Ghulam Ishaq Khan Institute (GIKI)

Assignment # 4

Subject: Data Structures and Algorithms (DSA)	**Course Code:** CS - 221 - A - Fall - 25
Class: BS AI – Batch 34	**Submission Deadline:** 14-Dec-2025 – 11:59 PM
Course Instructor: M. Qasim Riaz - Lecturer - FCSE	**Total Marks:** 30

Note (Read notes & instructions first)

- First, read the instructions and statements of each exercise/question carefully then write the solution.
- **Answers to questions will be submitted in "Handwritten" or just "Code file" is mentioned in front of each question. So, submit it accordingly.**
- *For handwritten file:*
  - In case of multiple questions, give heading of each question's number or the exercise you are going to solve (don't write statement of question)
  - Then **scan all questions and pages of handwritten document > convert into pdf > upload at teams**
- *For C++ Code File:*
  - Create a different file for each question.
  - Write Question no in the name of file.
  - Merge all files into One Zip file and upload it in your class Team's group (in the assignment section of your notebook).
  - The name of each Zip file should contain your roll number & assignment number.
  - For Example, if your roll number is 2022532 and you are doing 2nd assignment then file name of your Zip file should be written as ---> 2022532_2.
  - Now upload all these files to Microsoft teams.

CHEATING/COPY CASE or LATE SUBMISSION will be graded as ZERO MARKS.

Question # 1: [Marks 10] – [Submit Code File] – Array Sorting (Counting Sort & Radix Sort)

Your Task is to understand & implement the logic and code of counting - sort and radix sort in arrays using the C++ as programming language. For the visual representation and understanding of the logic both types of sorting you can see its visual representation at following link:

Sorting (Bubble, Selection, Insertion, Merge, Quick, Counting, Radix) - VisuAlgo

Counting Sort - Data Structures and Algorithms Tutorials - GeeksforGeeks C++ Program For Radix Sort - GeeksforGeeks

Note: First implement the counting sort (it will work as pre-requisite) then implement the radix sort.

Question # 2: [Marks 20] – [Submit Code File] – Emergency Response Management System

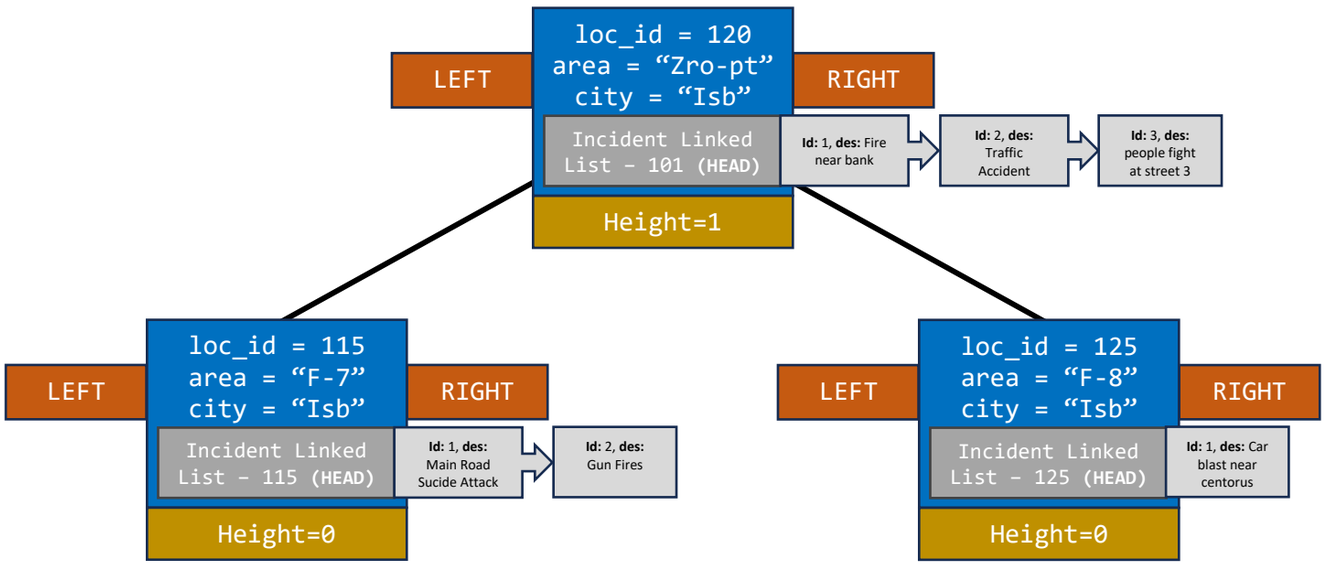
Government of Pakistan wants us to develop a system to keep record of emergency situations in different areas/locations in Islamabad city. The module we have to develop will be part of a greater **Emergency Response Management System (ERMS)** for Islamabad city. The system will overall manage emergency locations and incidents using both trees and linked list data structures. The city is represented as a **binary search tree (BST)**, with:

1. **Each node represents a location** in the city, identified by a unique integer ID (e.g., area codes).
2. **Each location contains a list of incidents**, where incidents are stored in a linked list.
3. You are required to create **Tree structure for locations**
    1. Implement an AVL Tree to store city locations to ensure balance and efficient searching.
    2. Each node in the AVL Tree represents a location and stores:
    3. Location ID (unique integer)
    4. This location_id will be used to judge that where that node should be place in BST
    5. A pointer to the head of a **singly linked list** for incidents at that location.
4. You are required to create **Linked lists for incident records at a specific location**
    1. Each location node maintains a linked list of incidents.
    2. Each incident in the list has an Incident ID (unique integer) and Description (e.g., "Fire at street X").
    3. Implement functions to: Add an incident to the linked list of a specific location, Delete an incident by its ID for a given location, Display all incidents at a location.

5. Operations on the Emergency Response System

1. Add a new location to the AVL Tree.
2. Remove a location from the AVL Tree (and all associated incidents).
3. Add a new incident to a location.
4. Delete an incident for a given location.
5. Display all locations and their associated incidents using an in-order traversal of the AVL Tree.

Note: Ensure the AVL Tree is correctly balanced after every insertion and deletion operation. Linked list operations (insertion, deletion, and traversal) should be implemented.



--- Good Luck ---