Cost analysis of the algorithm (c_1→problem_3.py)

```
class Node:
    def __init__(self, value):
        self.value = value                                  #c_1=1
        self.left = None                                    #c_2=1
        self.right = None                                   #c_3=1

def findMaxPreOrder(node):
    if node is None:                                        #c_4=n+1
        return float('-inf')                                #c_5=n
    currentValue = node.value                               #c_6=n
    leftMax = findMaxPreOrder(node.left)                    #(c_7+c_8)=n+1
    rightMax = findMaxPreOrder(node.right)
    return max(currentValue,leftMax,rightMax)               #c_9=n

def findMinInOrder(node):
    if node is None:                                        #c_10=n+1
        return float('inf')                                 #c_11=n
    leftMin = findMinInOrder(node.left)                     #c_12+c_14=n+1
    currentValue = node.value                               #c_13=n
    rightMin = findMinInOrder(node.right)                   #c_14*
    return min(leftMin,currentValue,rightMin)               #c_15=n

def postOrderValues(node):
    if node is None:                                        #c_16=n+1
        return []                                           #c_17=n
    return postOrderValues(node.left) + postOrderValues(node.right) +
[node.value]                                                #c_18=n

def findAveragePosOrder(node):
    values = postOrderValues(node)                          #c_19=n
    n = len(values)                                         #c_20=1
    if n == 0:                                              #c_21=1
        return float('inf')                                 #c_22=1
    return sum(values) / n                                  #c_23=n
```

- Basic operation:

  $c_4 = n + 1$

  $(c_7 + c_8) = n + 1$

  $c_{10} = n + 1$

$(c_{12} + c_{14}) = n + 1$

$c_{16} = n + 1$

- Time complexity calculation:

  $T(n) = (c_4 + c_{10} + c_{16}).(n + 1) + ((c_7 + c_8) + (c_{12} + c_{14})).(n + 1)$

  $T(n) = 3.(n + 1) + 2.(n + 1)$

  $T(n) = 3.n + 3 + 2.n + 2$

  $T(n) = 3.n + 2.n + 5$

  $T(n) = 5n + 5$

  $T(n) \in O(n)$

- Solving the recurrence:

  $T(n) = 2T(n/2) + 5,\ n > 1,\ T(0) = 2, T(1) = 7$

  $T(2) = 2T(2/2) + 5 = 2T(1) + 5 = 2.7 + 5 = 4 + 5 = 19$

  $T(3) = 2T(3/2) + 5 = 2T(1) + 5 = 2.7 + 5 = 14 + 5 = 19$

  $T(4) = 2T(4/2) + 5 = 2T(2) + 5 = 2.19 + 5 = 38 + 5 = 43$

  $T(5) = 2T(5/2) + 5 = 2T(2) + 5 = 2.19 + 5 = 38 + 5 = 43$

  $T(6) = 2T(6/2) + 5 = 2T(3) + 5 = 2.19 + 5 = 38 + 5 = 43$

  $T(7) = 2T(7/2) + 5 = 2T(3) + 5 = 2.19 + 5 = 38 + 5 = 43$

  $T(8) = 2T(8/2) + 5 = 2T(4) + 5 = 2.43 + 5 = 86 + 5 = 91$

$$T(n) = \sum_{i=0}^{log_2 n - 1} 5.2^i + 7n$$