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		Venkataramana Veeramsetty	
Instructor(s) Name		Dr. V. Venkataramana (Co-ordinator) Dr. T. Sampath Kumar Dr. Pramoda Patro Dr. Brij Kishor Tiwari Dr.J.Ravichander Dr. Mohammand Ali Shaik Dr. Anirodh Kumar Mr. S.Naresh Kumar Dr. RAJESH VELPULA Mr. Kundhan Kumar Ms. Ch.Rajitha Mr. M Prakash Mr. B.Raju Intern 1 (Dharma teja) Intern 2 (Sai Prasad) Intern 3 (Sowmya) NS_2 (Mounika)	
Course Code	24CS002PC215	**Course Title**	AI Assisted Coding
Year/Sem	II/I	**Regulation**	R24
Date and Day of Assignment	Week6 - Monday	**Time(s)**	
Duration	2 Hours	**Applicable to Batches**	
Assignment Number:11.5(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,	Week 6 - Friday

Graphs, and Hash Tables.
- Enhance code quality with AI-generated comments and performance suggestions.

Task 1: Smart Contact Manager (Arrays & Linked Lists)
Scenario:
SR University's student club wants a simple **Contact Manager App** to store members' names and phone numbers. The app should allow adding, searching, and deleting contacts efficiently.

- Use **arrays** to store contacts initially.
- Implement the same system using a **linked list** for dynamic memory allocation.
- Compare both approaches (array vs. linked list) in terms of insertion and deletion efficiency.
- Use **GitHub Copilot** suggestions to implement search and delete methods

Task 2: Emergency Help Desk (Stack Implementation)
Scenario:
SR University's IT Help Desk receives **support tickets** from students and staff. Since urgent issues need to be resolved in the order they were received, but escalation requires "last in, first out," a **stack-based system** is ideal.

- Implement a **stack** to handle support tickets.
- Provide operations: push(ticket), pop(), and peek().
- Simulate at least 5 tickets arriving and being resolved.
- Use **Copilot AI** to suggest additional stack operations (like checking if stack is empty or full).

Task 3: Library Book Search (Queues & Priority Queues)
Scenario:
The SRU Library system manages book borrow requests. Students join a **queue** when they request books. However, faculty requests should be given higher priority.

- Implement a **queue** for book requests (FIFO).
- Extend it to a **priority queue** where faculty members' requests are served before students.
- Use **Copilot** to generate enqueue and dequeue methods.
- Test with a mix of student and faculty requests.

	Task 4: Navigation Assistant (Trees & Graphs) **Scenario:** The university's navigation app helps new students find classrooms. Buildings and rooms are represented as **nodes** connected by **paths**. A **graph** or **tree** structure can model this system. • Create a **binary search tree (BST)** to store building names in alphabetical order. • Implement insert, search, and traversal (inorder, preorder, postorder) using **Copilot**. • Extend the system into a **graph** representation of rooms and paths. • Implement a shortest path algorithm (like BFS) with Copilot's assistance.	