

Code :

The screenshot shows a PyCharm IDE with a project named 'TermProjectFinals\_Jacoblvn'. The main editor displays the code for 'Tree.py', which defines a 'TreeNode' class and a 'create\_tree' function. The code is split into two parts, 'PartA' and 'PartB', which implement different tree structures. The left sidebar shows the project structure, and the bottom status bar indicates the current file is 'Tree.py'.

```
1 from LinkedBinaryTree import LinkedBinaryTree
2
3 usage
4
5 def create_tree1():
6     tree = LinkedBinaryTree()
7     root = tree._add_root('.')
8     left = tree._add_left(root, '@')
9     tree._add_left(left, '@')
10    tree._add_right(left, '@')
11    right = tree._add_right(root, '@')
12    left_right = tree._add_left(right, '@')
13    tree._add_left(left_right, '@')
14    tree._add_right(left_right, '@')
15    right_right = tree._add_right(right, '@')
16    tree._add_left(right_right, '@')
17    tree._add_right(right_right, '@')
18    return tree
19
20 usage
21
22 def create_tree2():
23     tree = LinkedBinaryTree()
24     root = tree._add_root('.')
25     left = tree._add_left(root, '@')
26     left_left = tree._add_left(left, '@')
27     tree._add_left(left_left, '@')
28     tree._add_right(left_left, '@')
29     tree._add_right(left, '@')
30     right = tree._add_right(root, '@')
31     tree._add_left(right, '@')
32     tree._add_right(right, '@')
33     return tree
34
35 usage
36
37 def create_tree3():
38     tree = LinkedBinaryTree()
39     root = tree._add_root('.')
40     left = tree._add_left(root, '@')
```

[illegible]

```
Project > TermProjectFinals_Jacobvian > main > TermProject2PartA_Jacobvian.py > TermProject2PartB_Jacobvian.py > Term Project #2.py > main.py > LinkedBinaryTree.py > Tree.py > BinaryTree.py

TermProject2PartA_Jacobvian.py
10 def create_tree1():
11     tree = LinkedBinaryTree()
12     root = tree._add_root('a')
13     left = tree._add_left(root, 'b')
14     left_left = tree._add_left(left, 'c')
15     tree._add_left(left_left, 'd')
16     tree._add_right(left_left, 'e')
17     tree._add_right(left, 'f')
18     tree._add_right(right, 'g')
19     return tree
20
21 usage
22 def create_tree2():
23     tree = LinkedBinaryTree()
24     root = tree._add_root('a')
25     tree._add_right(root, 8)
26
27     left = tree._add_left(root, '/')
28
29     num_left = tree._add_left(left, '+')
30     tree._add_left(num_left, '+')
31     tree._add_right(num_left, '-')
32     tree._add_left(tree.left(num_left), 5)
33     tree._add_right(tree.left(num_left), 2)
34     tree._add_left(tree.right(num_left), 2)
35     tree._add_right(tree.right(num_left), 1)
36
37     den_right = tree._add_right(left, '+')
38     tree._add_left(den_right, '+')
39     tree._add_right(den_right, '-')
40     tree._add_left(tree.left(den_right), 2)
```

Run TermProject2PartB\_Jacobvian

Postorder traversal: d b h g e f c a r

131:1 CRLF UTF-8 4

```
Project > TermProjectFinals_Jacobvian > main > TermProject2PartA_Jacobvian.py > TermProject2PartB_Jacobvian.py > Term Project #2.py > main.py > LinkedBinaryTree.py > Tree.py > BinaryTree.py

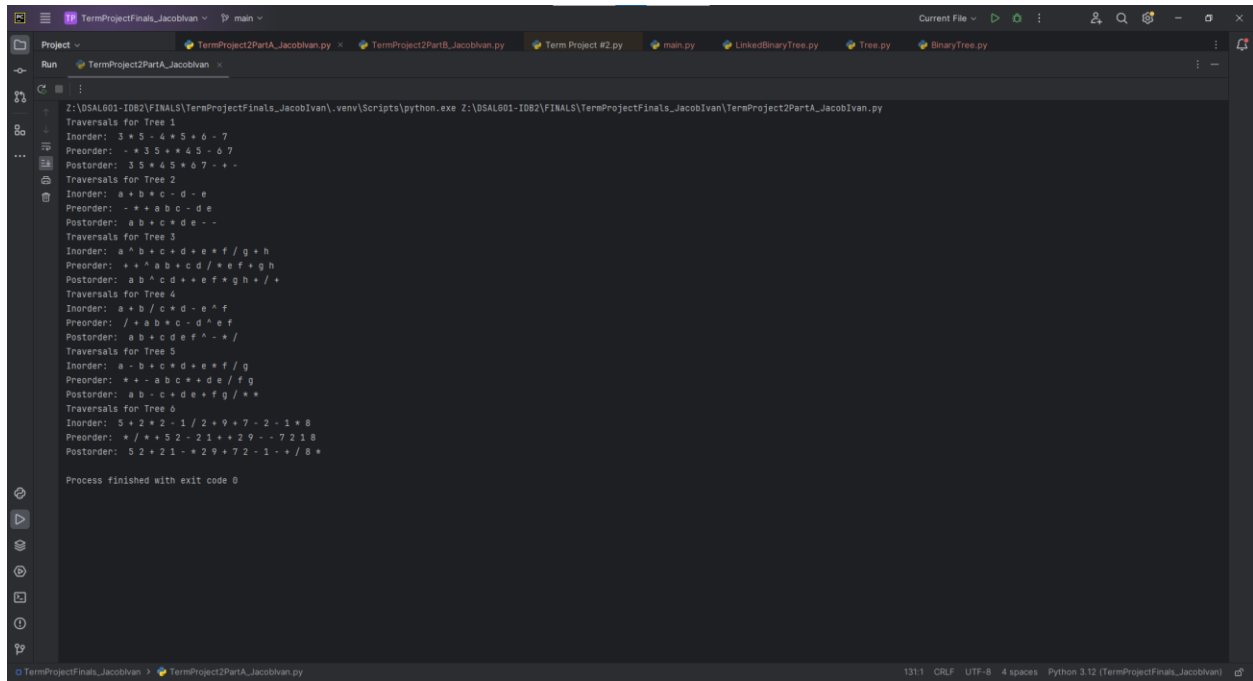
TermProject2PartB_Jacobvian.py
10 den_right = tree._add_right(left, '+')
11 tree._add_left(den_right, '+')
12 tree._add_right(den_right, '-')
13 tree._add_left(tree.left(den_right), 2)
14 tree._add_right(tree.left(den_right), 4)
15 tree._add_left(tree.right(den_right), '-')
16 tree._add_right(tree.right(den_right), 1)
17 tree._add_left(tree.left(tree.right(den_right)), 7)
18 tree._add_right(tree.left(tree.right(den_right)), 2)
19
20 return tree
21
22 usage
23 def print_traversals(tree, tree_num):
24     print(f"Traversals for tree {tree_num}")
25     print("Inorder: ", end=" ")
26     for node in tree.inorder():
27         print(node.element(), end=" ")
28     print()
29
30     print("Preorder: ", end=" ")
31     for node in tree.preorder():
32         print(node.element(), end=" ")
33     print()
34
35     print("Postorder: ", end=" ")
36     for node in tree.postorder():
37         print(node.element(), end=" ")
38     print()
39
40 # Create and print traversals for all trees
41 trees = [create_tree1(), create_tree2(), create_tree3(), create_tree4(), create_tree5(), create_tree6()]
42
43 for i, tree in enumerate(trees, start=1):
44     print_traversals(tree, i)
45
46
```

Run TermProject2PartB\_Jacobvian

Postorder traversal: d b h g e f c a r

131:1 CRLF UTF-8 4

## OUTPUT:



```

Z:\DSAL601-IDB2\FINALS\TermProjectFinals_JacobIvan\venv\Scripts\python.exe Z:\DSAL601-IDB2\FINALS\TermProjectFinals_JacobIvan\TermProject2PartA_JacobIvan.py
Traversals for Tree 1
Inorder: 3 * 5 - 4 * 5 + 6 - 7
Preorder: - * 3 5 + * 4 5 - 6 7
Postorder: 3 5 + 4 5 + 6 7 - + -
Traversals for Tree 2
Inorder: a + b * c - d - e
Preorder: - + + a b c - d e
Postorder: a b + c * d e - -
Traversals for Tree 3
Inorder: a ^ b + c * d + e * f / g + h
Preorder: + + ^ a b + c d / * e f + g h
Postorder: a b ^ c d + e * f * g h + / +
Traversals for Tree 4
Inorder: a + b / c * d - e ^ f
Preorder: / + a b + c - d ^ e f
Postorder: a b + c d e f ^ - * /
Traversals for Tree 5
Inorder: a - b * c + d + e * f / g
Preorder: * + - a b c + d e / f g
Postorder: a b - c + d e * f g / + *
Traversals for Tree 6
Inorder: 5 + 2 * 2 - 1 / 2 + 9 + 7 * 2 - 1 * 8
Preorder: * / + * 5 2 - 2 1 + + 2 9 - - 7 2 1 8
Postorder: 5 2 + 2 1 - * 2 9 + 7 2 - 1 - + / 8 *

Process finished with exit code 0

```