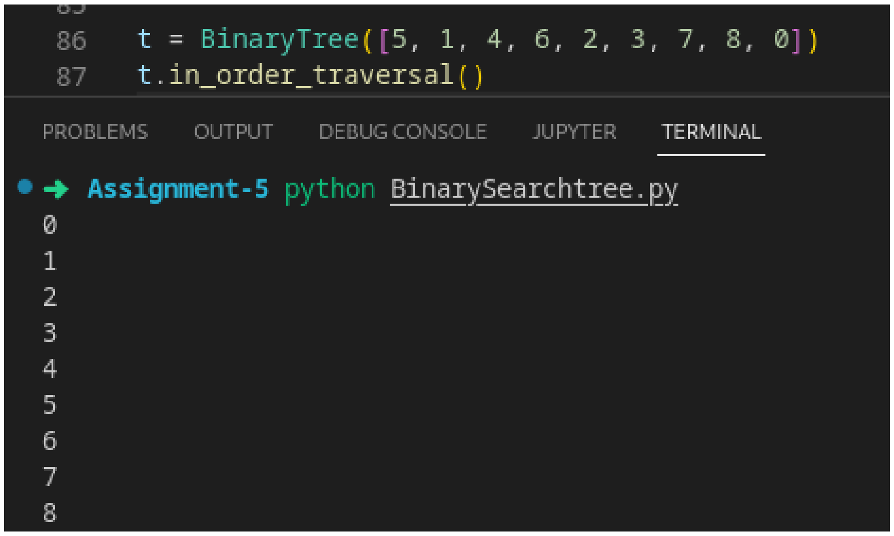
Assignment-5 (30 points)
Binary Search Tree

Due date: 10/4/2022

Exercise-1: (15 points)

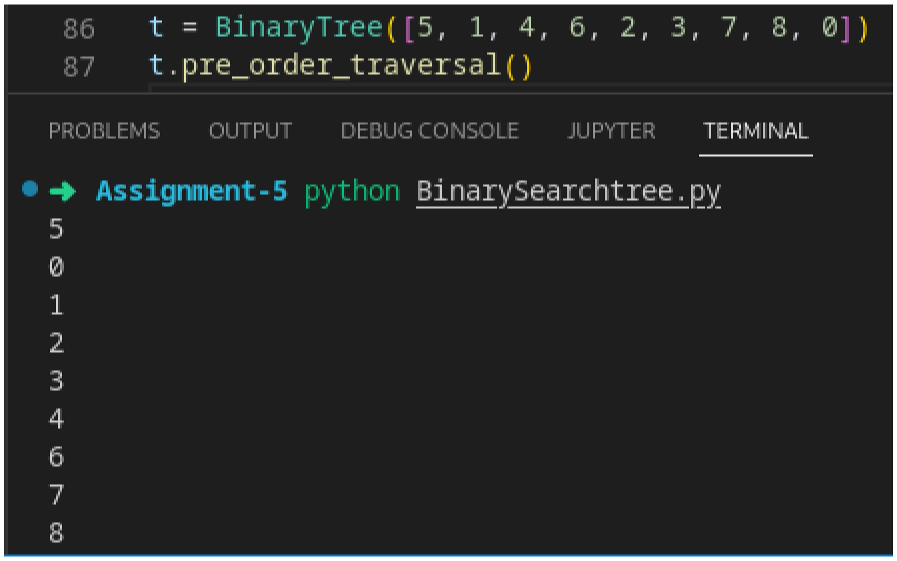
Develop a BinarySearchtree.py which can perform the following functions:

- Insert node to a tree
- Perform In-order traversal
- Perform Pre-order traversal
- Perform Post-order traversal
- Find a node



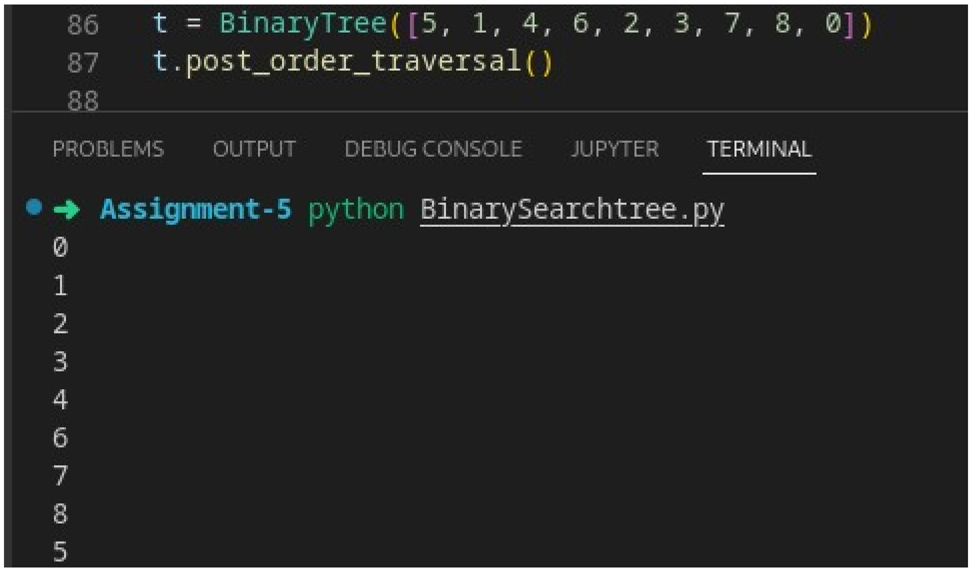
```
t = BinaryTree([5, 1, 4, 6, 2, 3, 7, 8, 0])
t.in_order_traversal()
```

```
● ➜ Assignment-5 python BinarySearchtree.py
0
1
2
3
4
5
6
7
8
```



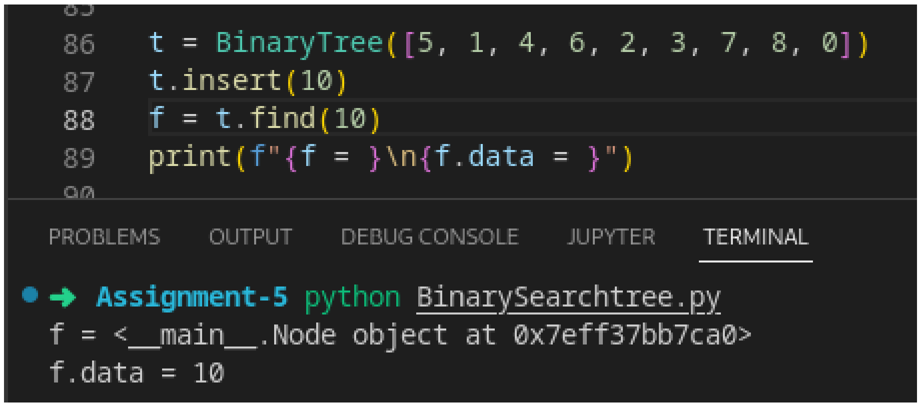
```
t = BinaryTree([5, 1, 4, 6, 2, 3, 7, 8, 0])
t.pre_order_traversal()
```

```
● ➜ Assignment-5 python BinarySearchtree.py
5
0
1
2
3
4
6
7
8
```



```
t = BinaryTree([5, 1, 4, 6, 2, 3, 7, 8, 0])
t.post_order_traversal()
```

```
● ➜ Assignment-5 python BinarySearchtree.py
0
1
2
3
4
6
7
8
5
```



```
t = BinaryTree([5, 1, 4, 6, 2, 3, 7, 8, 0])
t.insert(10)
f = t.find(10)
print(f"{f = }\n{f.data = }")
```

```
● ➜ Assignment-5 python BinarySearchtree.py
f = <__main__.Node object at 0x7eff37bb7ca0>
f.data = 10
```

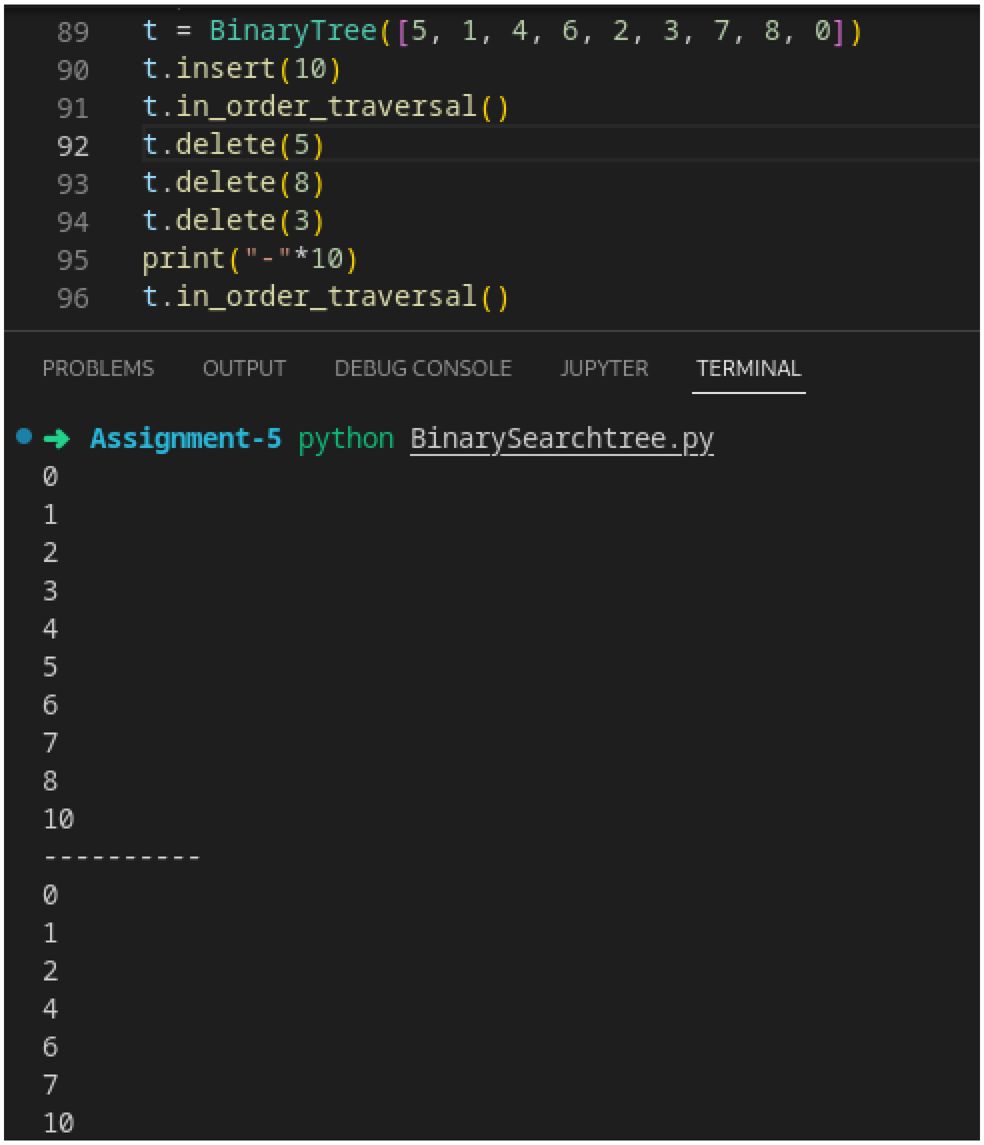
Exercise-2: (15 points)

Write a function to remove a node from a tree data structure? This function should consider all the three cases: case-1: remove a leaf node, case-2: remove a node with one child and case-3: remove a node with two children.

Perform the time complexity for this function. Briefly explain?

This delete function works by keeping tracks of all the values in the node inside an array, and it perform deletion by removing the values in that array and rebuild the tree.

Since everytime something is delete it rebuilding the tree the time complexity of this function is O(n)



```python
t = BinaryTree([5, 1, 4, 6, 2, 3, 7, 8, 0])
t.insert(10)
t.in_order_traversal()
t.delete(5)
t.delete(8)
t.delete(3)
print("-"*10)
t.in_order_traversal()
```

PROBLEMS OUTPUT DEBUG CONSOLE JUPYTER TERMINAL

```
● ➜ Assignment-5 python BinarySearchtree.py
0
1
2
3
4
5
6
7
8
10
----------
0
1
2
4
6
7
10
```