Task(1-7)

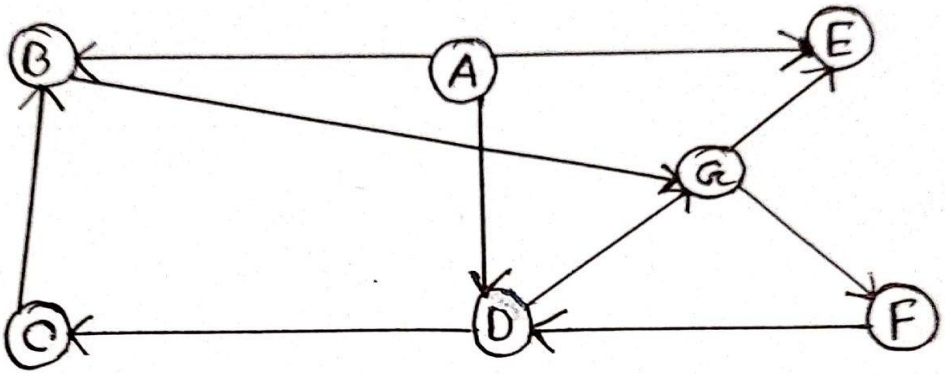
```
class TreeNoDe:
    def __init__(self,data):
        self.data = data
        self.left = None
        self.right = None
        self.parent = None

#------------------Task-1----------------------

    def TreeHeighT (self, rooT):
        if rooT is None:
            return -1
        else:
            return 1+ max(self.TreeHeighT(rooT.left), self.TreeHeighT(rooT.right))

#------------------Task-2----------------------

    def TreeLeveL (self, rooT):
        if rooT is None:
            return 0
        else:
            return 1+ self.TreeLeveL(rooT.parent)

#------------------Task-3----------------------

    def Pre_OrDeR(self, noDe):
        if noDe is not None:
            print(noDe.data, end = " ")
            self.Pre_OrDeR(noDe.left)
            self.Pre_OrDeR(noDe.right)


#------------------Task-4----------------------

    def In_OrdeR(self, rooT):
        if rooT is not None:
            self.In_OrdeR(rooT.left)
            print(rooT.data, end = " ")
            self.In_OrdeR(rooT.right)
```

```
#------------------Task-5---------------------

    def PosT_OrDer(self, noDE):
        if noDE is not None:
            self.PosT_OrDer(noDE.left)
            self.PosT_OrDer(noDE.right)
            print(noDE.data, end = " ")


#------------------Task-6---------------------

    def ExacTLy_SaMe(self, noDE1, noDE2):

        if noDE1 == None and noDE2 == None:
            return 1

        elif (noDE1 == None and noDE2 != None) or (noDE1 != None and noDE2 == None):
            return 0


        if noDE1.data != None and noDE2.data != None and noDE1.data == noDE2.data:
            if self.ExacTLy_SaMe(noDE1.left, noDE2.left) and self.ExacTLy_SaMe(noDE1.right,
noDE2.right):
                return 1



        return 0

#------------------Task-7---------------------

    def Copy_Tree(self,noDe):
        if noDe !=None:
            tEmP=TreeNoDe(noDe.data)
            tEmP.left=self.Copy_Tree(noDe.left)
            tEmP.right=self.Copy_Tree(noDe.right)
            print(tEmP.data, end=" ")
        elif noDe==None:
            return None

#------------------Code-Tester----------------------

trEE=TreeNoDe(10)
trEE.left=TreeNoDe(20)
trEE.right=TreeNoDe(30)
```

```
trEE.left.left=TreeNoDe(40) #child_under 2
trEE.left.right=TreeNoDe(50) #child_under 2
trEE.right.left=TreeNoDe(60)  #child_under 3
trEE.right.right=TreeNoDe(70)  #child_under 3
trEE.left.left.left=TreeNoDe(80) #child_under 4

#-------------------printing method----------------------
print('Height:',trEE.TreeHeighT(trEE))
print('Level:',trEE.TreeLeveL(trEE))
print()
print('pre order Traverse:')
trEE.Pre_OrDeR(trEE)
print()
print('\nIn order Traverse:')
trEE.In_OrdeR(trEE)
print()
print('\nPost order Traverse:')
trEE.PosT_OrDer(trEE)
print()
#-------------------same or not----------------------

trEE1=TreeNoDe(10)
trEE1.left=TreeNoDe(20)
trEE1.right=TreeNoDe(30)
trEE1.left.left=TreeNoDe(40)    #child under 2
trEE1.left.right=TreeNoDe(50)    #child under 2
trEE1.right.left=TreeNoDe(60)    #child under 3
trEE1.right.right=TreeNoDe(70)    #child under 3
trEE1.left.left.left=TreeNoDe(80) #child under 4
if trEE1.ExacTLy_SaMe(trEE,trEE1)==0:
    print('\nNo two trees are Not Same')
elif trEE1.ExacTLy_SaMe(trEE,trEE1)==1:
    print('\nYes two trees are exactly same ')

#-------------------Copy node last  task----------------------
print()
print('Copy Node:')
trEE.Copy_Tree(trEE)
```

Task-8

Task – 8



ID – 20301268