National University of Computer and Emerging Sciences



Lab Manual 08

Data Structures Lab

| Course Instructor | Dr. Zareen Alamgir |
| --- | --- |
| Lab Instructor (s) | Fariha Maqbool  Humna Shabir |
| Section | F |
| Semester | Fall 2022 |

Department of Computer Science

FAST-NU, Lahore, Pakistan

## Objectives

After performing this lab, students shall be able to:

* Learn and practice AVL trees.

Implement a template based AVL tree and create the functions described below:

**Task 1:**

* Create a function ‘**insert’** to insert a node into AVL keeping in view that the tree remains balanced after insertion.

For this task you need to compute the **Balancing Factor (BF)** of each node

Implement functions for four types of rotation namely **RR, LL, LR** and **RL**

Create a function to compute the BF of each node.

**Node \*RotateRight(Node \*y)**

**Node \*RotateLeft(Node \*x)**

**Int balanceFactor(Node \*n)**

**Node \*insert(Node \*node, int val)**

* A **‘search’** function to search an element in the tree **Bool search(int val)**
* A **‘delete’** function to delete the node containing the value
* **Constructor**, **copy constructor** and **destructor**
* A **‘print’** function to print the **inorder, preorder** and **postorder** traversal

A suitable main function to test the above functions.

**Note: You can make any helping functions to complete the above functions.**

**Task 2:**

Implement an AVL tree that can handle **duplicate values** and implement the following functions:

* Create a function **‘isAVL’** which takes a tree as argument and tells whether the tree is an AVL or not.
* A **recursive** function ‘**findmin’** that finds the minimum element using recursion.
* **Recursive** functions **‘find2ndmin’** that finds the 2nd minimum element and **‘find3rdmin’** that finds third minimum element in the tree.
* An **‘itprint’** function that prints the preorder traversal using **iterator**.