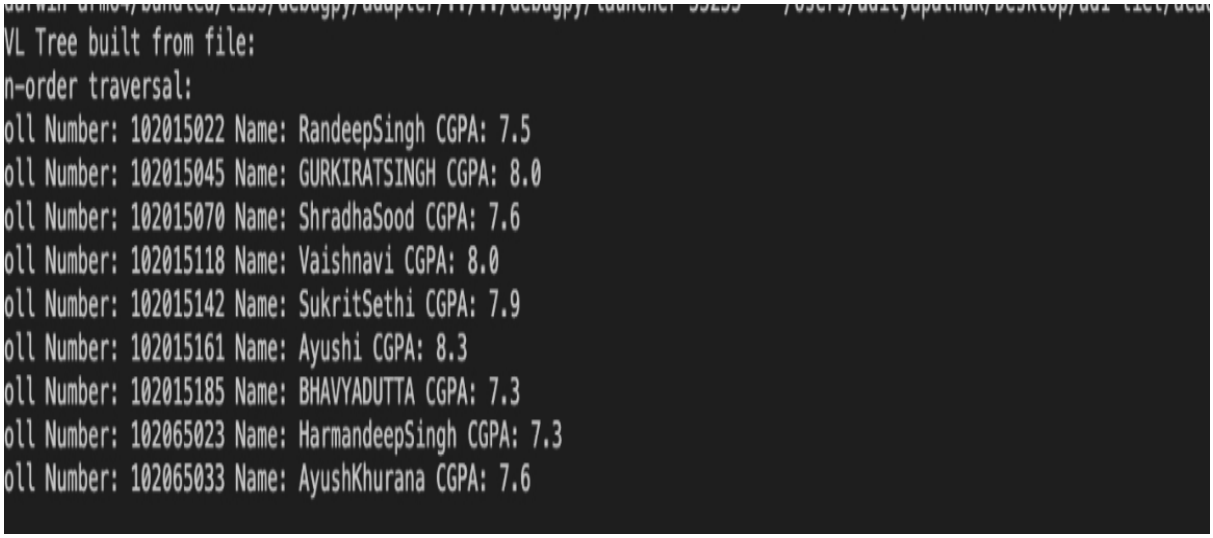
- PRABHNOOR SINGH
- 102115059
- Assignment 3(b)

```python
class treeNode:
    def __init__(self, rnum, name, cgpa):
        self.rnum = rnum
        self.name = name
        self.cgpa = cgpa
        self.height = 1
        self.left = None
        self.right = None

class avlTree:
    def __init__(self):
        self.root = None

    def getHeight(self, node):
        if node is None:
            return 0

        return node.height

    def getBalance(self, node):
        if node is None:
            return 0

        return self.getHeight(node.left) - self.getHeight(node.right)

    def rightRotate(self, y):
        x = y.left
        T2 = x.right
        x.right = y
        y.left = T2
        y.height = 1 + max(self.getHeight(y.left), self.getHeight(y.right))
        x.height = 1 + max(self.getHeight(x.left), self.getHeight(x.right))

        return x

    def leftRotate(self, x):
        y = x.right
        T2 = y.left
        y.left = x
        x.right = T2
        x.height = 1 + max(self.getHeight(x.left), self.getHeight(x.right))
        y.height = 1 + max(self.getHeight(y.left), self.getHeight(y.right))

        return y

    def insert(self, root, rnum, name, cgpa):
        if root is None:
            return treeNode(rnum, name, cgpa)

        if rnum < root.rnum:
```

```
            root.left = self.insert(root.left, rnum, name, cgpa)
        elif rnum > root.rnum:
            root.right = self.insert(root.right, rnum, name, cgpa)
        else:
            return root

        root.height = 1 + max(self.getHeight(root.left), self.getHeight(root.right))
        balance = self.getBalance(root)

        if balance > 1 and rnum < root.left.rnum:
            return self.rightRotate(root)

        if balance < -1 and rnum > root.right.rnum:
            return self.leftRotate(root)

        if balance > 1 and rnum > root.left.rnum:
            root.left = self.leftRotate(root.left)
            return self.rightRotate(root)

        if balance < -1 and rnum < root.right.rnum:
            root.right = self.rightRotate(root.right)
            return self.leftRotate(root)

        return root

    def delete(self, root, rnum):
        if root is None:
            return root

        if rnum < root.rnum:
            root.left = self.delete(root.left, rnum)
        elif rnum > root.rnum:
            root.right = self.delete(root.right, rnum)
        else:
            if root.left is None:
                temp = root.right
                root = None
                return temp
            elif root.right is None:
                temp = root.left
                root = None
                return temp
            temp = self.getMinValueNode(root.right)
            root.rnum = temp.rnum
            root.right = self.delete(root.right, temp.rnum)
        if root is None:
            return root
        root.height = 1 + max(self.getHeight(root.left),self.getHeight(root.right))
        balance = self.getBalance(root)
```

OUTPUT:


VL Tree built from file:
n-order traversal:
oll Number: 102015022 Name: RandeepSingh CGPA: 7.5
oll Number: 102015045 Name: GURKIRATSINGH CGPA: 8.0
oll Number: 102015070 Name: ShradhaSood CGPA: 7.6
oll Number: 102015118 Name: Vaishnavi CGPA: 8.0
oll Number: 102015142 Name: SukritSethi CGPA: 7.9
oll Number: 102015161 Name: Ayushi CGPA: 8.3
oll Number: 102015185 Name: BHAVYADUTTA CGPA: 7.3
oll Number: 102065023 Name: HarmandeepSingh CGPA: 7.3
oll Number: 102065033 Name: AyushKhurana CGPA: 7.6