaryTree{
	private BinaryNode root;
	/**
	 * Two Constructors
	 */
	public BinaryTree(){root = null;} //constructs root -> null;

	public BinaryTree(int x){
		root = new BinaryNode(x);
	}
	/**
	 * Access methods - getLeft, getRight & getRootObj.
	 */
	public BinaryTree getLeft() throws BinaryTreeException {
		if(root == null) throw new BinaryTreeException("Empty Tree.");
		else{
			BinaryTree t = new BinaryTree();
			t.root = root.left;
			return t;
		}
	}
	public BinaryTree getRight() throws BinaryTreeException {
		if(root == null) throw new BinaryTreeException("Empty Tree.");
		else{
			BinaryTree t = new BinaryTree();
			t.root = root.right;
			return t;
		}
```

```
	}
	public int getRootObj() throws BinaryTreeException{
		if(root == null) throw new BinaryTreeException("Empty Tree.");
		else return root.element;
	}
	/**
	 * Update methods - setLeft and setRight.
	 */
	public void setLeft(BinaryTree t) throws BinaryTreeException{
		if(root == null) throw new BinaryTreeException("Empty Tree.");
		else root.left = t.root;
	}
	public void setRight(BinaryTree t) throws BinaryTreeException{
		if(root == null) throw new BinaryTreeException("Empty Tree.");
		else root.right = t.root;
	}
	/**
	 * Other methods - isEmpty, inorder & insert.
	 */
	public boolean isEmpty(){return root == null;}

	public static void inorder(BinaryTree t) throws BinaryTreeException{
		if(!t.isEmpty()){
			inorder(t.getLeft());
			System.out.println(t.getRootObj());
			inorder(t.getRight());
		}
	}
	public static BinaryTree insert(BinaryTree t, int x){
		if (t.isEmpty())
			return new BinaryTree(x);
		else{
			if((t.getRootObj()) < x)
				t.setRight(insert(t.getRight(), x));
			else
				t.setLeft(insert(t.getLeft(), x));
			return t;
		}
	}
}
```

```
package p2_Submit;
/**
 * Project 2 - Plot the Binary Search Tree.
 */
public class BinaryTreeException extends RuntimeException{
	public BinaryTreeException(String err){
		super(err);
	}
}
```

```
package p2_Submit;
/**
 * Project 2 - Plot the Binary Search Tree.
 */
import java.io.*;
public class Project2 {
	public static void main(String[] args) {
		if(args.length == 0)System.out.println("No file specified.");
		else{
			FileReader theFile;
			BufferedReader inFile;
			String oneLine;
			try{//FileNotFoundException must be caught
				theFile = new FileReader(args[0]);
				inFile = new BufferedReader(theFile);
				/**
				 * now read the text file line by line.
				 */
				while ((oneLine = inFile.readLine()) != null){
					String numbers[] = oneLine.split(" ");
					BinaryTree temp1 = new BinaryTree();
					char [][] arr = new char [9][40];
					/**
					 * Assign every element of char Array to WhiteSpace.
					 */
					for(int i = 0; i <= 8; i++) {
						for(int j = 0; j <= 39; j++) {
							arr[i][j] = ' ';
						}
					}
					/**
					 * insert all the numbers in the line to the
					 		BinaryTree.
					 */
					for(int i = 0; i < numbers.length; i++) {
						temp1 = BinaryTree.insert(temp1,
									Integer.parseInt(numbers[i]))
									;
					}
					/**
					 * A BST is established. Plot the tree into char
					  	Array.
					 */
					plotToCharArr(temp1, arr, 0, 20, 0);
					//Print out the Char Array
					for(int i = 0; i <= 8; i++) {
						for(int j = 0; j <= 39; j++) {
						System.out.print(Character.toString(arr[i][j]))
					;
						}
```

```
                                    System.out.println();
                              }
                              //read another line.
                              inFile.readLine();
                        }//while
                  }//try
                  catch (Exception e){System.out.println(e);}
            }//else
      }//main
      public static void plotToCharArr(BinaryTree t, char[][] arr, int rowStart,
                                          int colStart, int reduceOffSet) {
            int spaceOffSet = 17;

            if(!(t.isEmpty())) {
                  /**
                   * Store the integer into char array;
                   *    access to the 1st digit by dividing 10;
                   *    access to the 2nd digit by modulus 10 since char only can
                        store 1 digit.
                   */
                  arr[rowStart][colStart] = (char)('0' + t.getRootObj()/10);
                  arr[rowStart][colStart + 1] =  (char)('0' + t.getRootObj()%10);
                  /**
                   * The line below is to designing the tree "branches" ("\" &
                          "/");
                   */
                  //each time of recursion, the offSet is decreasing by 1/2.
                  reduceOffSet = reduceOffSet + 2;
                  int nextLineOffSet = spaceOffSet/reduceOffSet;
                  /**
                   * Therefore, if the root has the left (or right) child
                   *    draw "/"(or "\") on next line at the proper location;
                   */
                  if(!(t.getLeft().isEmpty())) arr[rowStart+1][colStart -
                                          nextLineOffSet/2] = '/';
                  if(!(t.getRight().isEmpty())) arr[rowStart+1][colStart +
                                          nextLineOffSet/2] = '\\';
                  /**
                   * Plot the rest of the Binary nodes into char array recursively.
                   */
                  plotToCharArr(t.getLeft(), arr, rowStart+2, colStart -
                                    nextLineOffSet, reduceOffSet);
                  plotToCharArr(t.getRight(), arr, rowStart+2, colStart +
                                    nextLineOffSet, reduceOffSet);
            }
      }//plotToCharArr
}//Class
```