**QUAID E AWAM UNIVERSITY OF ENIGINEERING SCIENCE AND TECHNOLOYG**

**Week 1–2: Basics**

1. **Introduction to Python for Data Structures**
   * Variables, loops, conditionals
   * Lists, tuples, dictionaries
   * Problem: Store and retrieve student records.
2. **Complexity Analysis Lab**
   * Write code to measure execution time of sorting/searching.
   * Compare operations in O(1), O(log n), O(n).

**Week 3–5: Linear Data Structures**

1. **Arrays & Lists**
   * Implement dynamic arrays.
   * Problem: Reverse array, rotate array, merge two arrays.
2. **Stacks**
   * Implement stack using list & linked list.
   * Problems: Balanced parentheses checker, postfix expression evaluation.
3. **Queues**
   * Implement queue & circular queue.
   * Problems: Bank queue simulation, producer-consumer problem.
4. **Deque & Priority Queue**
   * Double-ended queue using collections.
   * Priority queue using heapq.

**Week 6–8: Non-linear Data Structures**

1. **Linked Lists**
   * Singly, doubly, and circular linked list.
   * Problems: Insertion, deletion, reversal, detect cycle.
2. **Trees**
   * Binary Tree construction & traversal (inorder, preorder, postorder, level-order).
   * Problems: Height of tree, count leaf nodes.
3. **Binary Search Tree (BST)**
   * Insertion, deletion, searching.
   * Applications: Dictionary search, AI decision trees introduction.
4. **Heaps**
   * Max-heap, min-heap.
   * Problem: Heap sort.

**Week 9–10: Advanced Structures**

1. **Graphs**
   * Represent graph (adjacency list, adjacency matrix).
   * BFS, DFS implementations.
   * Applications: Path finding, social network analysis.
2. **Shortest Path Algorithms**
   * Dijkstra’s algorithm.
   * A\* introduction (link to AI pathfinding).

**Week 11–12: Hashing & Maps**

1. **Hash Tables**
   * Implement hash map with collision handling.
   * Application: Word frequency counter.
2. **Tries (Prefix Trees)** *(Optional but useful in AI)*
   * Implement trie for dictionary lookups.
   * Application: Autocomplete feature.

**Week 13–14: Integration with AI**

1. **Using Data Structures in AI**
   * Represent a search problem with trees & graphs.
   * Implement BFS/DFS for state space search.
2. **Mini Project Lab**
   * Examples:
     + Pathfinding in a maze using BFS/DFS.
     + Spell checker using Tries & Hashing.
     + Priority scheduling using heap.