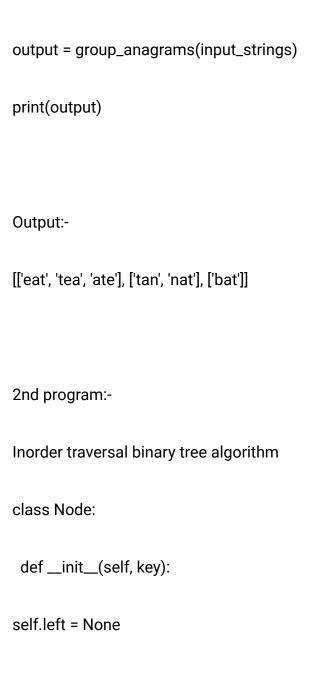
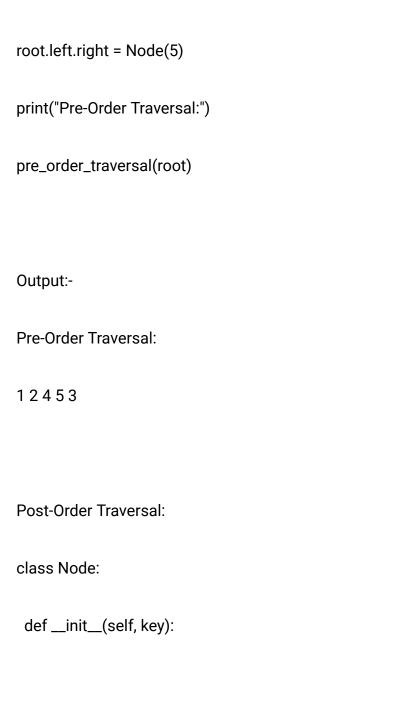
```
Python assignment -2
1st program :-
def group_anagrams(strs):
 anagrams = defaultdict(list)
 for s in strs:
 anagrams[tuple(sorted(s))].append(s)\\
 return list(anagrams.values())
input_strings = ['eat', 'tea', 'tan', 'ate', 'nat', 'bat']
```



```
self.right = None
self.value = key
def in_order_traversal(root):
 if root:
  in_order_traversal(root.left)
print(root.value, end=' ') in_order_traversal(root.right)
root = Node(1)
root.left = Node(2)
root.right = Node(3)
root.left.left = Node(4)
root.left.right = Node(5)
```

print("In-Order Traversal:")
in_order_traversal(root)
Output:-
In-Order Traversal:
42513
Pre-Order Traversal:
class Node:
definit(self, key):
self.left = None

```
self.right = None
   self.value = key
def pre_order_traversal(root):
 if root:
   print(root.value, end=' ')
 pre_order_traversal(root.left)
pre_order_traversal(root.right)
root = Node(1)
root.left = Node(2)
root.right = Node(3)
root.left.left = Node(4)
```



```
self.left = None
   self.right = None
   self.value = key
def post_order_traversal(root):
 if root: post_order_traversal(root.left) post_order_traversal(root.right)
print(root.value, end=' ')
root = Node(1)
root.left = Node(2)
root.right = Node(3)
root.left.left = Node(4)
root.left.right = Node(5)
```

print("Post-Order Traversal:")
post_order_traversal(root)
Output:-
Post-Order Traversal:
45231
3rd program :-
def bubble_sort(arr):
n = len(arr)
for i in range(n):

for j in range(0, n - i - 1):

if arr[j] > arr[j + 1]:

arr[j], arr[j + 1] = arr[j + 1], arr[j]

a = [12, 5, 7, 18, 11, 6, 12, 4, 17, 1]

bubble_sort(a)

print("Sorted list:", a)

Output:-

[1, 4, 5, 6, 7, 11, 12, 12, 17, 18]

4th program :-

```
def linear_search(arr, x):
 for i in range(len(arr)):
   if arr[i] == x:
     return True, i
 return False, -1
data = [11, 23, 58, 31, 56, 77, 43, 12, 65, 19]
result = linear_search(data, 31)
print(result)
Output:-
(True, 3)
```