Lesson 03 Demo 02

Working with Binary Tree

Objective: To demonstrate the important methods for binary tree in JavaScript

Tools required: Visual Studio Code and Node.js

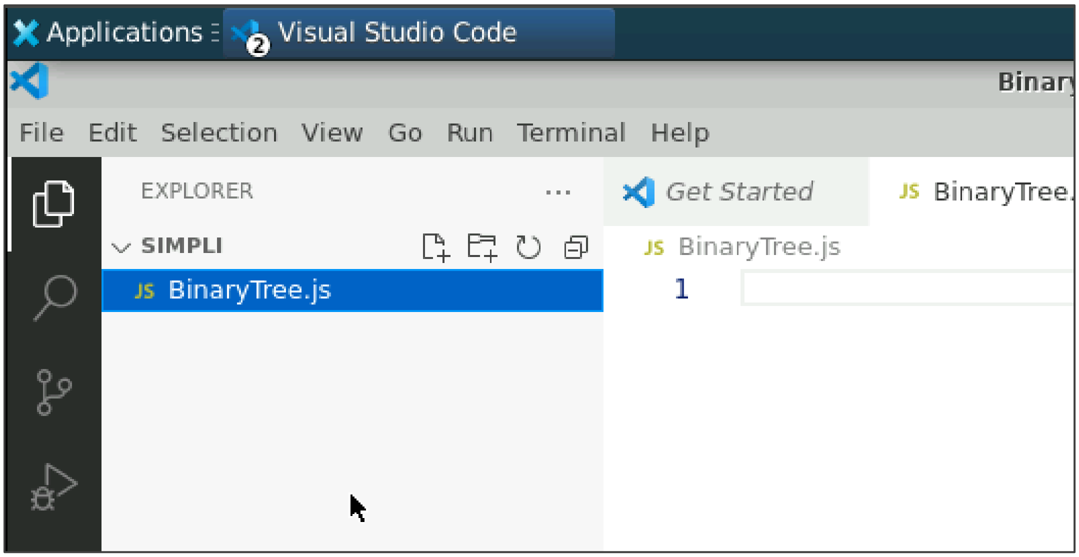
Prerequisites: Familiarity with binary tree basics and JavaScript

Steps to be followed:

1. Create and execute the JS file

Step 1: Create and execute the JS file

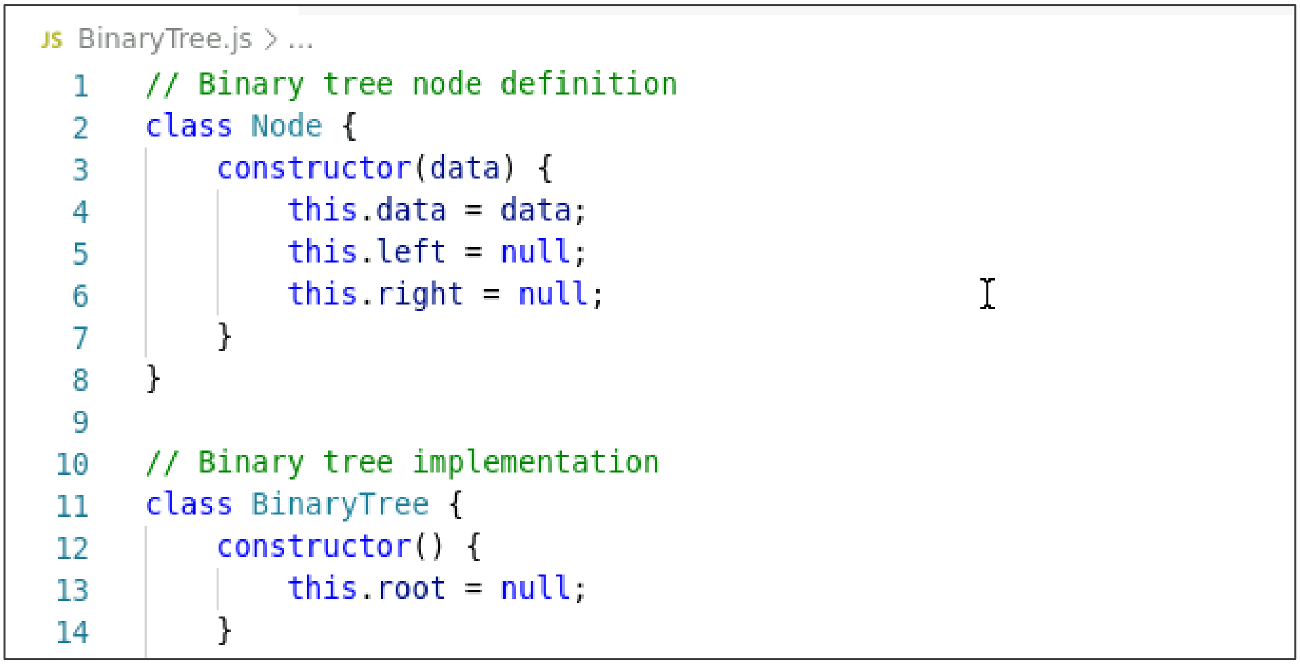
1.1 Open the Visual Studio Code editor and create a JavaScript file named **BinaryTree.js**



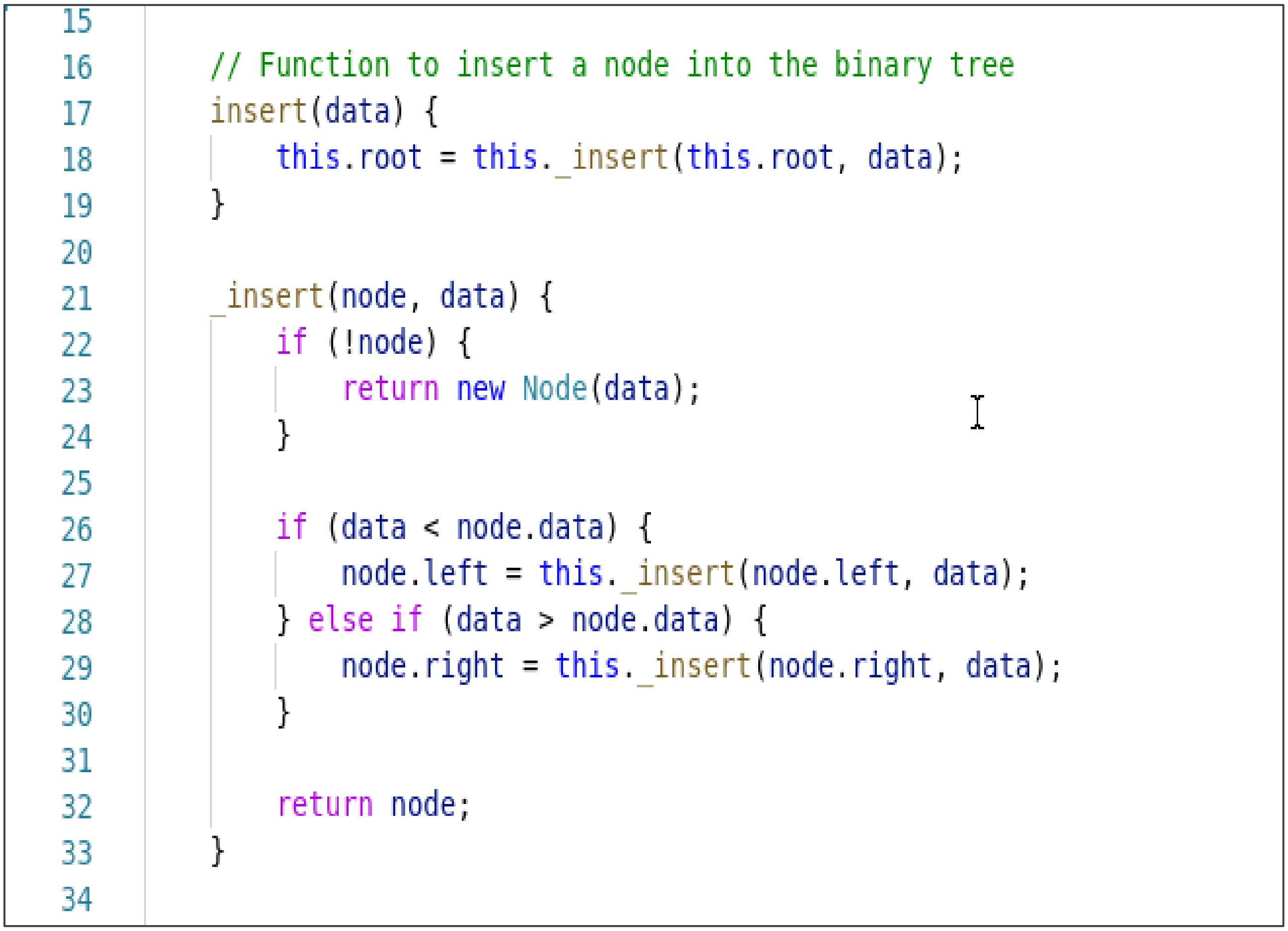
1.2 Write the code given below in the **BinaryTree.js** file:

```
// Binary tree node definition
class Node {
   constructor(data) {
      this.data = data;
      this.left = null;
      this.right = null;
   }
}
// Binary tree implementation
class BinaryTree {
   constructor() {
      this.root = null;
   }
   // Function to insert a node into the binary tree
   insert(data) {
      this.root = this._insert(this.root, data);
   }
   _insert(node, data) {
      if (!node) {
         return new Node(data);
      }
      if (data < node.data) {
         node.left = this._insert(node.left, data);
      } else if (data > node.data) {
         node.right = this._insert(node.right, data);
      }
      return node;
   }
   // Function to search for a node in the binary tree
   search(data, node = this.root) {
      if (!node) {
         return false;
      }
      if (data === node.data) {
return true;
      } else if (data < node.data) {
         return this.search(data, node.left);
      } else {
         return this.search(data, node.right);
      }
   }
```

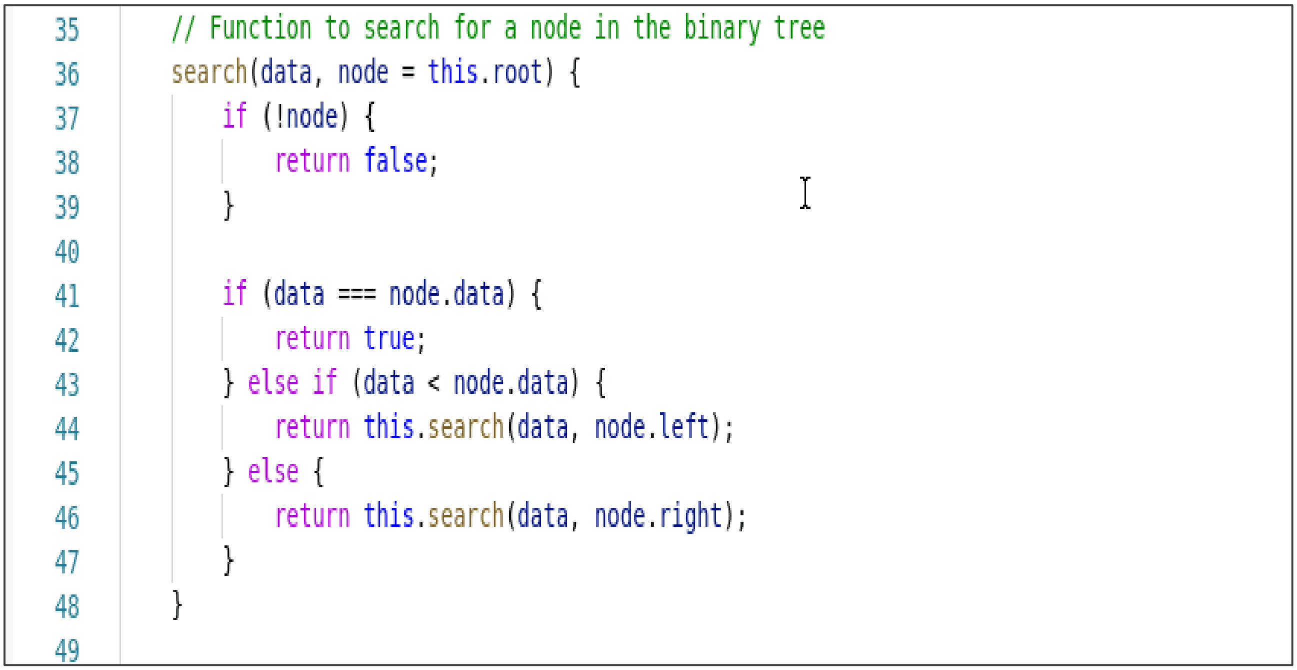
```
  // Function to find the minimum value in the binary tree
  findMin(node = this.root) {
    if (!node) {
      return null;
    }

    while (node.left) {
      node = node.left;
    }
    return node.data;
  }
}

// Example usage
const tree = new BinaryTree();
tree.insert(10);
tree.insert(5);
tree.insert(15);
tree.insert(3);
tree.insert(8);

console.log('Searching for 15:', tree.search(15));
console.log('Minimum value:', tree.findMin());
```



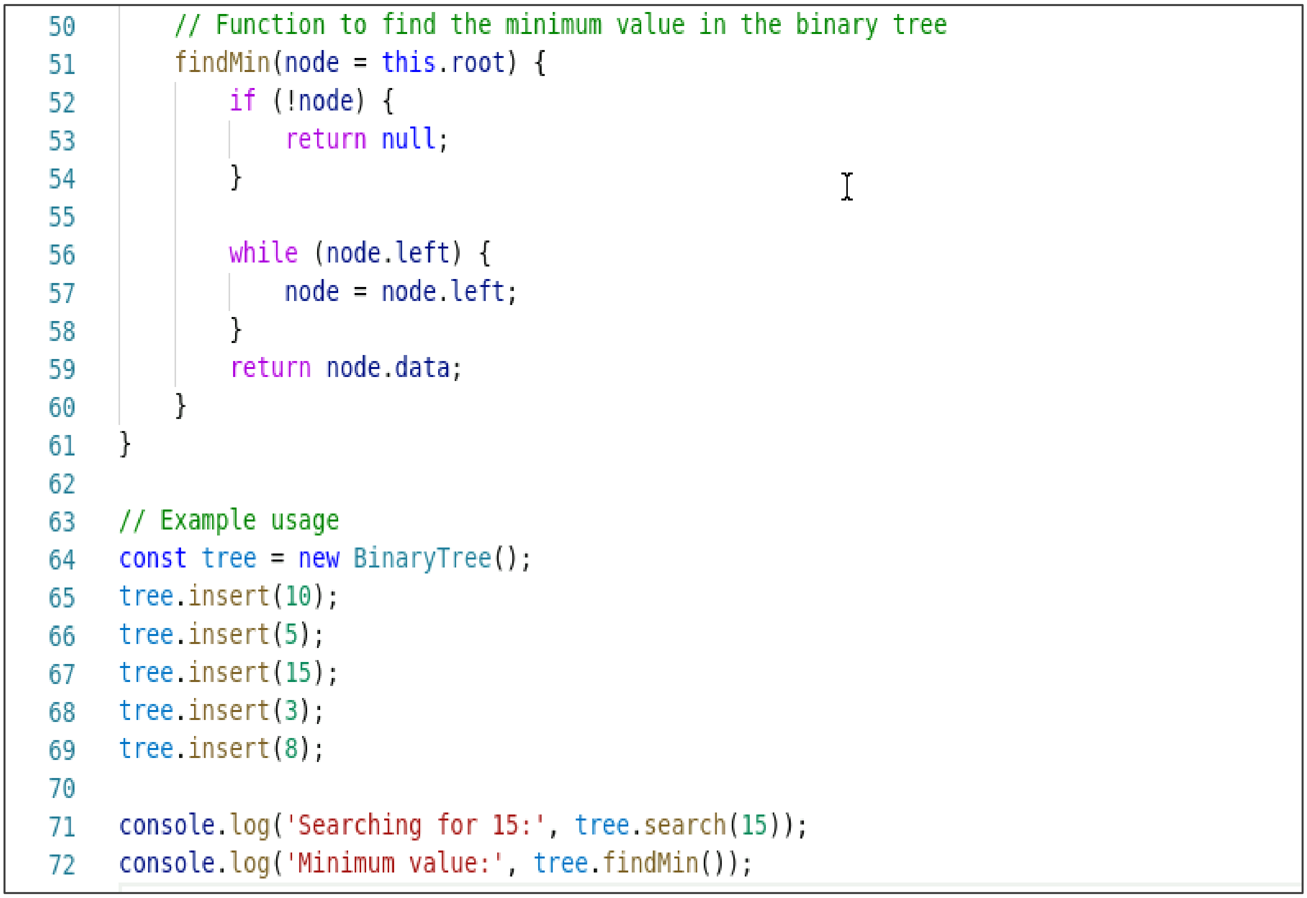
```
// Binary tree node definition
class Node {
    constructor(data) {
        this.data = data;
        this.left = null;
        this.right = null;
    }
}

// Binary tree implementation
class BinaryTree {
    constructor() {
        this.root = null;
    }
```



```

    // Function to insert a node into the binary tree
    insert(data) {
        this.root = this._insert(this.root, data);
    }

    _insert(node, data) {
        if (!node) {
            return new Node(data);
        }

        if (data < node.data) {
            node.left = this._insert(node.left, data);
        } else if (data > node.data) {
            node.right = this._insert(node.right, data);
        }

        return node;
    }

```

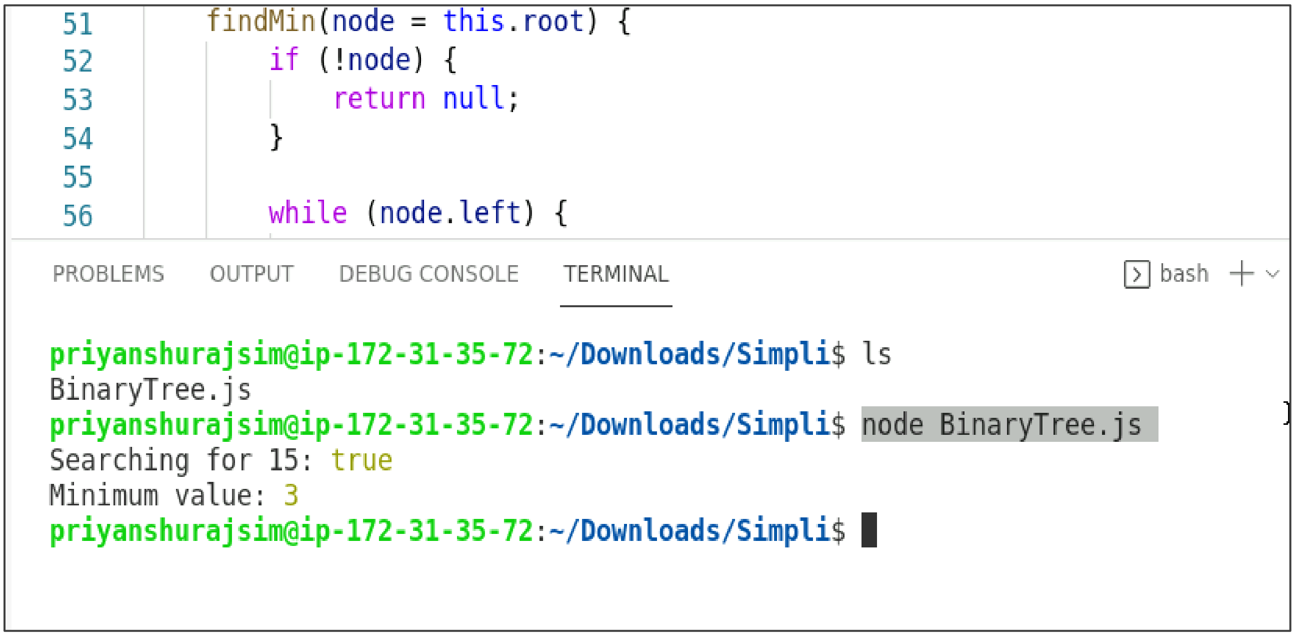


```
    // Function to search for a node in the binary tree
    search(data, node = this.root) {
        if (!node) {
            return false;
        }

        if (data === node.data) {
            return true;
        } else if (data < node.data) {
            return this.search(data, node.left);
        } else {
            return this.search(data, node.right);
        }
    }

```



```javascript
    // Function to find the minimum value in the binary tree
    findMin(node = this.root) {
        if (!node) {
            return null;
        }

        while (node.left) {
            node = node.left;
        }
        return node.data;
    }
}

// Example usage
const tree = new BinaryTree();
tree.insert(10);
tree.insert(5);
tree.insert(15);
tree.insert(3);
tree.insert(8);

console.log('Searching for 15:', tree.search(15));
console.log('Minimum value:', tree.findMin());
```

1.3 Save the file and execute it in the terminal using the command given below:
node BinaryTree.js



```javascript
    findMin(node = this.root) {
        if (!node) {
            return null;
        }

        while (node.left) {
```

```
priyanshurajsim@ip-172-31-35-72:~/Downloads/Simpli$ ls
BinaryTree.js
priyanshurajsim@ip-172-31-35-72:~/Downloads/Simpli$ node BinaryTree.js
Searching for 15: true
Minimum value: 3
priyanshurajsim@ip-172-31-35-72:~/Downloads/Simpli$
```

This example shows key techniques like searching for a node and finding the smallest value in a binary tree with JavaScript.

By following these steps, you have successfully implemented and executed the methods for binary tree in JavaScript.