DSA Lab10

23K2001

M.Muzammil Siddiqui

BCS-3J

Q1:

```
// AVL trees keep themselves balanced, so they’re faster for searching, inserting,
// and deleting compared to regular binary trees. The balance makes them efficient,
// especially when data is constantly being added or removed.
```

```
Elements in the tree through preOrder traversal:
10 6 4 8

After conversion to AVL:
6 4 10 8

Elements in the tree through preOrder traversal:
10 16 12 18

After conversion to AVL:
16 10 12 18
PS F:\Semester Material - Muzammil\FAST-KHI-Semester-3\Data Structures (LAB)\
```

Q2:

```
Students in the tree through inOrder traversal:
Name: Muzammil  Roll#: 10
Name: Ali       Roll#: 20
Name: Armagahan Roll#: 30
Name: Danish    Roll#: 40
Name: Sohail    Roll#: 50

inOrder traversal after adding Student with Roll# 15:
Name: Muzammil  Roll#: 10
Name: Huzaima   Roll#: 15
Name: Ali       Roll#: 20
Name: Armagahan Roll#: 30
Name: Danish    Roll#: 40
Name: Sohail    Roll#: 50

Height of the AVL Tree: 3
PS F:\Semester Material - Muzammil\FAST-KHI-Semester-3\Data
```

Q3:

```
Elements in the tree through preOrder traversal:
50 30 20 40 70 60 80
preOrder traversal after adding node with value '55':
50 30 20 40 70 60 55 80
PS F:\Semester Material - Muzammil\FAST-KHI-Semester-3\Data
```

Q4:

```
Balance Factor of node with value '10' : 0
Balance Factor of node with value '5' : 1
Balance Factor of node with value '15' : 0
Balance Factor of node with value '3' : 1
Balance Factor of node with value '7' : 1

Elements in the tree through preOrder traversal:
10 5 3 7 15

Balance Factor of node with value '12' : 0

preOrder traversal after adding node with value '12':
10 5 3 7 15 12

Height of the AVL Tree: 3
PS F:\Semester Material - Muzammil\FAST-KHI-Semester-3\Data Structures (LAB)
```

Q5:

```
Elements in the tree through preOrder traversal:
10 5 3 7 15 12
Kth (k=2) Largest: 12

Kth (k=3) Smallest: 7
PS F:\Semester Material - Muzammil\FAST-KHI-Semester-3\Data
```