|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE** | | | | | **DEPARTMENT OF COMPUTER SCIENCE ENGINEERING** | | | | |
| **Program Name:** B. Tech | | | | **Assignment Type: Lab** | | | **Academic Year:**2025-2026 | | |
| **Course Coordinator Name** | | | | Dr. Rishabh Mittal | | | | | |
| **Instructor(s) Name** | | | | |  | | --- | | Mr. S Naresh Kumar | | Ms. B. Swathi | | Dr. Sasanko Shekhar Gantayat | | Mr. Md Sallauddin | | Dr. Mathivanan | | Mr. Y Srikanth | | Ms. N Shilpa | | Dr. Rishabh Mittal (Coordinator) | | Dr. R. Prashant Kumar | | Mr. Ankushavali MD | | Mr. B Viswanath | | Ms. Sujitha Reddy | | Ms. A. Anitha | | Ms. M.Madhuri | | Ms. Katherashala Swetha | | Ms. Velpula sumalatha | | Mr. Bingi Raju | | | | | | |
| **CourseCode** | | | 23CS002PC304 | **Course Title** | | AI Assisted Coding | | | |
| **Year/Sem** | | | III/II | **Regulation** | | R23 | | | |
| **Date and Day**  **of Assignment** | | | **Week6 – Monday** | **Time(s)** | | 23CSBTB01 To 23CSBTB52 | | | |
| **Duration** | | | 2 Hours | **Applicable to**  **Batches** | | All batches | | | |
| **Assignment Number: 11.1**(Present assignment number)/**24**(Total number of assignments) | | | | | | | | | |
|  | | | | | | | | | |
|  | **Q.No.** | **Question** | | | | | | ***Expected Time***  ***to complete*** |  |
|  | 1 | **Lab 11 – Data Structures with AI: Implementing Fundamental Structures**  **Lab Objectives**   * Use AI to assist in designing and implementing fundamental data structures in Python. * Learn how to prompt AI for structure creation, optimization, and documentation. * Improve understanding of Lists, Stacks, Queues, Linked Lists, Trees, Graphs, and Hash Tables. * Enhance code quality with AI-generated comments and performance suggestions.   **Task Description #1 – Stack Implementation**  Task: Use AI to generate a Stack class with push, pop, peek, and is\_empty methods.  Sample Input Code:  class Stack:  pass  Expected Output:   * A functional stack implementation with all required methods and docstrings.   **Task Description #2 – Queue Implementation**  Task: Use AI to implement a Queue using Python lists.  Sample Input Code:  class Queue:  pass  Expected Output:   * FIFO-based queue class with enqueue, dequeue, peek, and size methods.   **Task Description #3 – Linked List**  Task: Use AI to generate a Singly Linked List with insert and display methods.  Sample Input Code:  class Node:  pass  class LinkedList:  pass  Expected Output:   * A working linked list implementation with clear method documentation.   **Task Description #4 – Binary Search Tree (BST)**  Task: Use AI to create a BST with insert and in-order traversal methods.  Sample Input Code:  class BST:  pass  Expected Output:   * BST implementation with recursive insert and traversal methods.   **Task Description #5 – Hash Table**  Task: Use AI to implement a hash table with basic insert, search, and delete methods.  Sample Input Code:  class HashTable:  pass  Expected Output:   * Collision handling using chaining, with well-commented methods.   **Task Description #6 – Graph Representation**  Task: Use AI to implement a graph using an adjacency list.  Sample Input Code:  class Graph:  pass  Expected Output:   * Graph with methods to add vertices, add edges, and display connections.   **Task Description #7 – Priority Queue**  Task: Use AI to implement a priority queue using Python’s heapq module.  Sample Input Code:  class PriorityQueue:  pass  Expected Output:   * Implementation with enqueue (priority), dequeue (highest priority), and display methods.   **Task Description #8 – Deque**  Task: Use AI to implement a double-ended queue using collections.deque.  Sample Input Code:  class DequeDS:  pass  Expected Output:   * Insert and remove from both ends with docstrings.   **Task Description #9 Real-Time Application Challenge – Choose the Right Data Structure**  **Scenario:** Your college wants to develop a Campus Resource Management System that handles:   1. Student Attendance Tracking – Daily log of students entering/exiting the campus. 2. Event Registration System – Manage participants in events with quick search and removal. 3. Library Book Borrowing – Keep track of available books and their due dates. 4. Bus Scheduling System – Maintain bus routes and stop connections. 5. Cafeteria Order Queue – Serve students in the order they arrive.   **Student Task:**   * For each feature, select the most appropriate data structure from the list below:   + Stack   + Queue   + Priority Queue   + Linked List   + Binary Search Tree (BST)   + Graph   + Hash Table   + Deque * Justify your choice in 2–3 sentences per feature. * Implement one selected feature as a working Python program with AI-assisted code generation.   Expected Output:   * A table mapping feature → chosen data structure → justification. * A functional Python program implementing the chosen feature with comments and docstrings.   **Task Description #10: Smart E-Commerce Platform – Data Structure Challenge**  An e-commerce company wants to build a Smart Online Shopping System with:   1. Shopping Cart Management – Add and remove products dynamically. 2. Order Processing System – Orders processed in the order they are placed. 3. Top-Selling Products Tracker – Products ranked by sales count. 4. Product Search Engine – Fast lookup of products using product ID. 5. Delivery Route Planning – Connect warehouses and delivery locations.   **Student Task:**   * For each feature, select the most appropriate data structure from the list below:   + Stack   + Queue   + Priority Queue   + Linked List   + Binary Search Tree (BST)   + Graph   + Hash Table   + Deque * Justify your choice in 2–3 sentences per feature. * Implement one selected feature as a working Python program with AI-assisted code generation.   Expected Output:   * A table mapping feature → chosen data structure → justification. * A functional Python program implementing the chosen feature with comments and docstrings. | | | | | | Week6 -Monday |  |