Lab Goal :

This lab was designed to teach you more about Binary Trees.

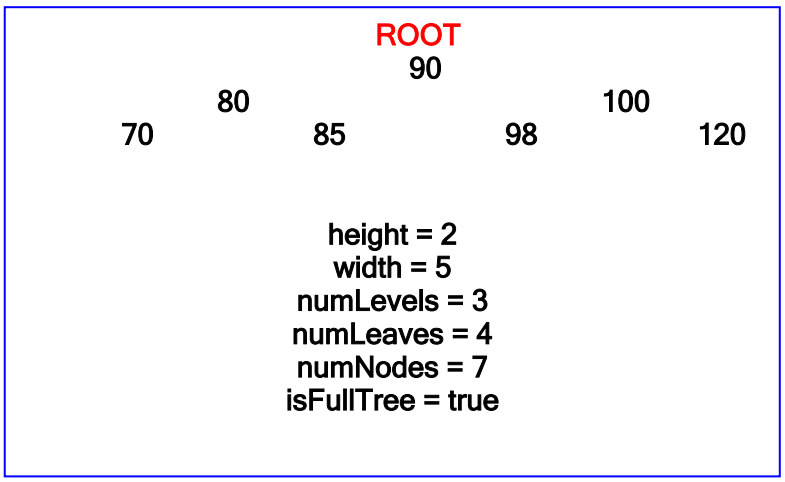
Lab Description :

Write a binary search tree class. You must write the following methods ::

inOrder()	this method will print the tree using a LDR traversal
preOrder()	this method will print the tree using a DLR traversal
postOrder()	this method will print the tree using a LRD traversal
revOrder()	this method will print the tree using a RDL traversal
toString()	returns a String that lists the values in their natural order
clear()	removes all nodes from the tree
isFull()	a boolean method that indicates whether this tree is full / perfect (i.e. every level is full)
search(value)	write a method to **search the tree for a value** and return true or false
getNumNodes()	returns the number of nodes in the tree
getNumLeaves()	returns the number of leaves in the tree
getNumLevels()	returns the number of levels in the tree
getHeight()	returns the height of the tree
getLargest()	returns the largest value
getSmallest ()	returns the smallest value
getWidth()	returns the maximum width / diameter of the tree
remove(value)	write a method to **remove** a node from the tree – must be recursive

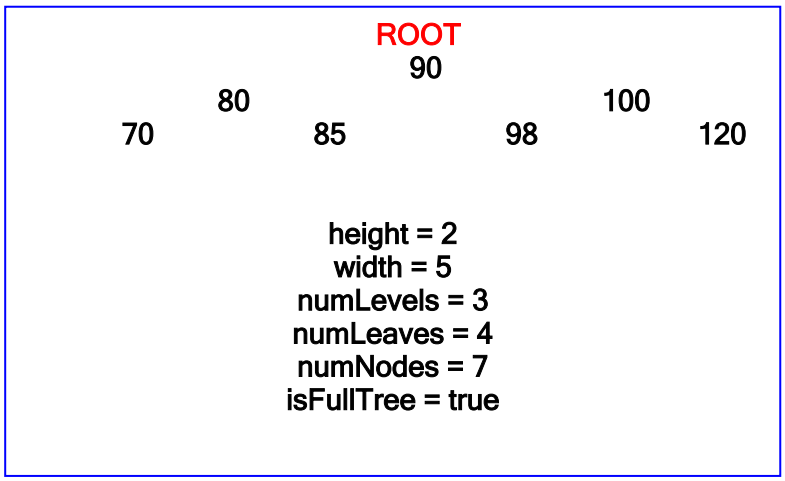
Sample Output :

```
IN ORDER
70 80 85 90 98 100 120

PRE ORDER
90 80 70 85 100 98 120

POST ORDER
70 85 80 98 120 100 90

REVERSE ORDER
120 100 98 90 85 80 70

Tree as a string   70  80  85  90  98  100  120
```



Continued on next page . . .

```
The tree is full.
The tree contains 100!
The tree does not contain 114!

Number of nodes is 7
Number of leaves is 4
Number of levels is 3
Tree height is 2
The largest tree node 120
The smallest tree node 70
Tree width is 5
```




```
Tree before removing any nodes (displayed here using level order traversal)
90 80 100 70 85 98 120


Tree after removing 90.
98 80 100 70 85 120


Tree after removing 70.
98 80 100 85 120


Tree after removing 85.
98 80 100 120


Tree after removing 98.
100 80 120


Tree after removing 80.
100 120


Tree after removing 120.
100


Tree after removing 100.
```