AI ASSISTED CODING

LAB ASSIGNMENT: 11.2

NAME:P.NISHANTH

HTNO:2403A52066

Batch No:-03

TASK: Stack Implementation

Task: Use AI to generate a Stack class with push, pop, peek, and is\_empty methods. Sample Input Code: class Stack: pass.

PROMPT:

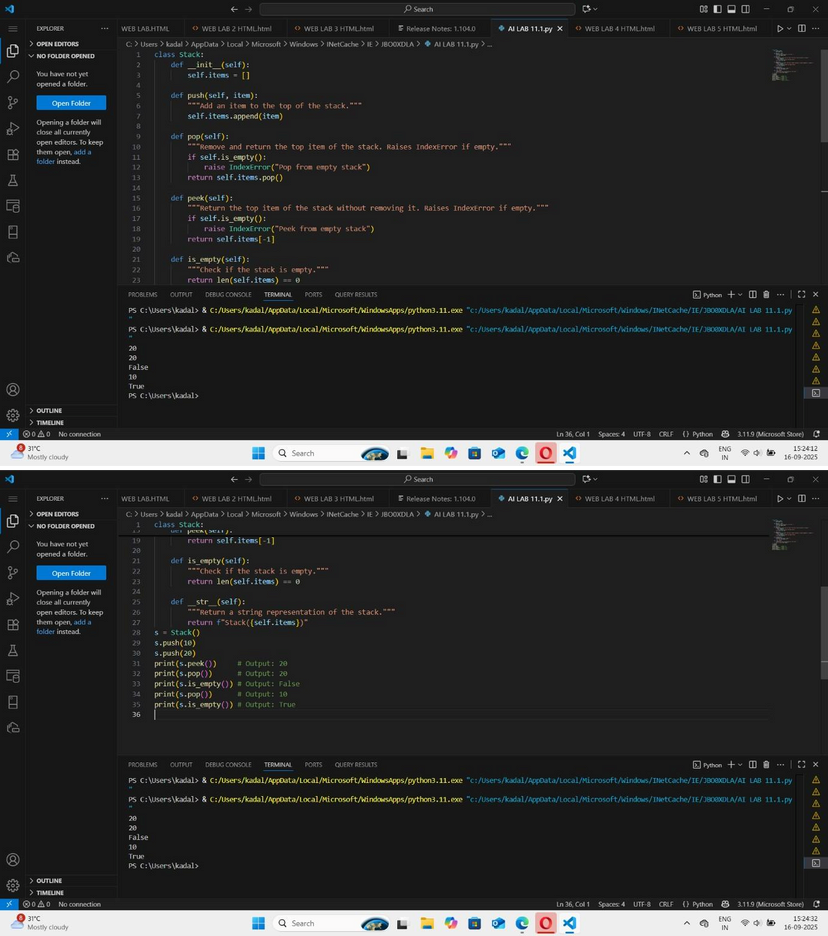
Generate python code and stack

Implementation

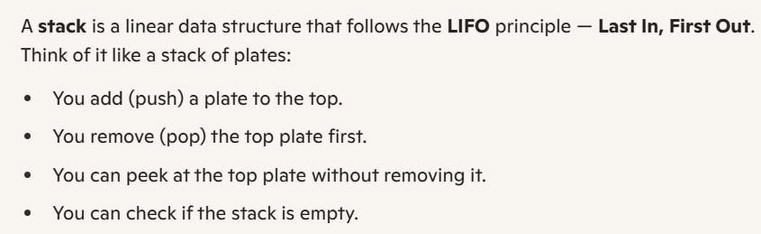
Task: Use AI to generate a Stack class with push, pop, peek, and is\_empty

methods. Sample Input Code: class Stack: pass.

CODE & OUTPUT:



EXPLAINATION:



TASK 2:

Queue Implementation

Task: Use AI to implement a Queue using Python lists.

Sample Input Code: class Queue: pass.

PROMPT:

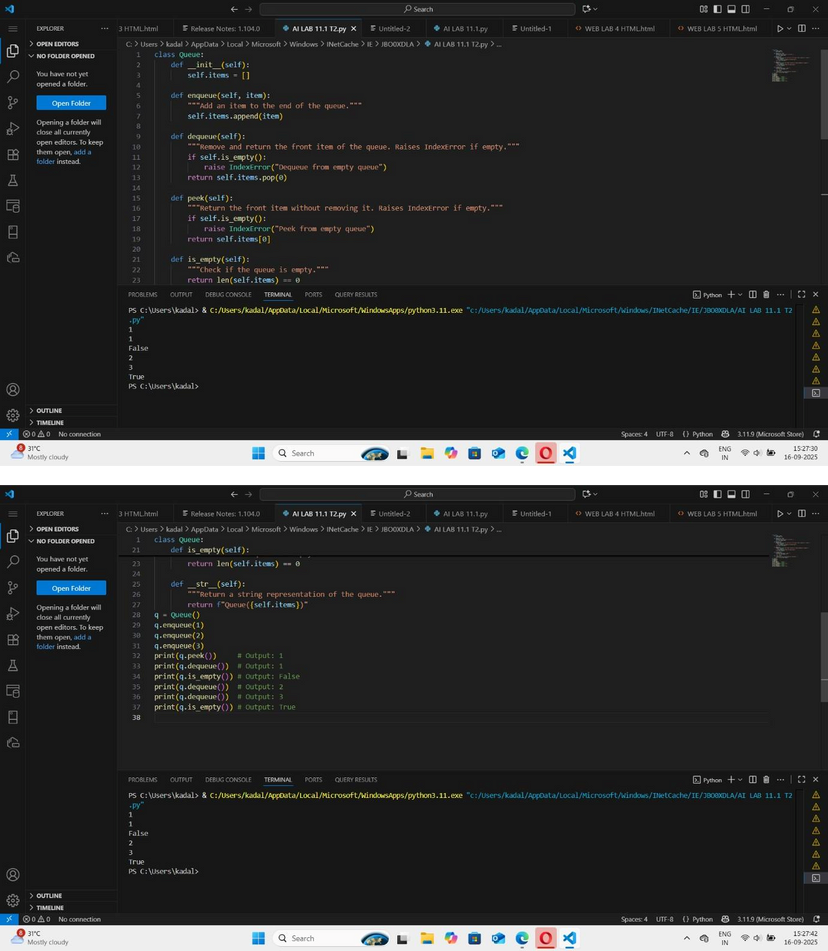
Generate python code and queue

Implementation

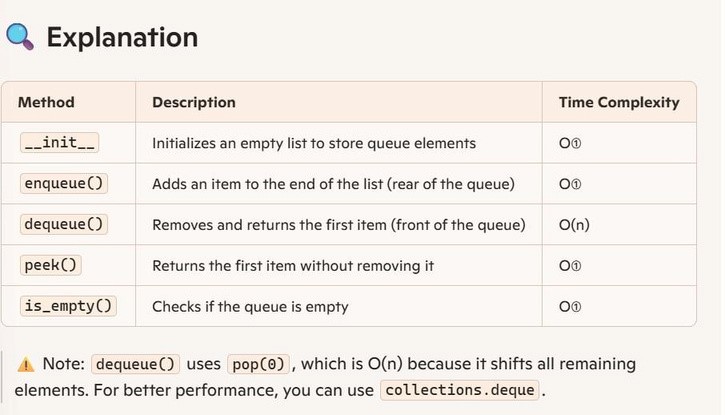
Task: Use AI to implement a Queue using Python lists.

Sample Input Code: class Queue: pass.

CODE & OUTPUT:



EXPLAINATION:



TASK 3:

Linked List

Task: Use AI to generate a Singly Linked List with insert and display methods.

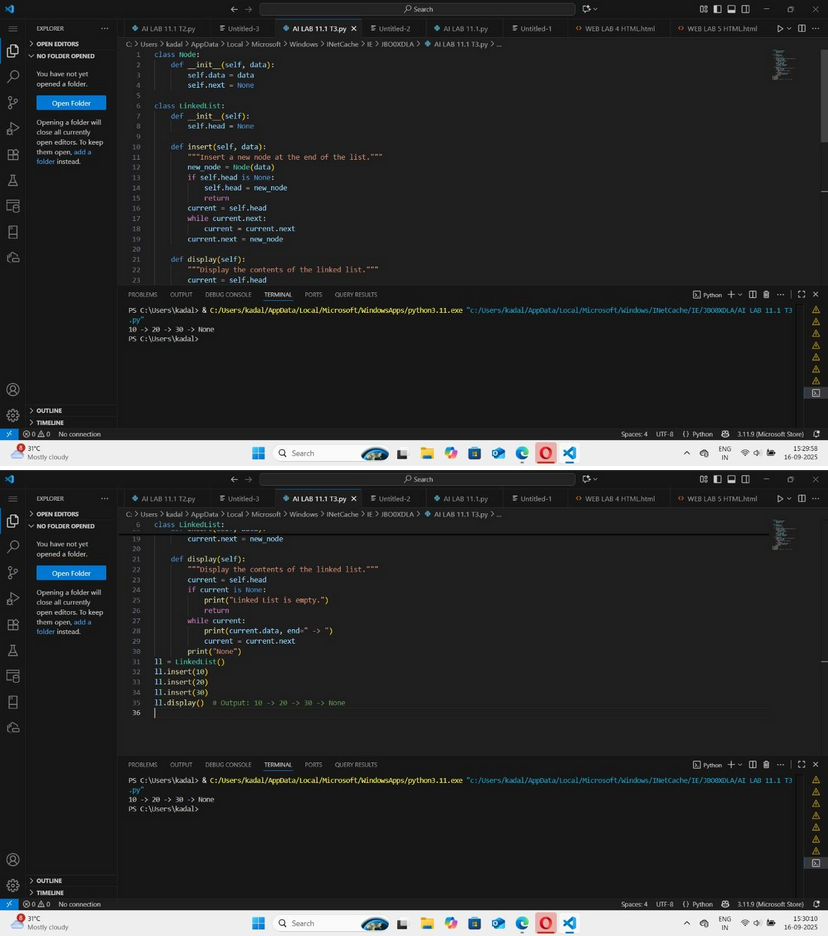
Sample Input Code: class Node: pass.

PROMPT:

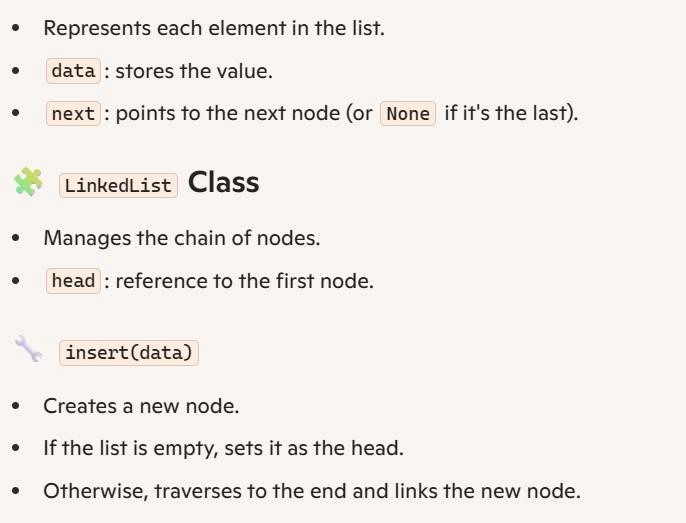
Generate python code and linked List Task: Use AI to generate a Singly Linked List with insert and display methods.

Sample Input Code: class Node: pass.

CODE & OUTPUT:



EXPLAINATION:



TASK 4:

Binary Search Tree (BST)

Task: Use AI to create a BST with insert and inorder traversal methods.

Sample Input Code: class BST: pass.

PROMPT:

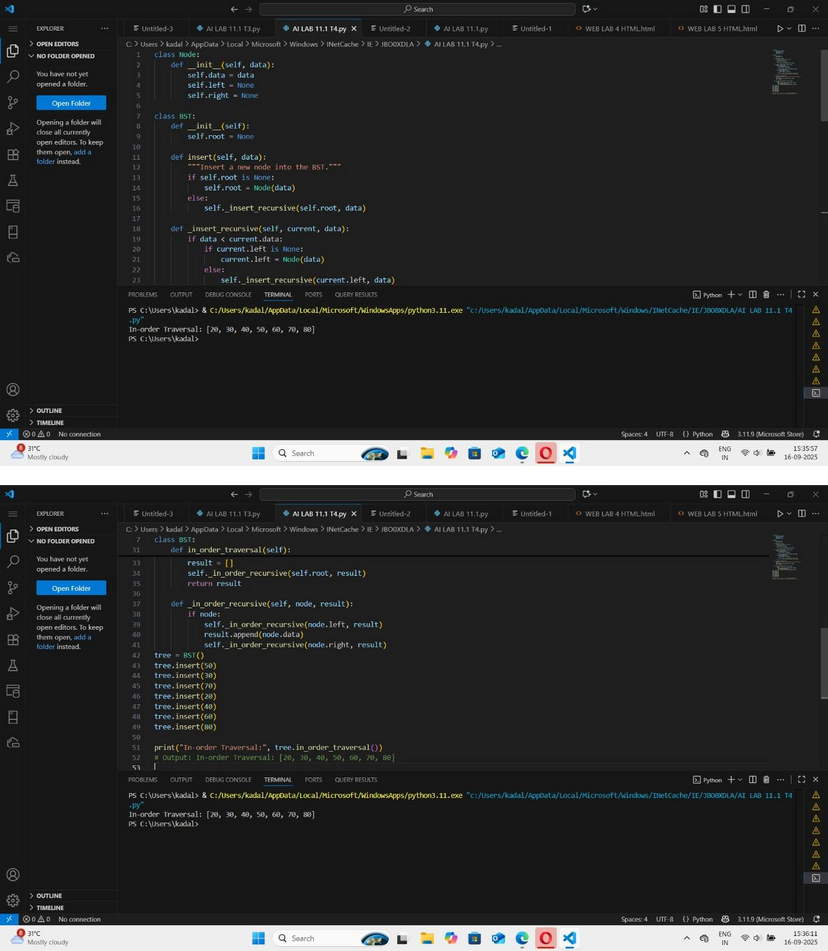
Generate python code and binary Search Tree

(BST)

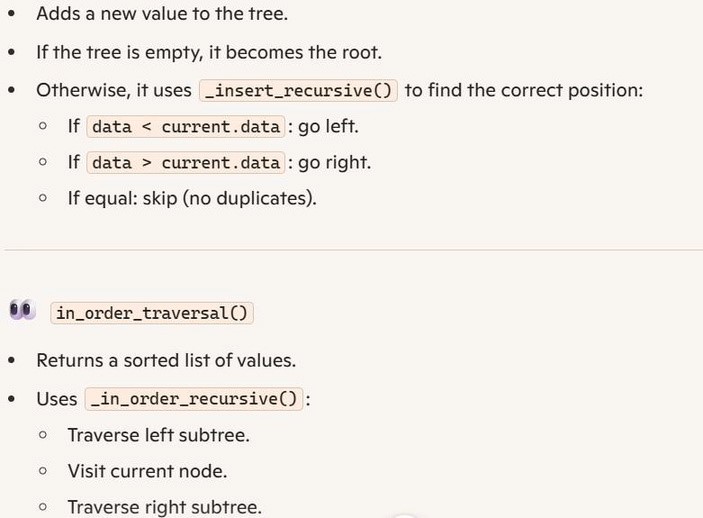
Task: Use AI to create a BST with insert and inorder traversal methods.

Sample Input Code: class BST: pass.

CODE & OUTPUT:



EXPLAINATION:



TASK 5:

Hash Table

Task: Use AI to implement a hash table with basic insert, search, and delete methods.

Sample Input Code: class HashTable: pass.

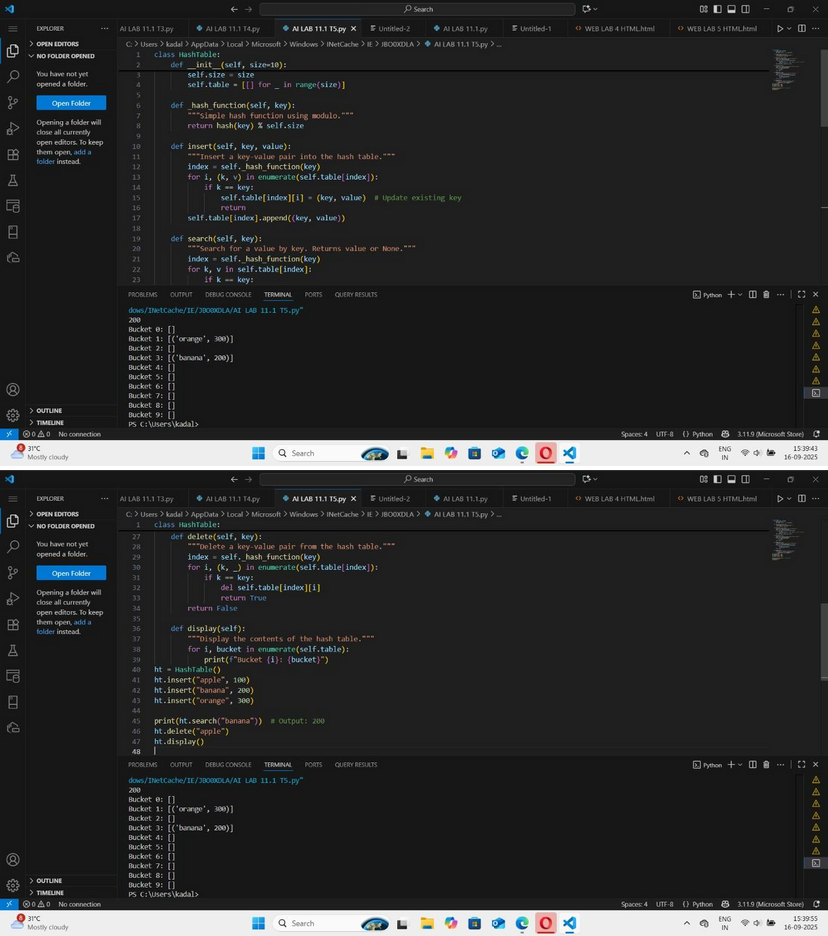
PROMPT:

Generate python code and hash Table Task: Use AI to implement a hash table with

basic insert, search, and delete methods.

Sample Input Code: class HashTable: pass.

CODE & OUTPUT:



EXPLAINATION:



TASK 6:

Graph Representation

Task: Use AI to implement a graph using an adjacency list.

Sample Input Code: class Graph: pass.

PROMPT:

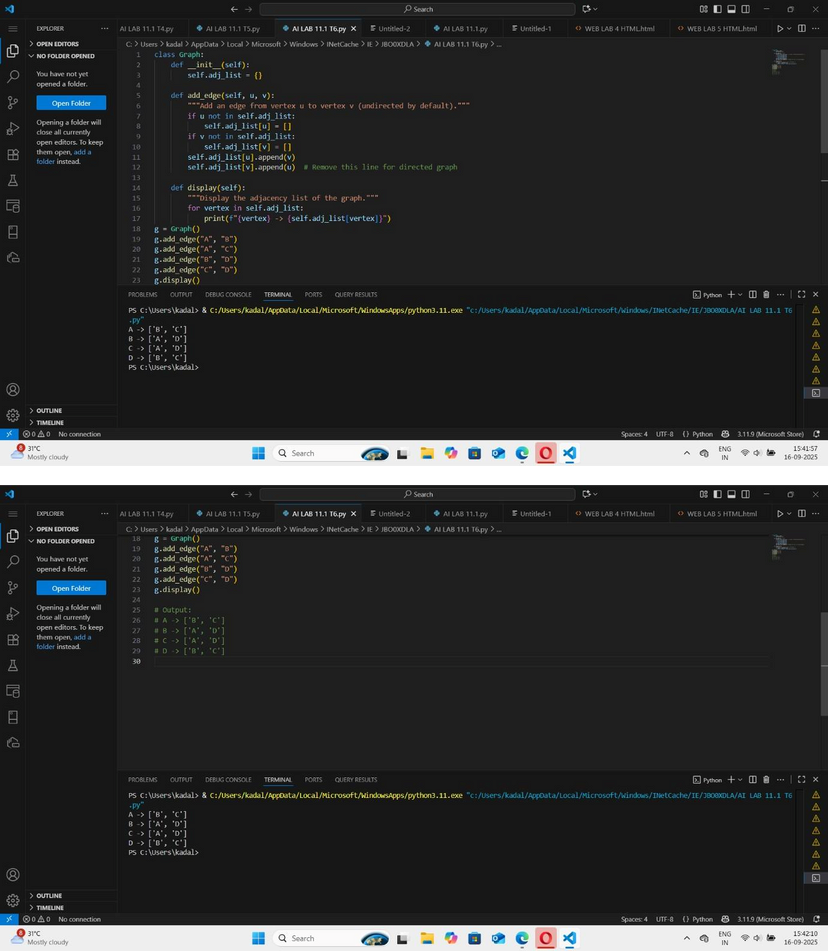
Generate python code and graph

Representation

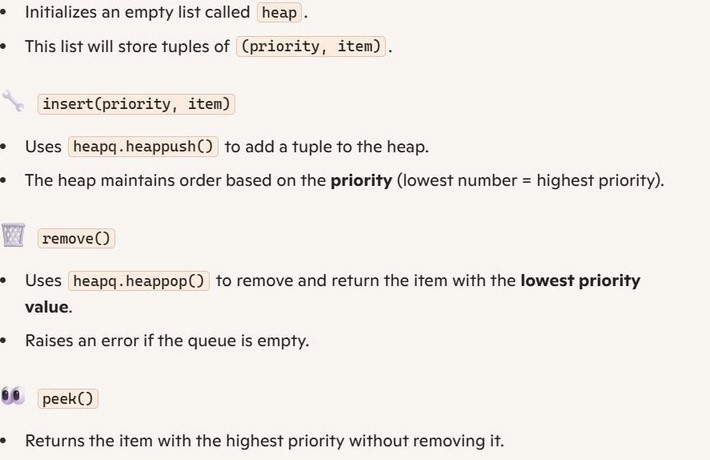
Task: Use AI to implement a graph using an adjacency list.

Sample Input Code: class Graph: pass.

CODE & OUTPUT:



EXPLAINATION:



TASK 7:

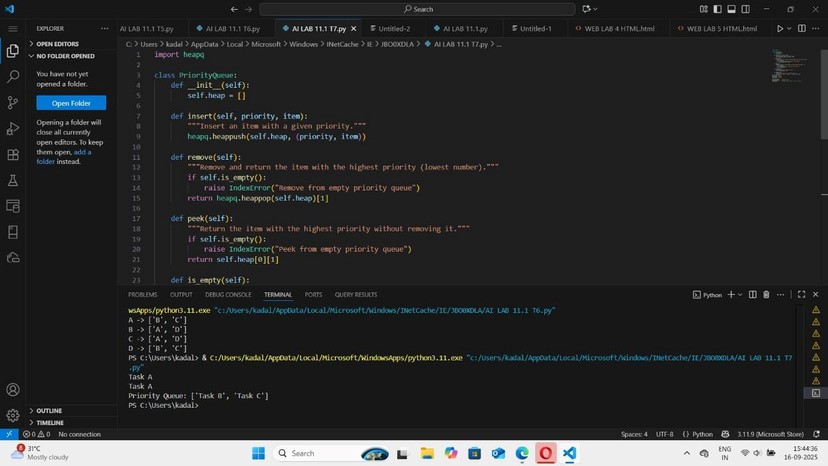
Priority Queue

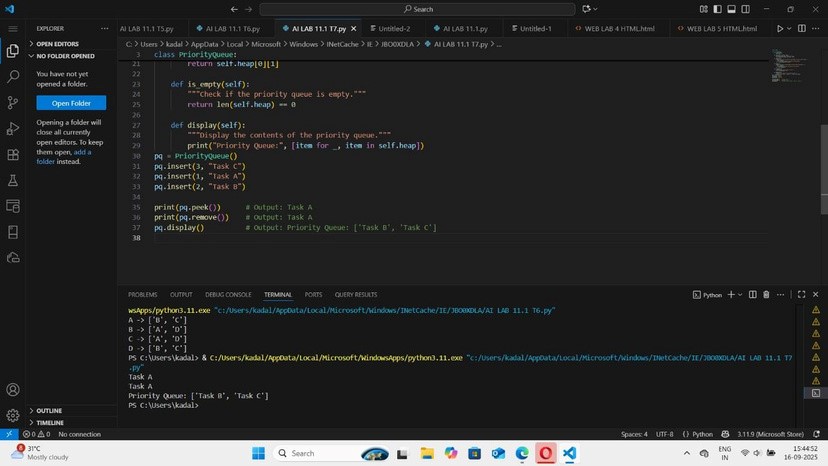
Task: Use AI to implement a priority queue using Python’s heapq module. Sample Input Code: class PriorityQueue: pass.

PROMPT:

Generate a python code and priority Queue Task: Use AI to implement a priority queue using Python’s heapq module. Sample Input Code: class PriorityQueue: pass.

CODE & OUTPUT:





EXPLAINATION:



TASK 8:

Deque

Task: Use AI to implement a double-ended queue using collections.deque.

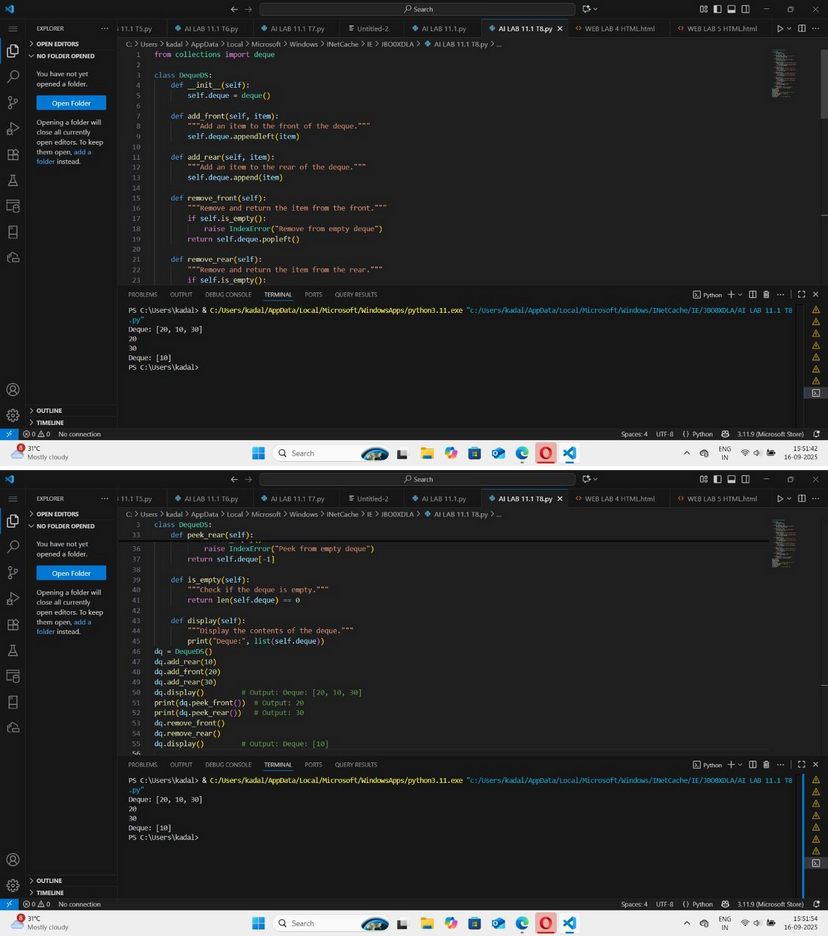
Sample Input Code: class DequeDS: pass.

PROMPT:

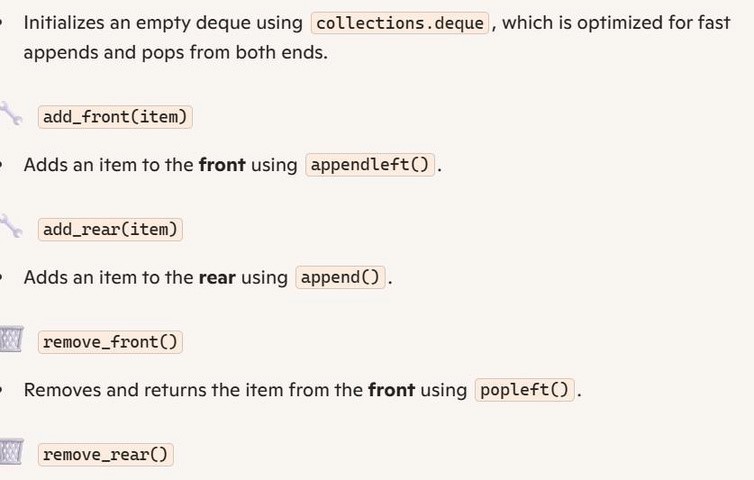
Generate python code and deque Task: Use AI to implement a double-ended queue using collections.deque.

Sample Input Code: class DequeDS: pass.

CODE &OUTPUT:



EXPLAINATION:



TASK 9:

AI-Generated Data Structure Comparisons Task: Use AI to generate a comparison table of different data structures (stack, queue,

linked list, etc.) including time complexities.

Sample Input Code:

# No code, prompt AI for a data structure comparison table.

PROMPT:

Generate python code and AI-Generated Data

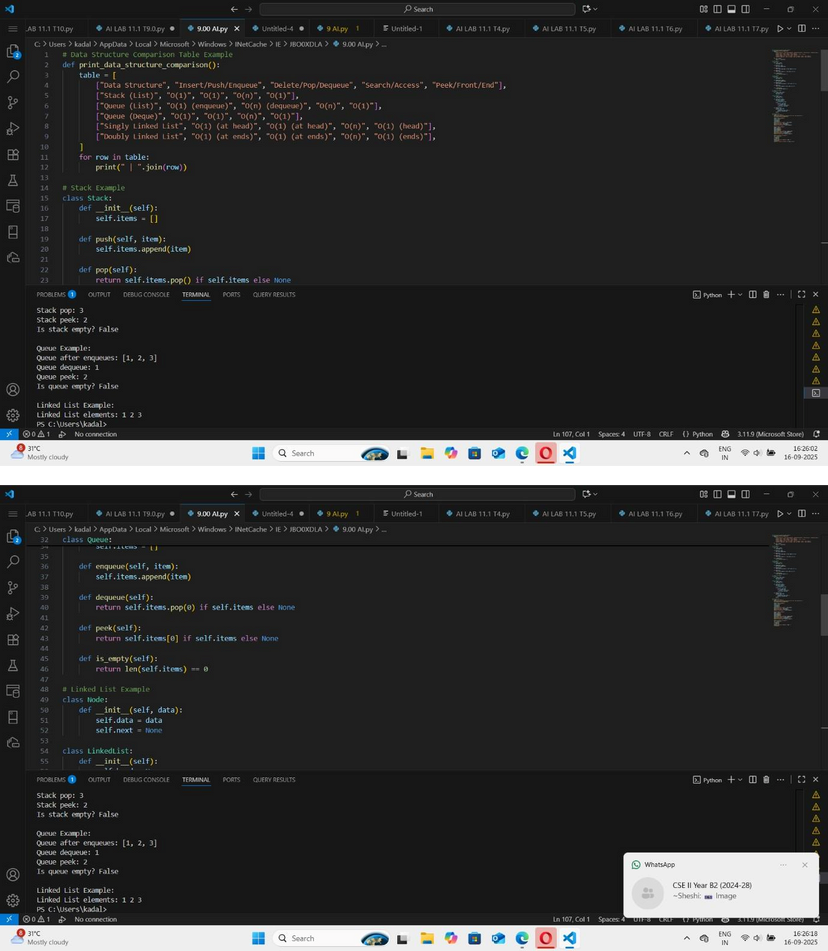
Structure Comparisons

Task: Use AI to generate a comparison table of different data structures (stack, queue, linked list, etc.) including time complexities.

Sample Input Code:

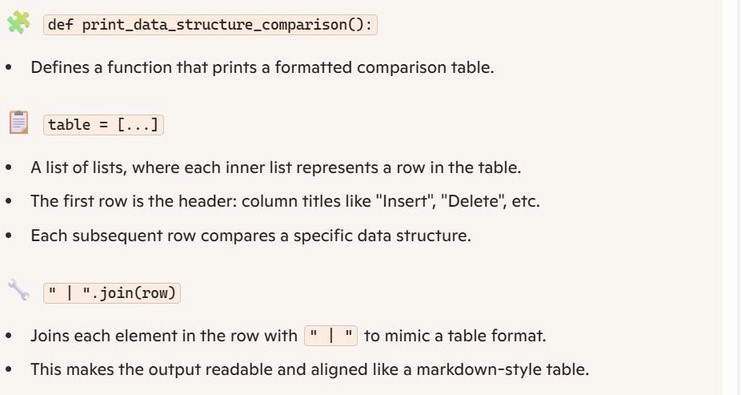
# No code, prompt AI for a data structure comparison table.

CODE & OUTPUT:





EXPLAINATION:



TASK 10:

Task Description #10 Real-Time Application Challenge – Choose the Right Data Structure Scenario:

Your college wants to develop a Campus Resource Management System that handles: 1. StudentAttendanceTracking–Dailylog of students entering/exiting the campus. 2. EventRegistrationSystem–Manage participants in events with quick search and removal.

1. LibraryBookBorrowing–Keeptrackofavailable books and their due dates.
2. BusSchedulingSystem–Maintainbusroutes and stop connections.
3. CafeteriaOrderQueue–Servestudentsin the order they arrive.

Student Task:

• For each feature, select the most appropriate data structure from the list below: o Stack o Queue o Priority

Queue o Linked List

* Binary Search Tree (BST)
* Graph o Hash Table oDeque
* Justifyyourchoicein2–3sentencesperfeature.
* Implementoneselectedfeatureasaworking Python program with AI- assisted code generation.

PROMPT:

Generate python code and task Description #10 Real-Time Application Challenge – Choose the

Right Data Structure Scenario:

Your college wants to develop a Campus Resource Management System that handles: 1. StudentAttendanceTracking–Dailylog of students entering/exiting the campus.

1. EventRegistrationSystem–Manageparticipants in events with quick search and removal.
2. LibraryBookBorrowing–Keeptrackofavailable books and their due dates.
3. BusSchedulingSystem–Maintainbusroutes and stop connections.
4. CafeteriaOrderQueue–Servestudentsin the order they arrive.

Student Task:

• For each feature, select the most appropriate data structure from the list below: o Stack o Queue o Priority Queue o Linked List o Binary Search Tree (BST)

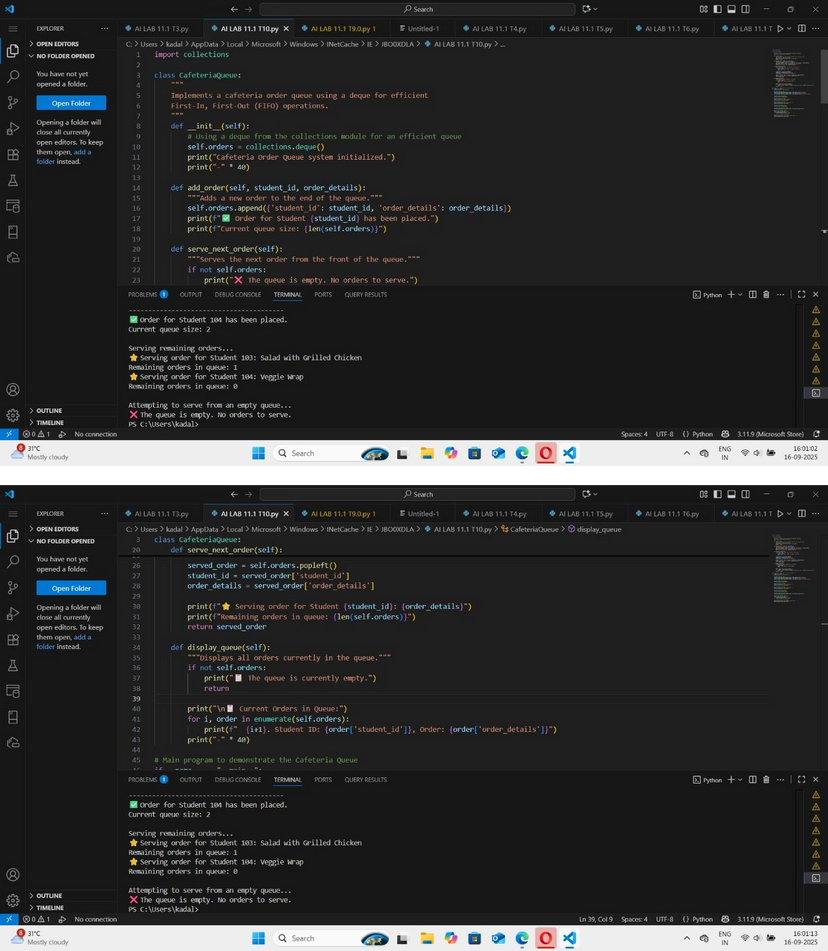
o Graph o

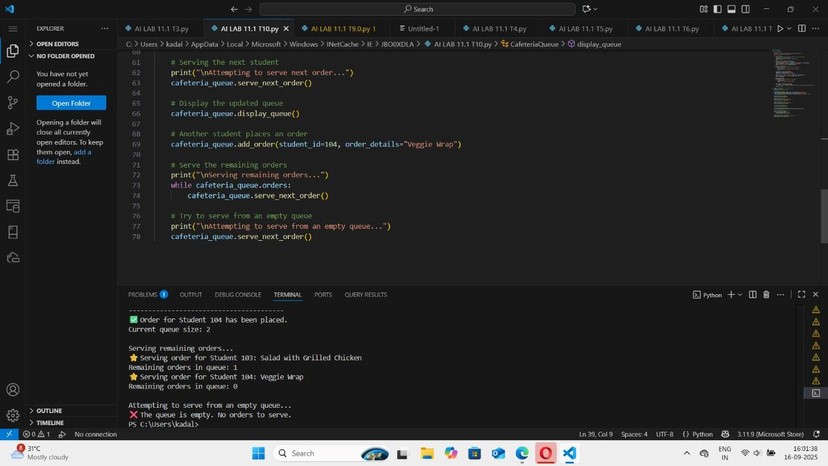
Hash Table o

Deque

* Justifyyourchoicein2–3sentencesperfeature.
* Implementoneselectedfeatureasaworking Python program with AI- assisted code generation.

CODE & OUTPUT:





EXPLAINATION:

