Planning Document

Activity - Virtual Lab Practice **Course Faculty: Dr Suresh Kumar K**

Course: Data Structures Using Python Lab **Class: III Sem IT**

Topic- Binary Search Tree

Learning Objective: To understand the different operations of binary search trees

Purpose

To develop an understanding of the Binary Search Tree, how elements are inserted, deleted and searched in a binary search tree. Similarly, the student will become familiar with the algorithm with the insertion, deletion, and search elements.

Concept of Binary Search Tree

A Binary Search Tree (BST) is a type of ***binary tree***, and each vertex has only up to 2 children. BST quickly allows us to maintain a sorted list of numbers where the nodes are arranged in a specific order. This is also called an ***ordered binary tree***.

BST property:

1. All nodes (vertices) in the left subtree of a vertex must have a value lesser than its own.
2. All nodes (vertices) in the right subtree of a vertex must have a value higher than its own.
3. Both the left subtree and right subtree must also be a binary search tree.

Example

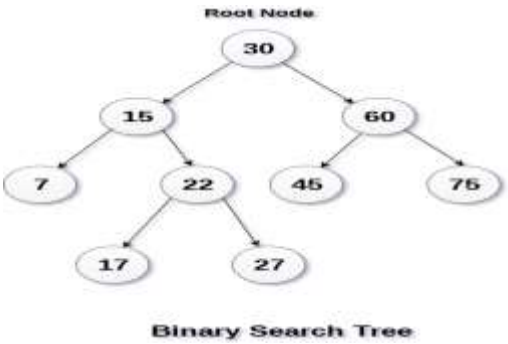


Figure 1a Examplebinary search tree

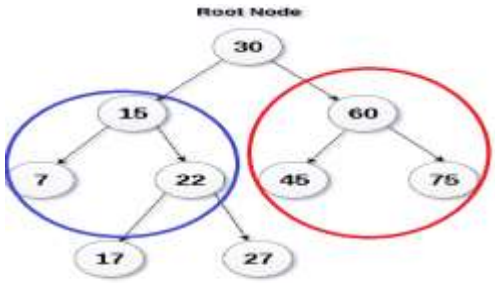


Figure 1b BST with left and right subtree

In figure 1a, the binary search has the root element as 30, and all its left subtree values are less than the root value, and all its right subtree values are greater than the root value. In figure 1b, both the left and right subtrees are also following the same property.

Pseudo Code

Insert/Create v

create new vertex

if the value to be inserted < this key

go left

else go right

Delete v

search for vertex v

if (v is a leaf)

delete leaf v

else if (v has one child)

bypass v

else replace v with the successor

Search v

if (this == null)

return null

else if (this key == search value)

return this

else if (this key < search value)

search right

else search left

Link to Open the virtual lab tool

Use the following link **https://visualgo.net/en/bst** to open the binary search tree simulator.

Steps to Create a Binary Search Tree

Step 1 Select ***Create option*** at left side menu and select ***Empty option*** to empty the old.

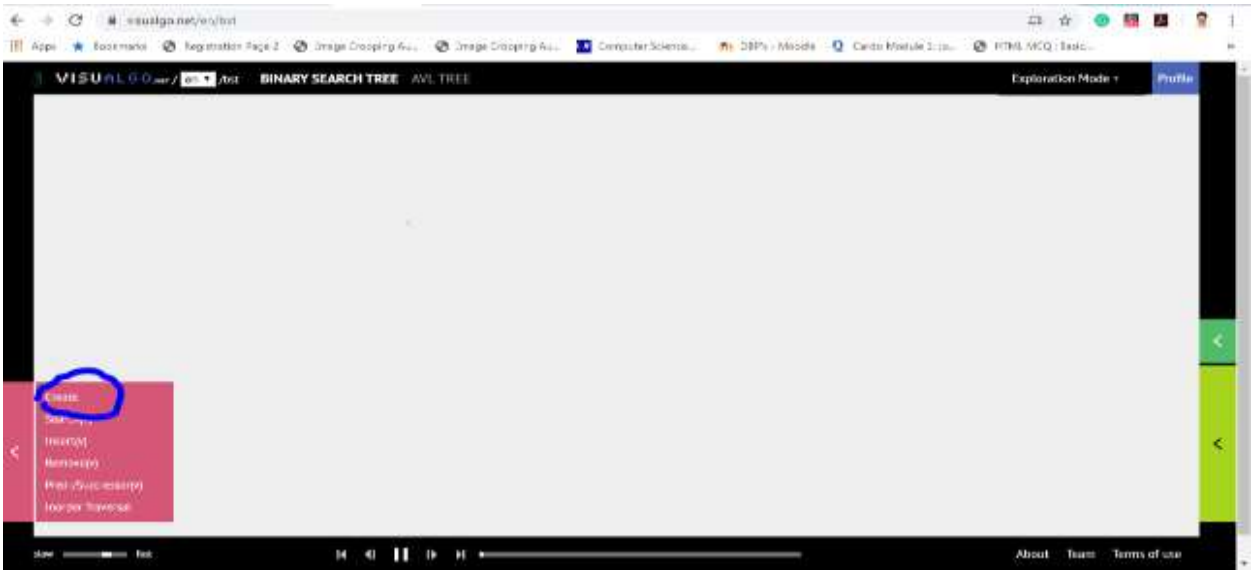


Figure 1 Creating a Binary search tree

Figure 2 Inserting first element 30

Step 2 Select ***insert option*** to insert further elements by using (,) comma as a delimiter (Example) 15,12,18,45,38,49 and press ***go button***.
Now the animator executes the pseudo code and decides where correctly the element need to be inserted.
For example, the binary search tree has 30 as the root element, and if 15 need to be inserted, the algorithm works as follows.
It checks for the root element < the element to be inserted if the condition satisfies the item will be entered at the right-hand side of the binary tree; otherwise, it will add at left. Here the state is 30 < 15, which is false so that the element will be inserted at the left side of the root.

Figure 4 After inserted the element 15

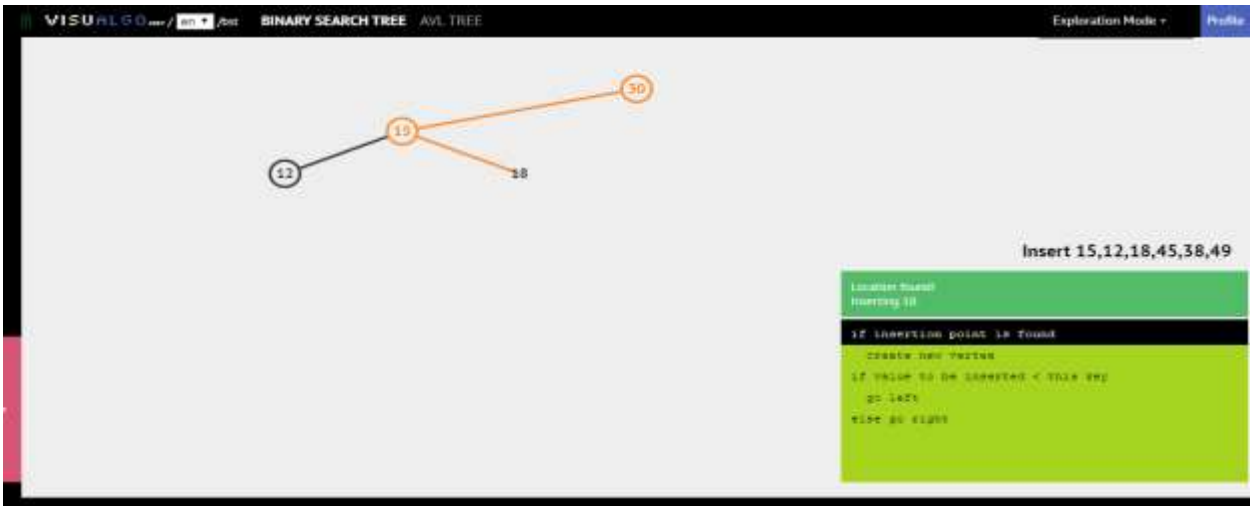


Figure 5 When the element 18 is inserted.

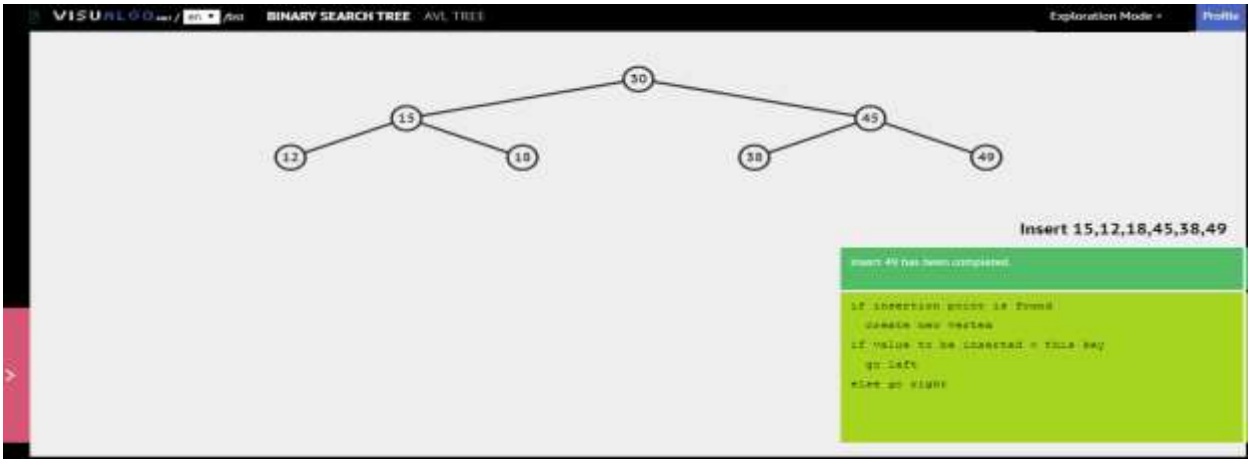


Figure 6 After the tree constructed

To Insert an Element in Binary Search Tree

Step 1 to Insert an element; select the ***Insert option*** in the menu.
Step 2 type the element name.
Step 3 press ***go button*** (Example to insert the element is 8)

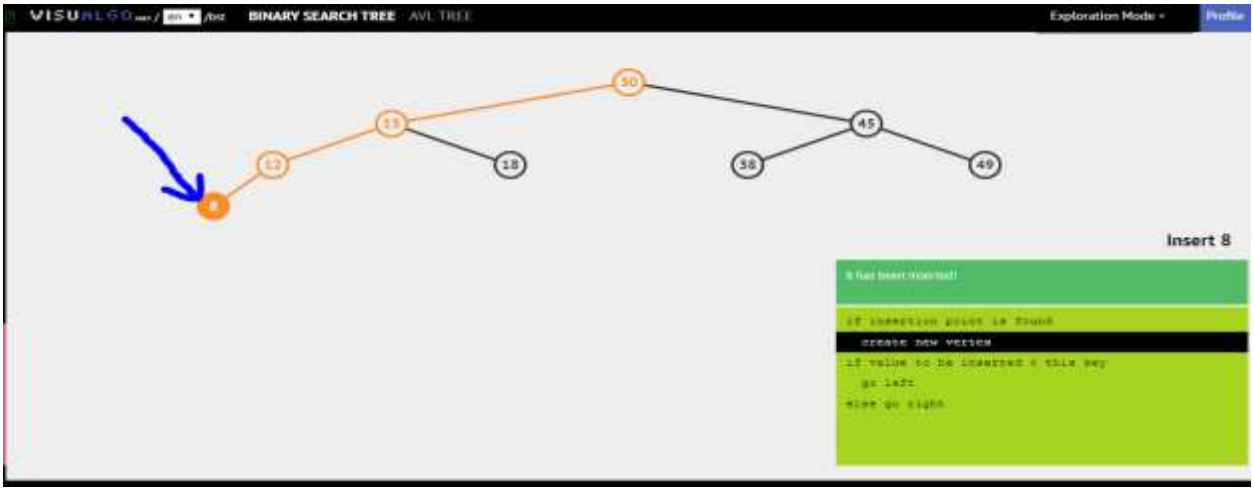


Figure 7 Inserting Element 8

To Delete an Element in Binary Search Tree

Step 1 to delete an element, select the ***remove option*** in the menu.
Step 2 Type the element name to remove.
Step 3 press the ***go button***
(Example to delete the element is 15)

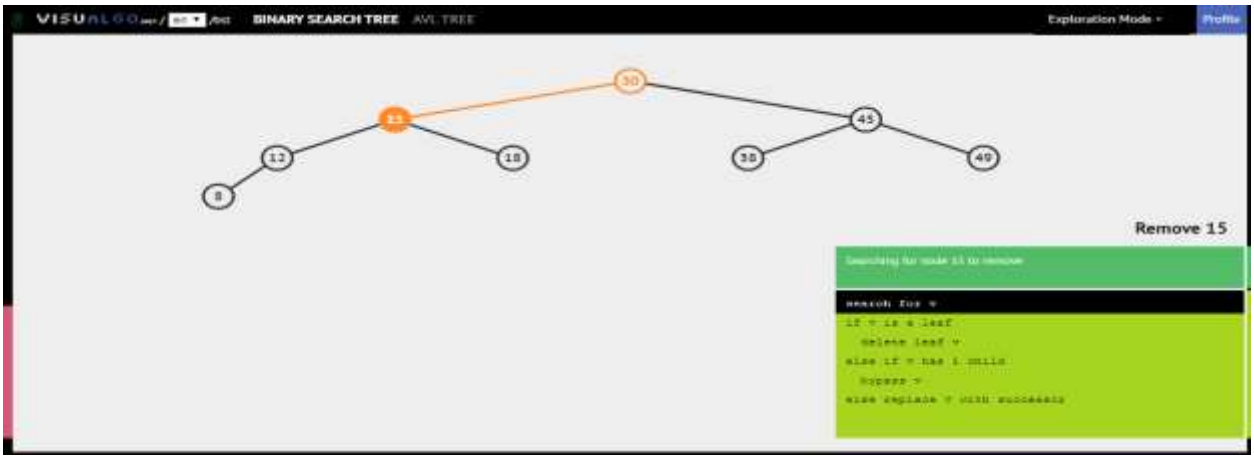


Figure 8 Find the element 15 to remove

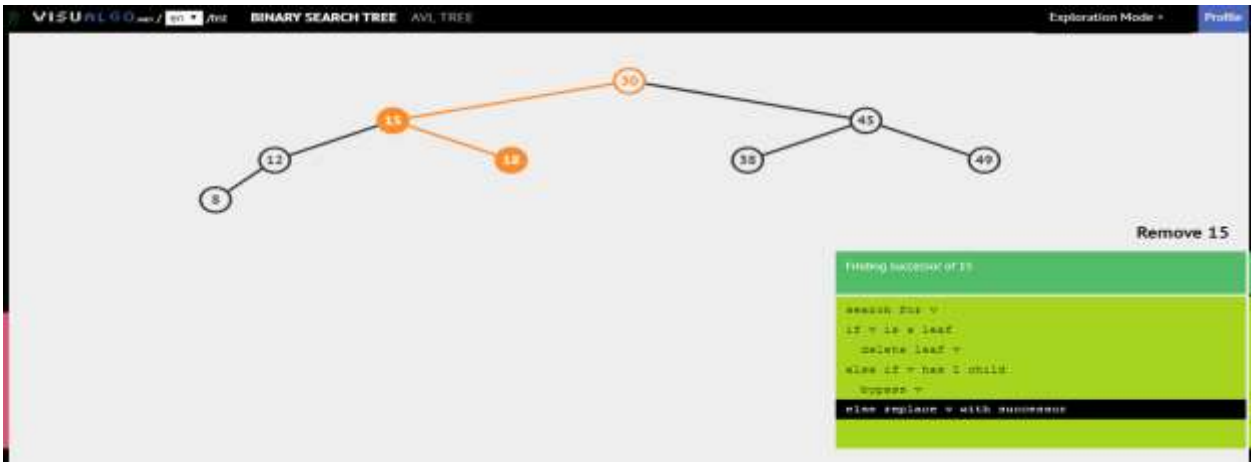


Figure 9 Select the highest element from right sub tree of 15

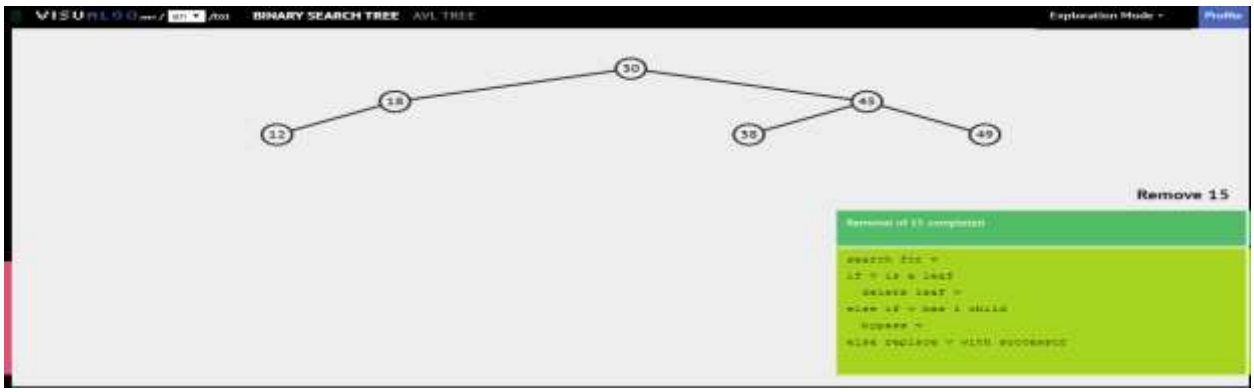


Figure 10 Replace Highest sub tree element with the deleted element. (18 with 15)

To Search an Element in Binary Search Tree

Step 1 to search an element, select the ***Search option*** on the menu.
Step 2 Select ***find minimum*** option for minimum value element or Select ***find maximum*** option for Maximum value element or Type the element name to search a random element.
Step 3 press the ***go button***

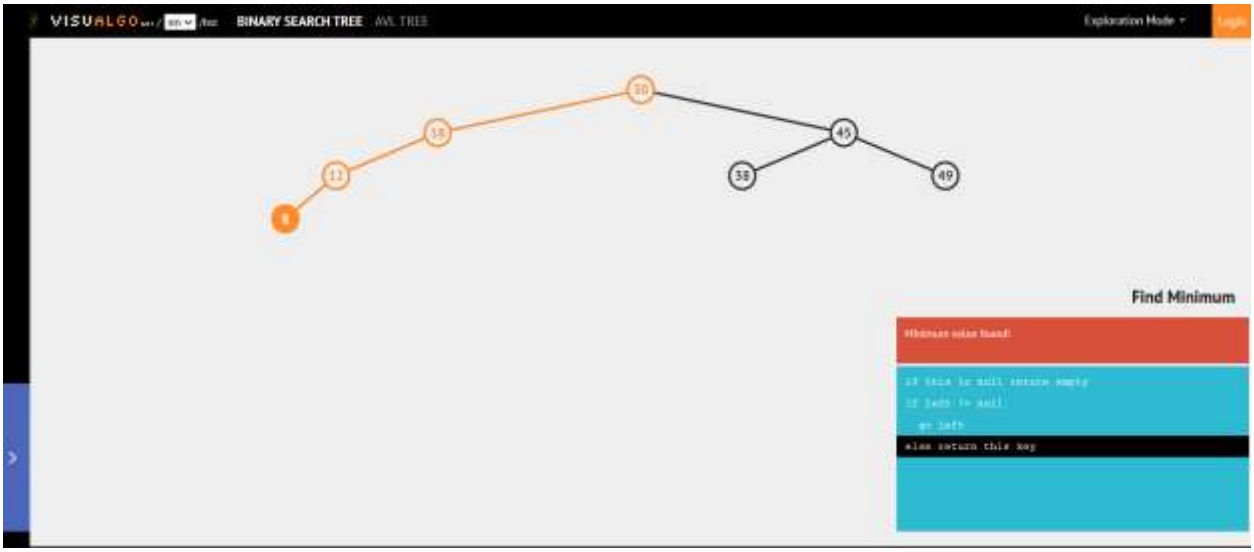


Figure 11 finds a minimum element in BST

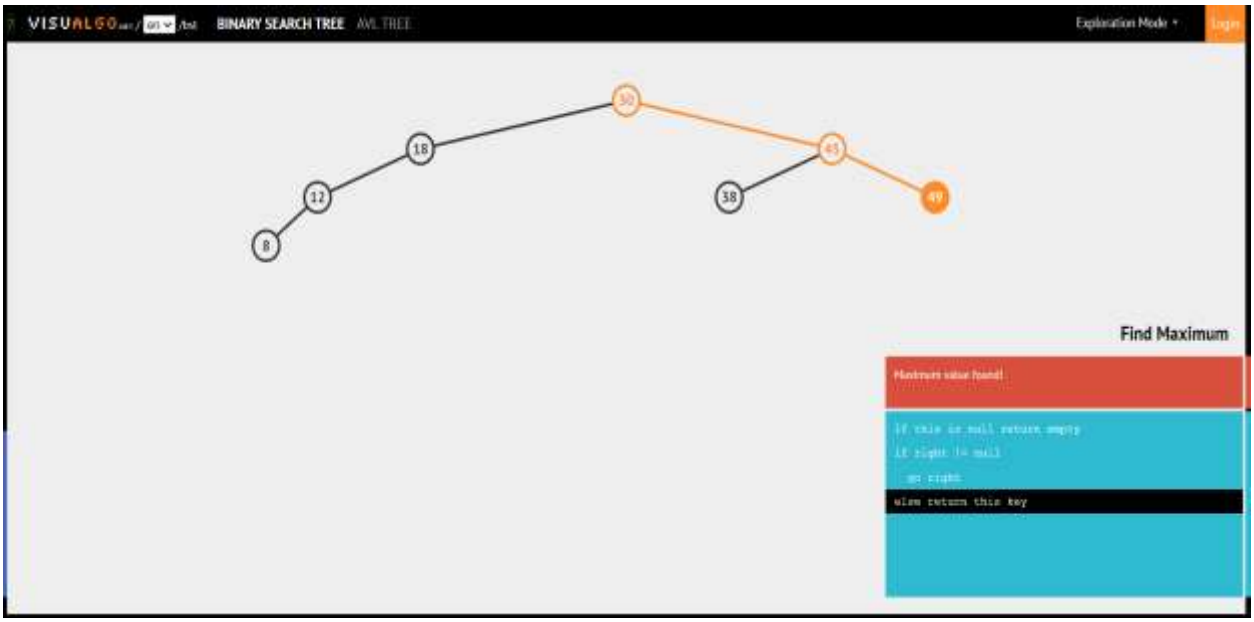


Figure 12 finds the Maximum element in BST

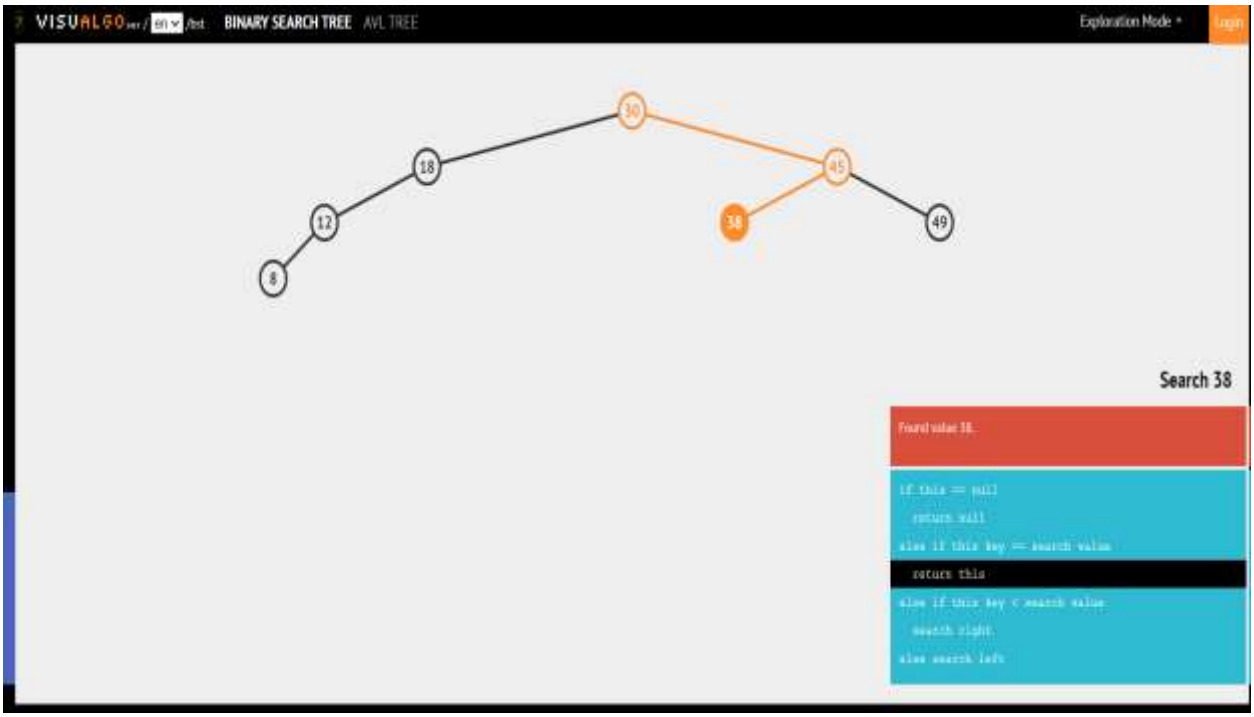


Figure 13 search for a random element (38)

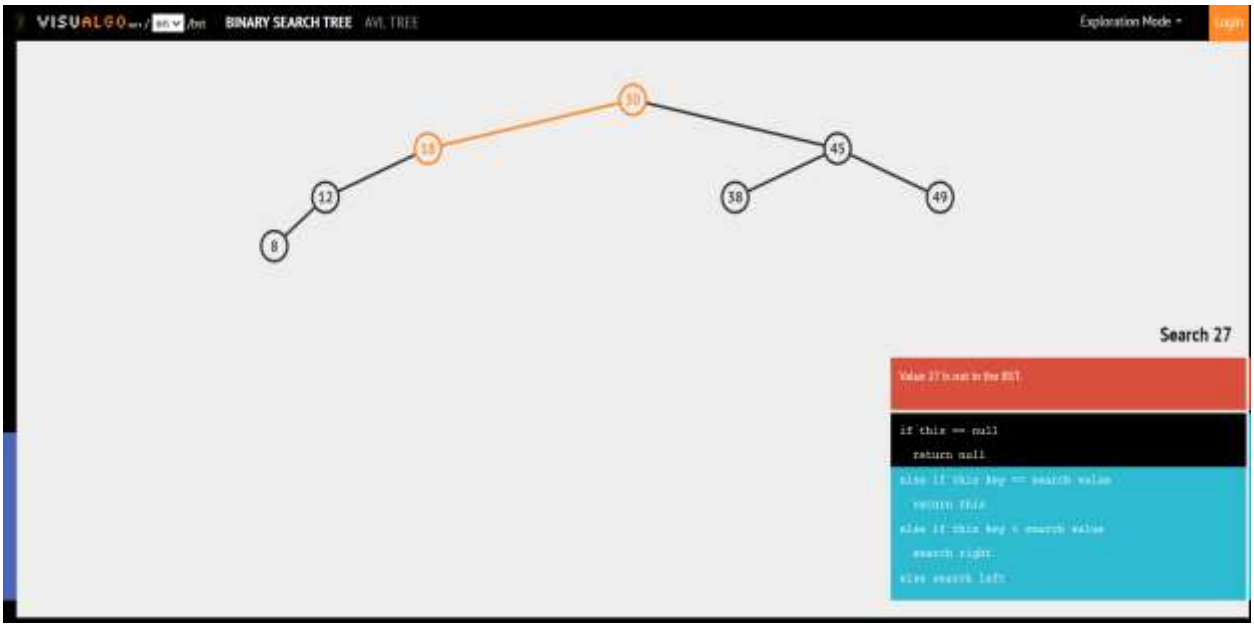
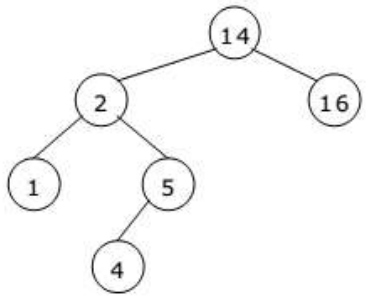


Figure 14 Searching element 27

Quiz Question

1. For the binary search tree shown in the figure below. Suppose we remove the root, replacing it with some node from the left subtree. What will be the new root node?



   a) 3
   b) 4
   c) 5
   d) 2

Ans – (c)

Explanation - In the left subtree, 5 is the biggest element.

2. In a binary search tree, which traversal type would print the values in the nodes in sorted order?

   a) Preorder
   b) Postorder
   c) In order
   d) None of the above

Ans – (c)

Explanation - In order traversal of the binary search tree is equal to ascending order of the tree's nodes.

3. Which of the following statements about binary search trees is true?
   a) Every binary search tree has at least two nodes.
   b) Every binary search tree root has exactly two children.
   c). Every root node has at least two children.
   d). Every non-root node has exactly one parent.

Ans – (d)

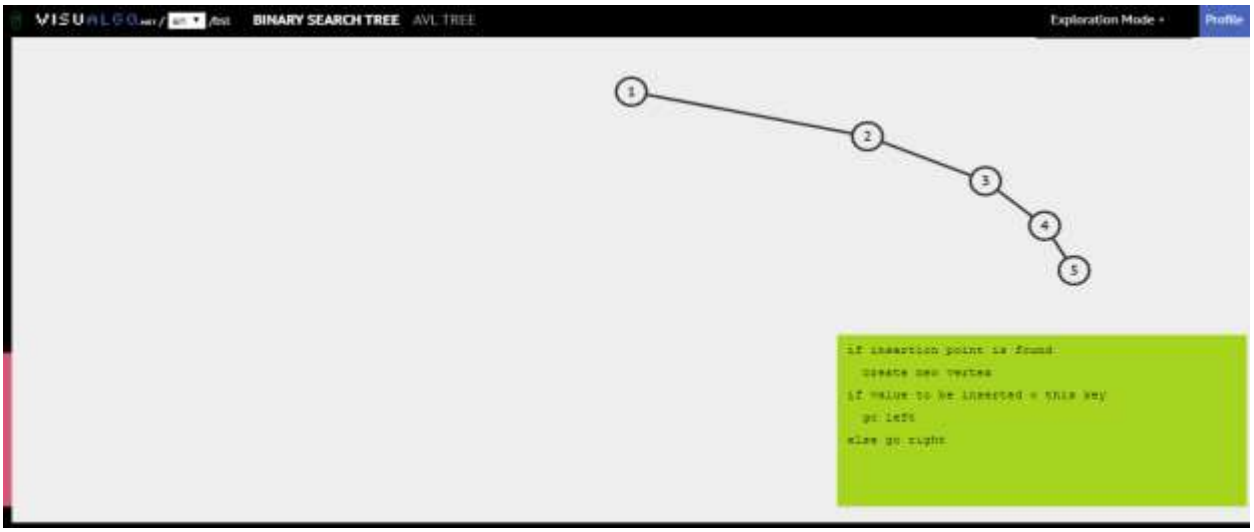
Explanation – In-tree data structure, every non-root node will have exactly one parent node, and only the root node doesn't contain a parent node.

Questions to practice

1. What will happen if the values are given in order (ex 1,2,3,4,5.)?
2. If the inserting element is already present, what will the algorithm do?

Answer for question 1:

The binary search tree will be formed on the right-hand side, which we call Right Skewed Tree.



Answer for question 2:

The algorithm will not accept the duplicate element.