AI ASSISTED CODING

LAB-11.2

NAME:MANIKANTA.MILKURI

ENROLL.NO:2403A52072

BATCH:04

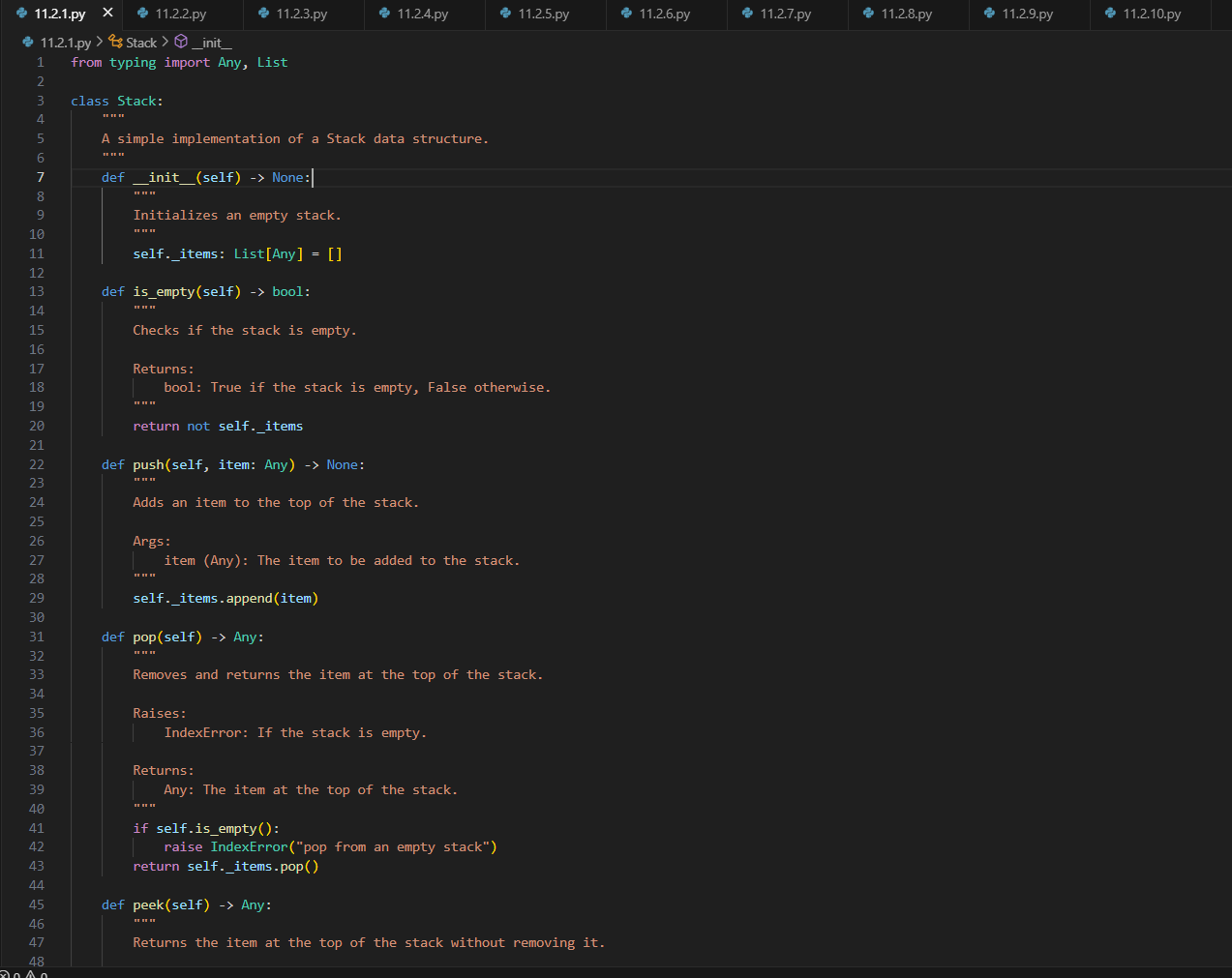
**TASK-01:**

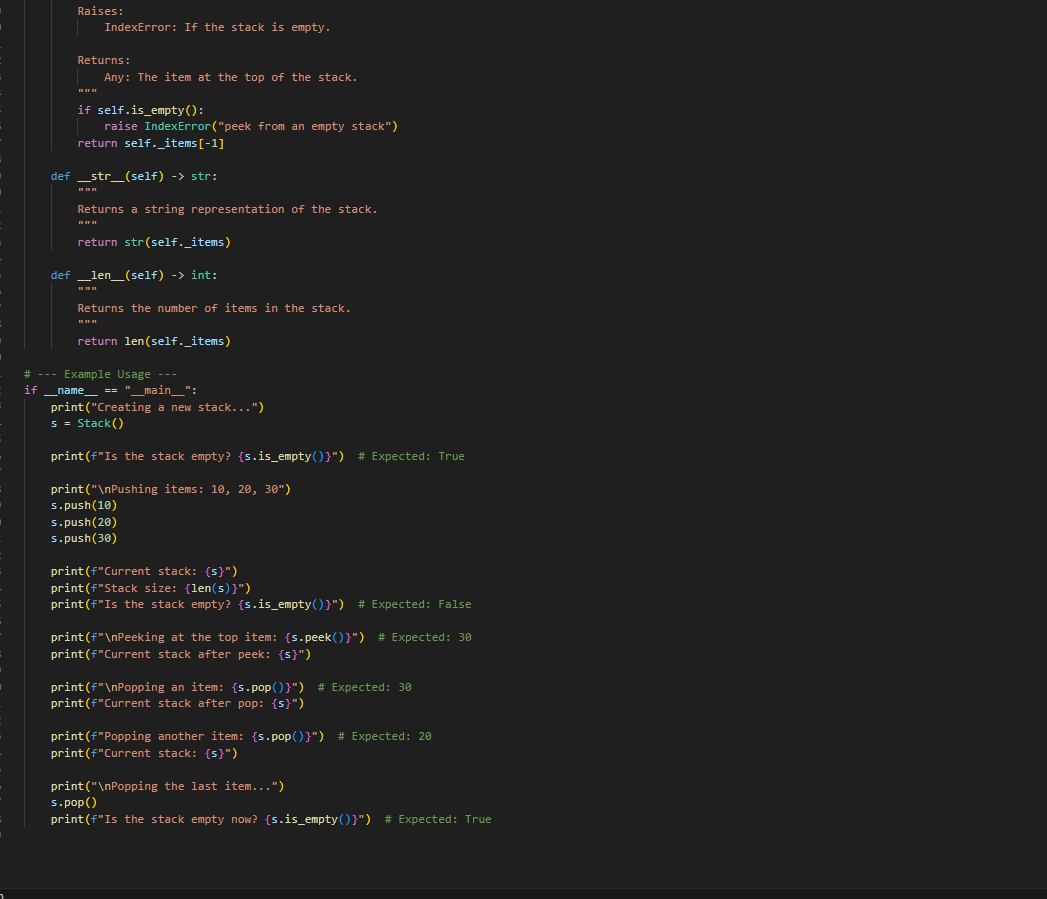
Use AI to generate a Stack class with push, pop, peek, and is\_empty methods.  
Sample Input Code:  
class Stack:  
pass

**PROMPT:**

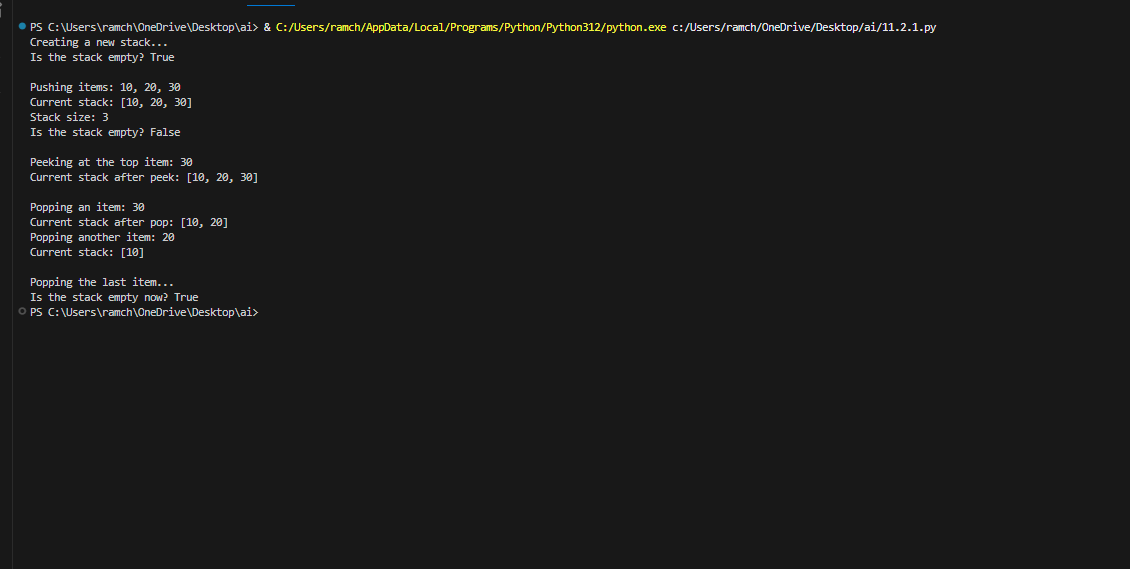
Generate a python stack class using push ,pop, is empty.

**CODE:**





**OUTPUT:**

****

**OBSERVATION:**

The AI generated the code in an efficient way according to the prompt as it developed the stack class.

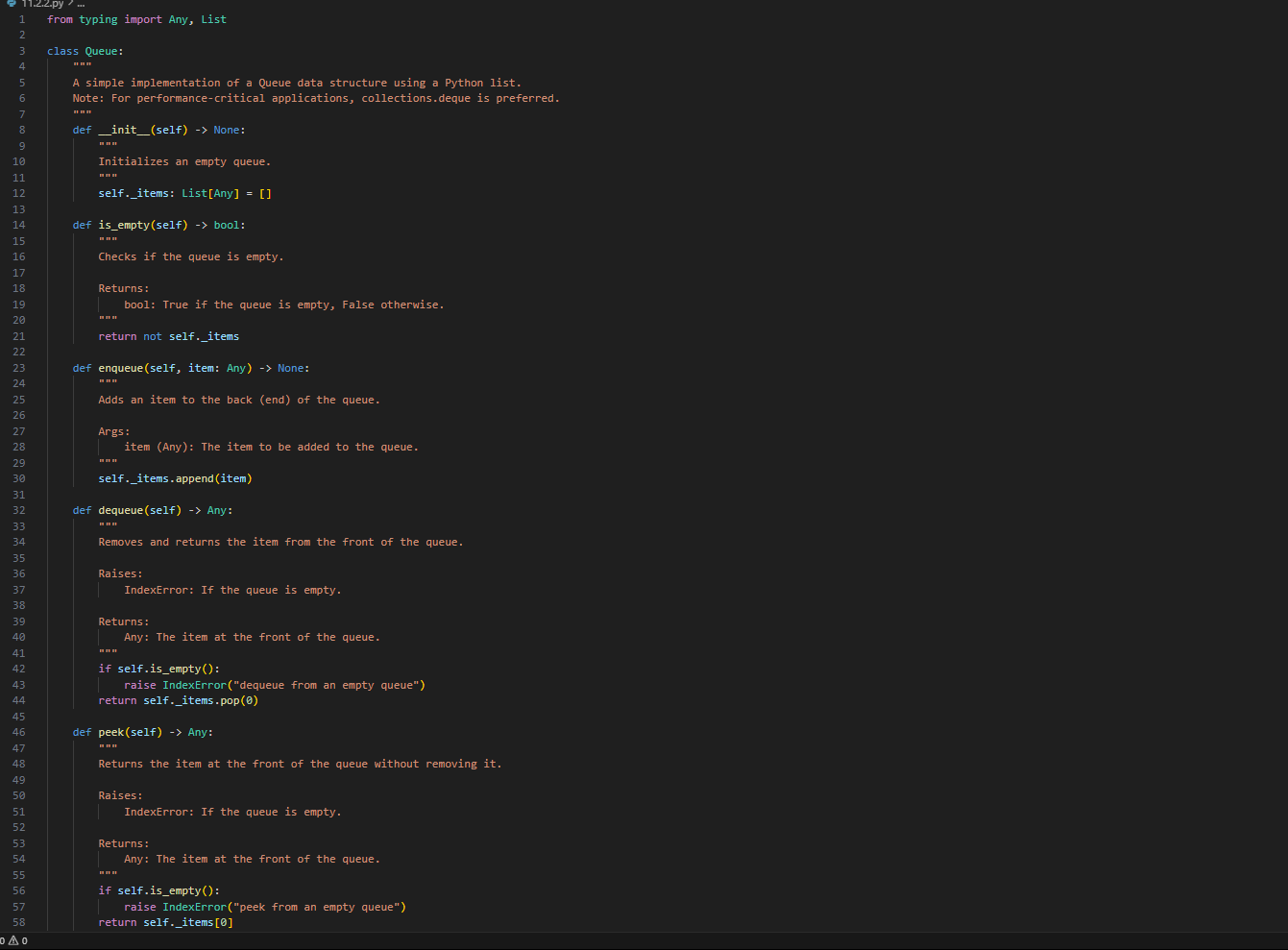
**TASK-02:**

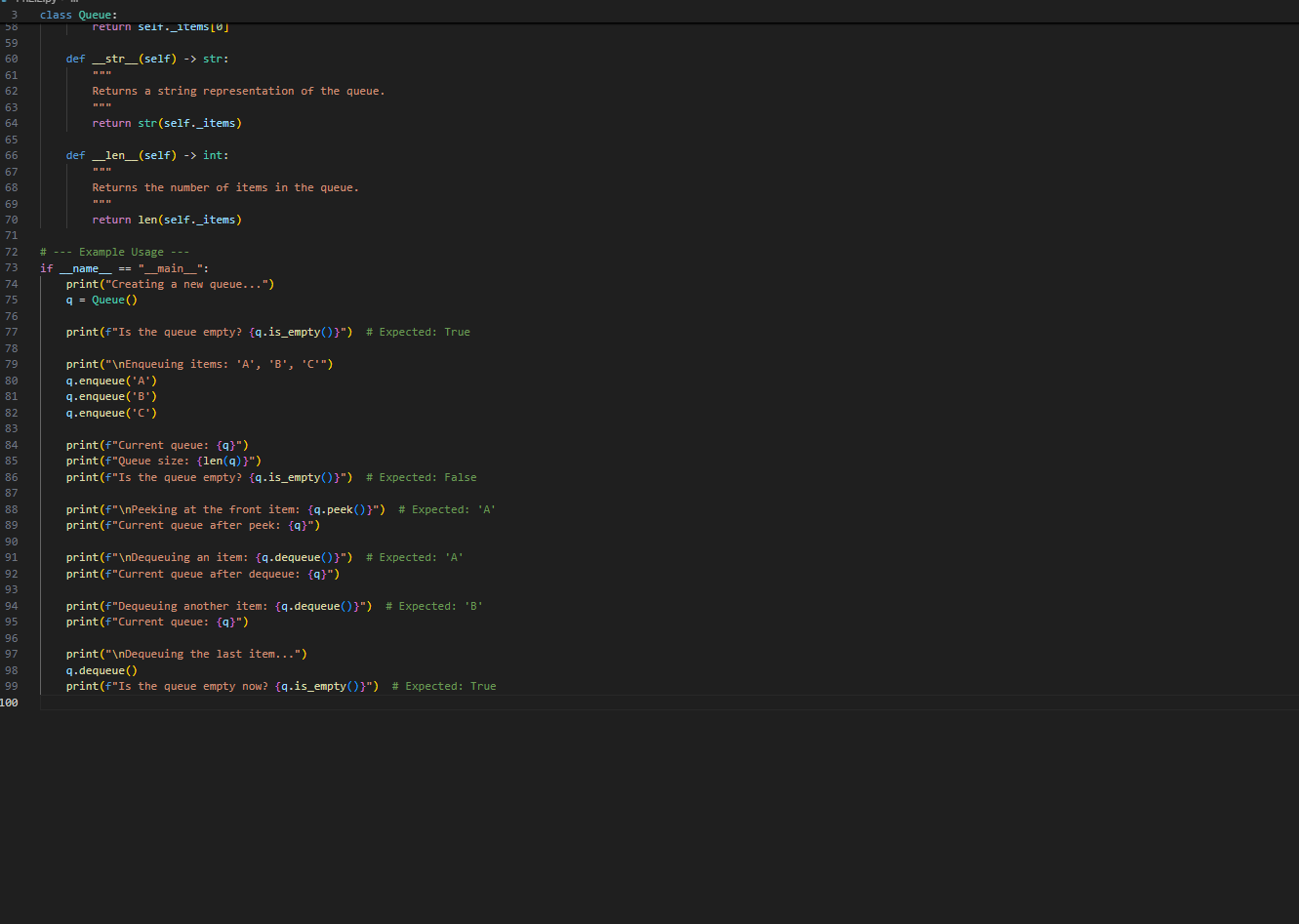
Use AI to implement a Queue using Python lists.  
Sample Input Code:  
class Queue:  
pass

**PROMPT:**

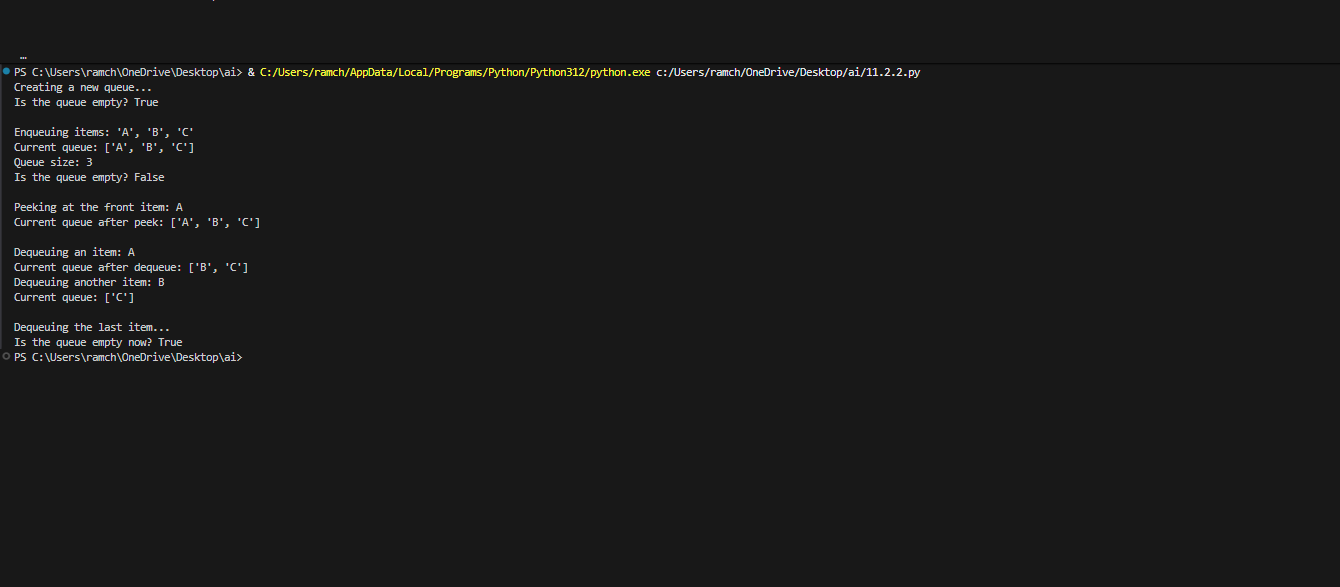
Generate a python code to implement the queue in data structures

**CODE:**

****

****

**OUTPUT:**

****

**OBSERVATION:**

The code generated by AI is more accurate about the queue in data structures and it also passed all the test cases.

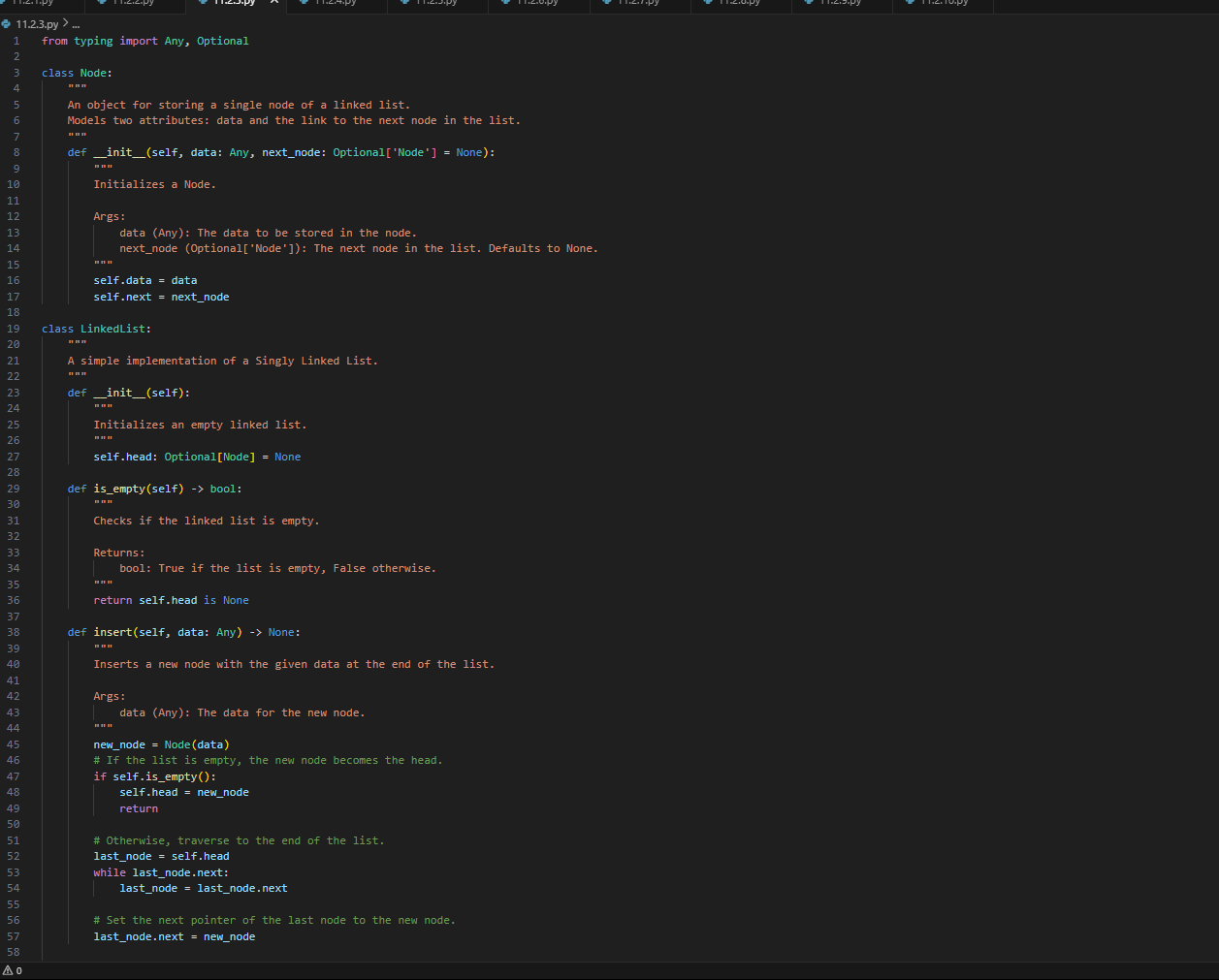
**TASK-03:**

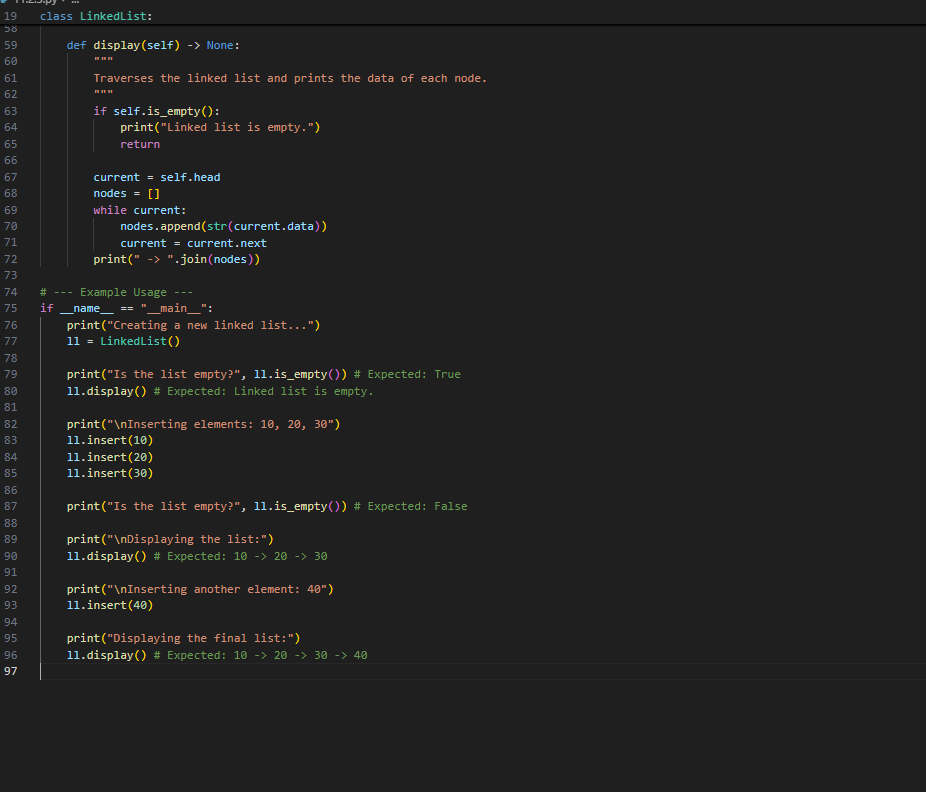
Use AI to generate a Singly Linked List with insert and display methods.  
Sample Input Code:  
class Node:  
pass  
class LinkedList:  
pass

**PROMPT:**

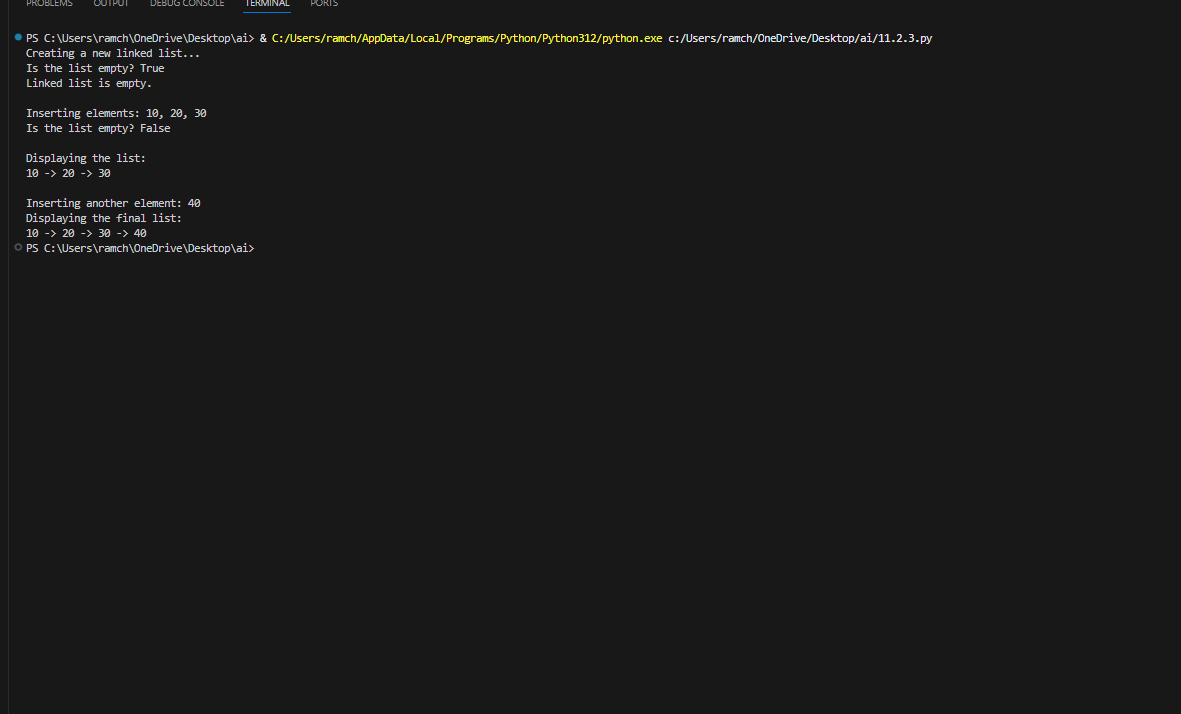
Generate a python code of singly linked list with insert and display methods.

**CODE:**

****

****

**OUTPUT:**

****

**OBSERVATION:**

The code generated by the AI is used to perform the operations in the data structures like singly linked list performing insert and delete operations accurately.

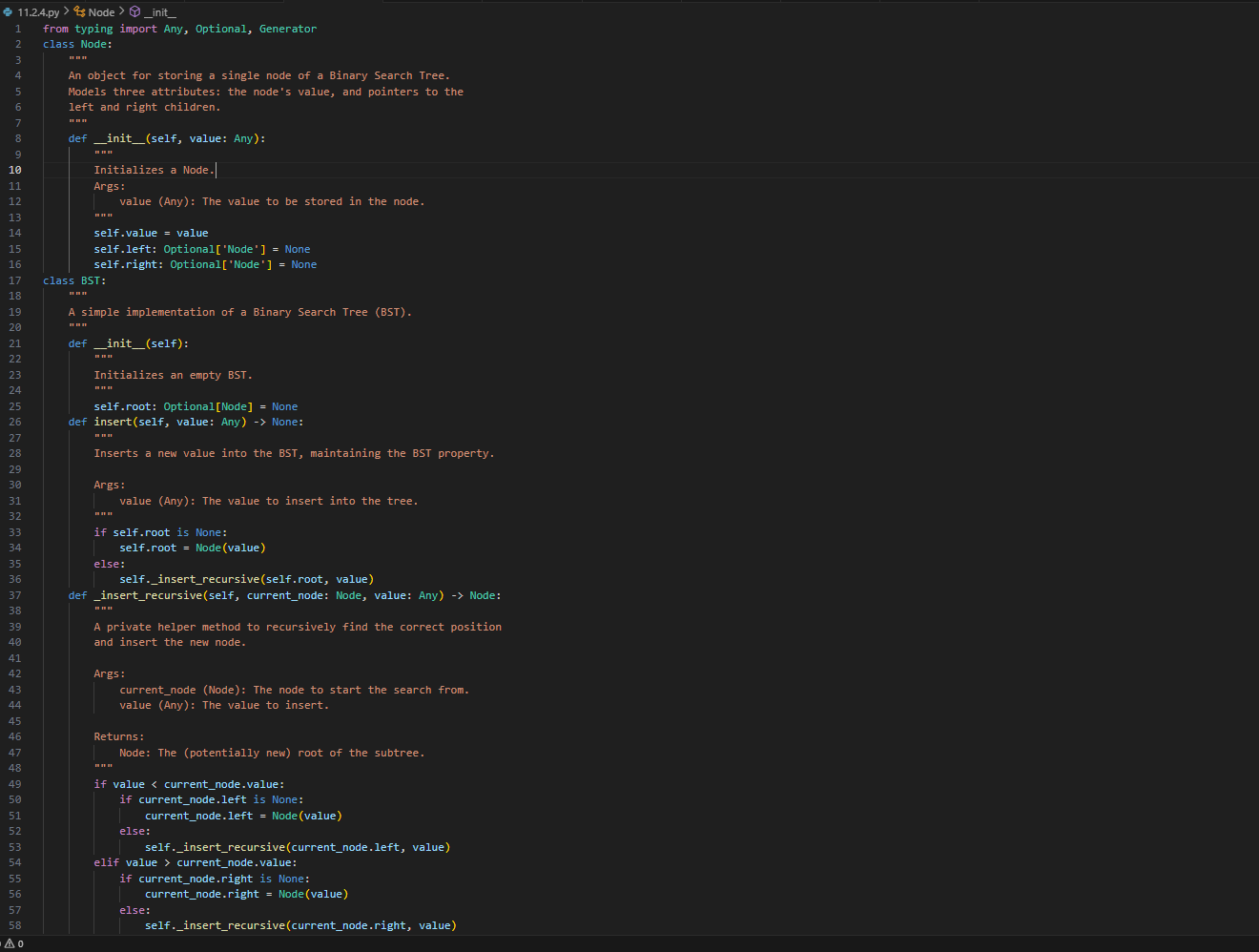
**TASK-04:**

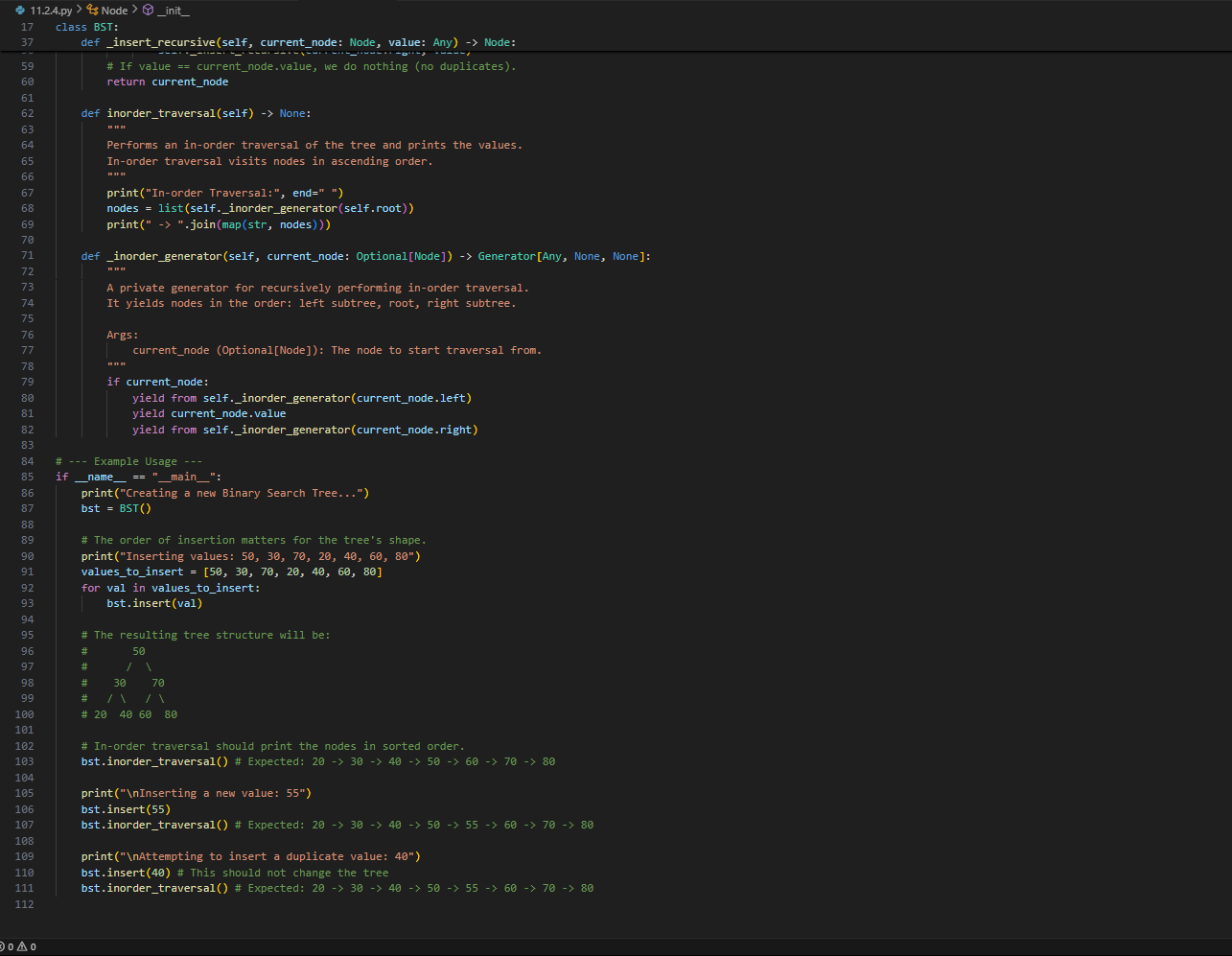
Use AI to create a BST with insert and in-order traversal methods.  
Sample Input Code:  
class BST:  
pass

**PROMPT:**

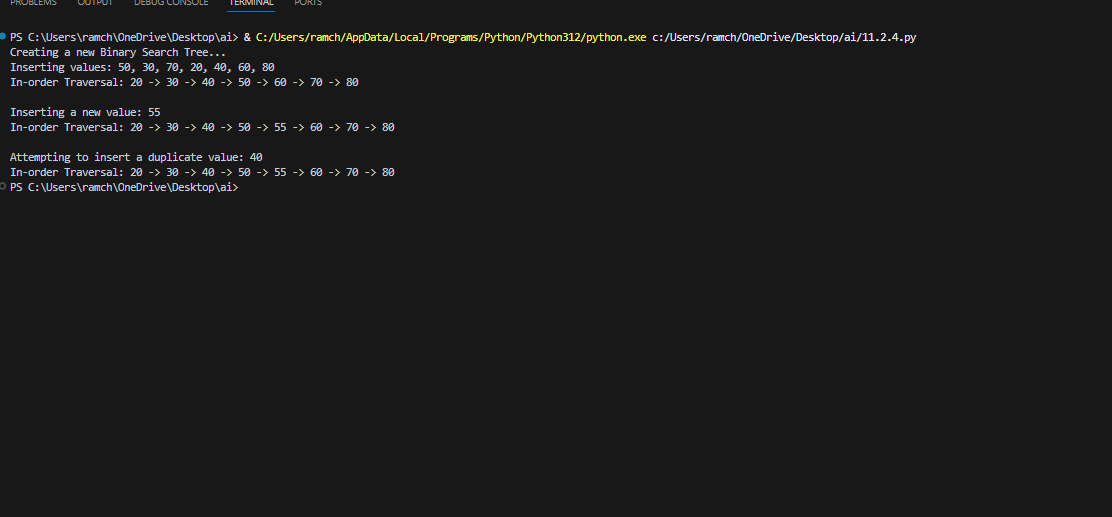
Generate a python code which creates a BST with insert and in-order methods.

**CODE:**

****

****

**OUTPUT:**

****

**OBSERVATION:**

The BST in data structures with insertion and in-order traversal methods are generated in python which are more helpful and in an efficient way to understand.

**TASK-05:**

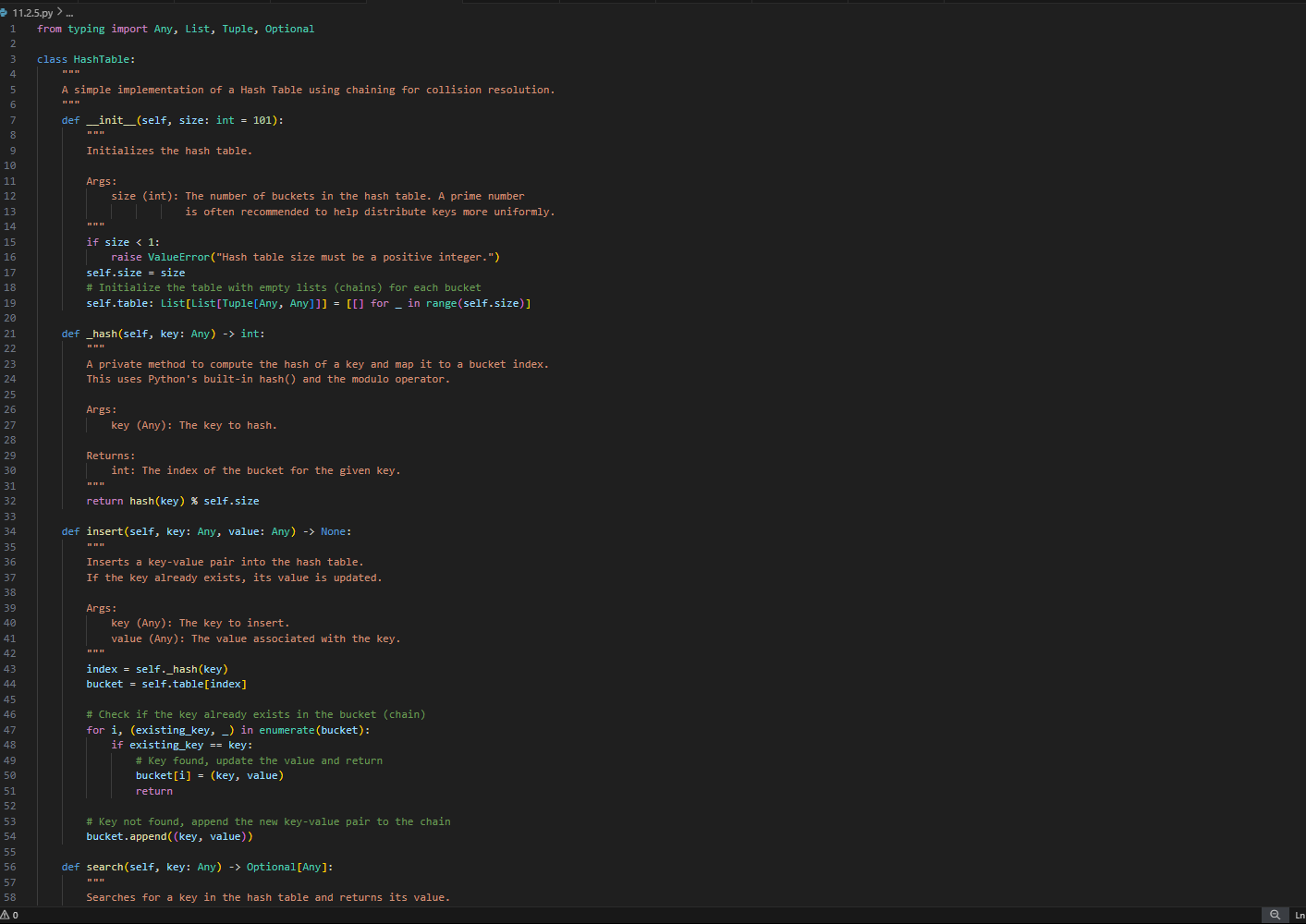
Use AI to implement a hash table with basic insert, search, and delete

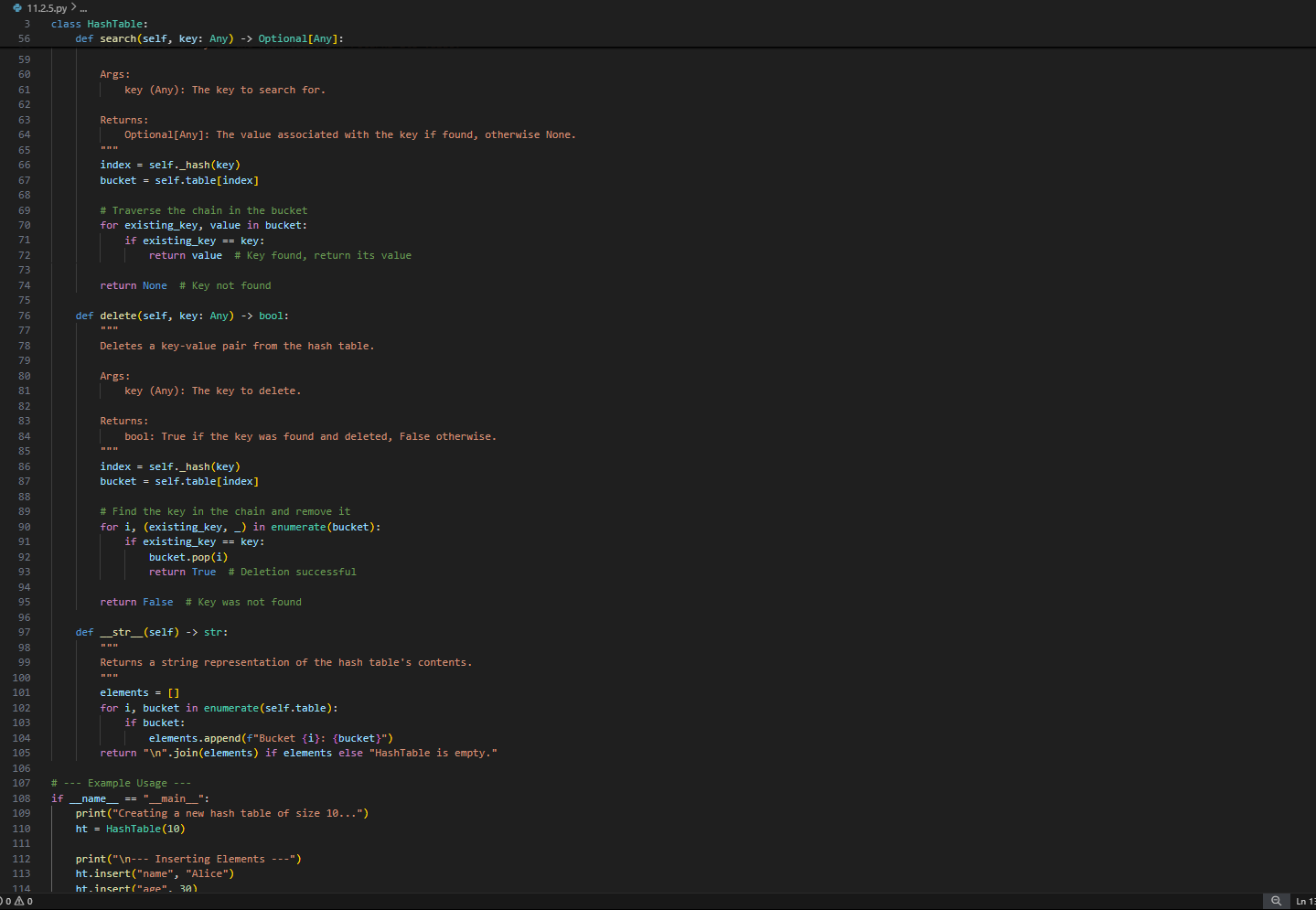
methods.  
Sample Input Code:  
class HashTable:  
pass

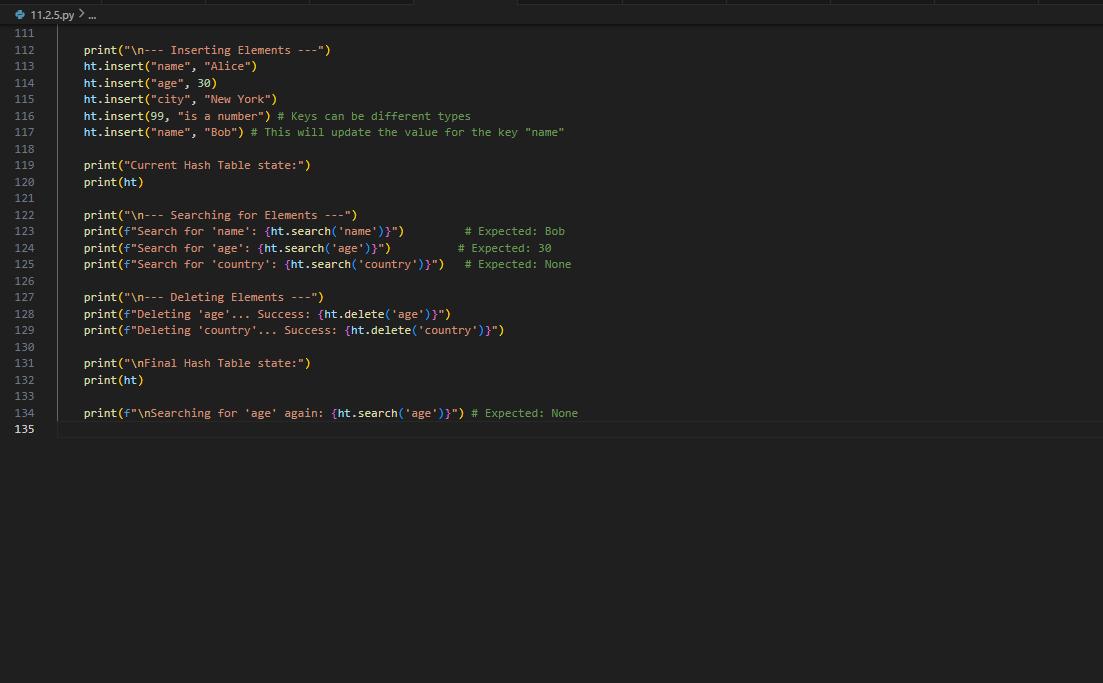
**PROMPT:**

Generate a python code which helps to implement the hashtable with basic insert and delete methods.

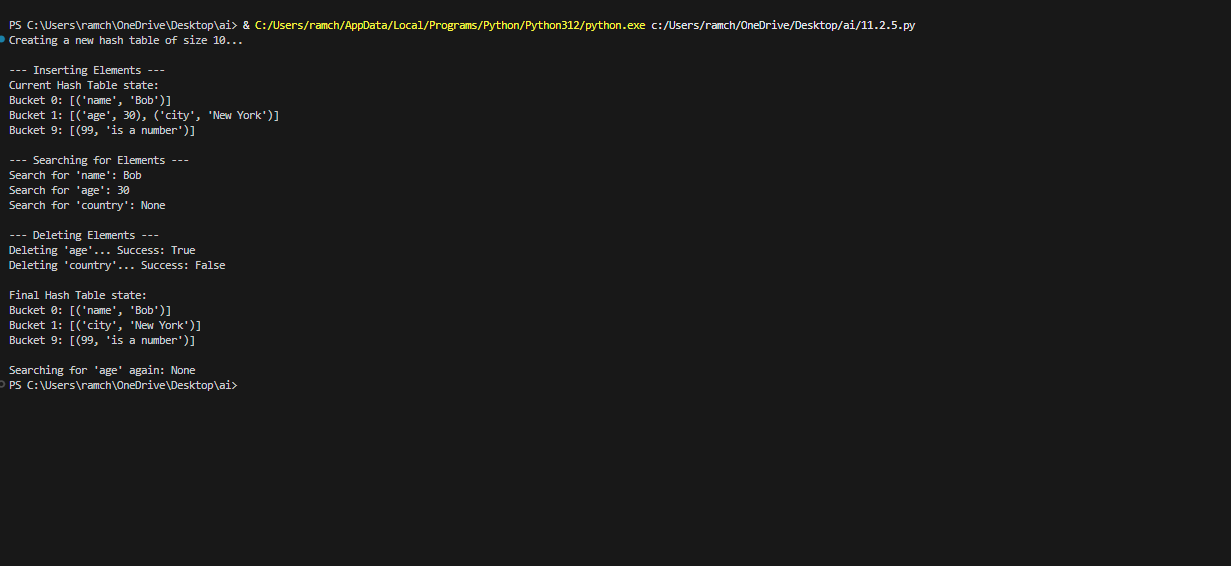
**CODE:**

****

****

****

**OUTPUT:**

****

**OBSERVATION:**

AI generated the code of data structures topic hashtable with basic insert, search and delete which aids us to understand the hashtable using the programming language like python.

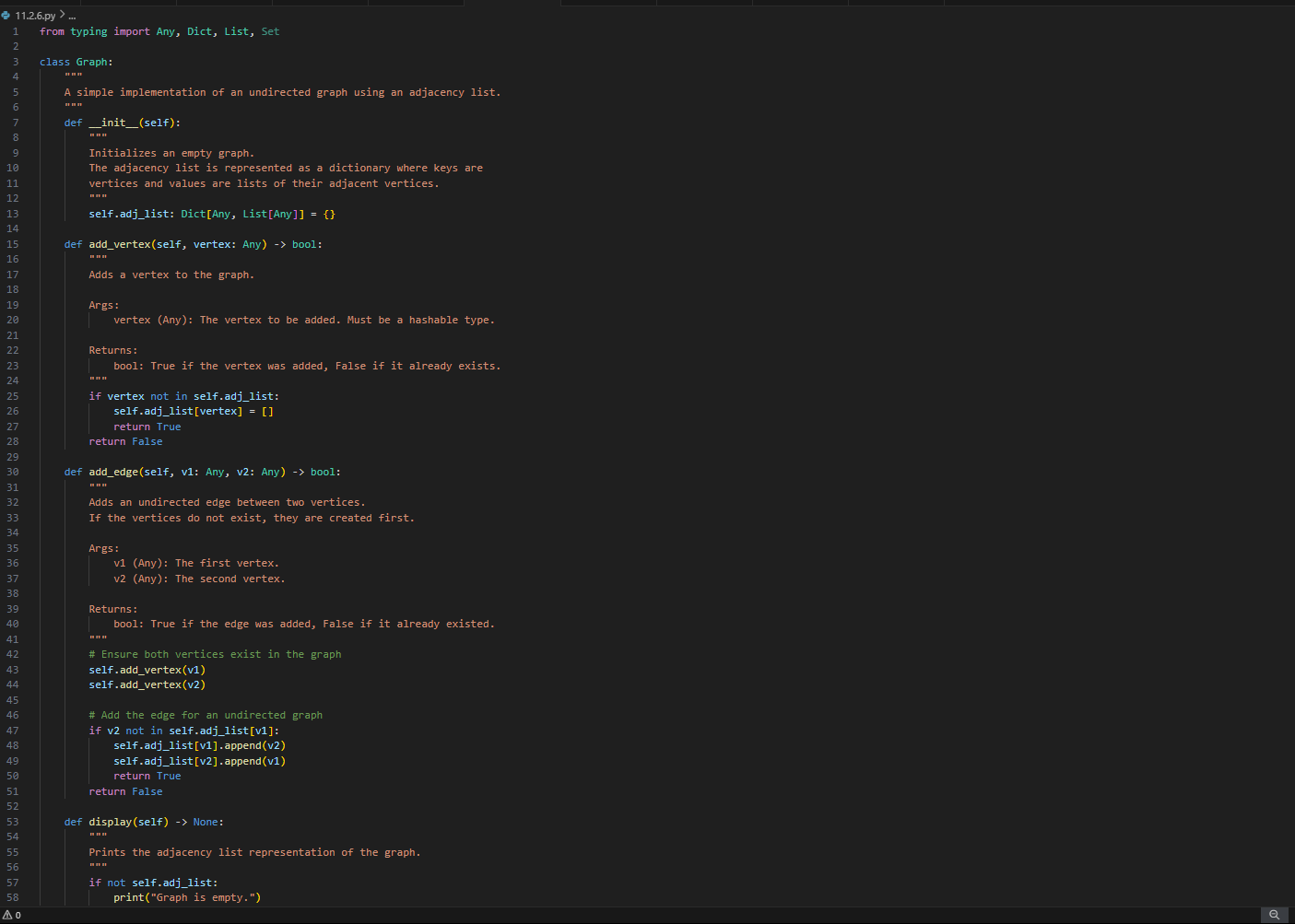
**TASK-06:**

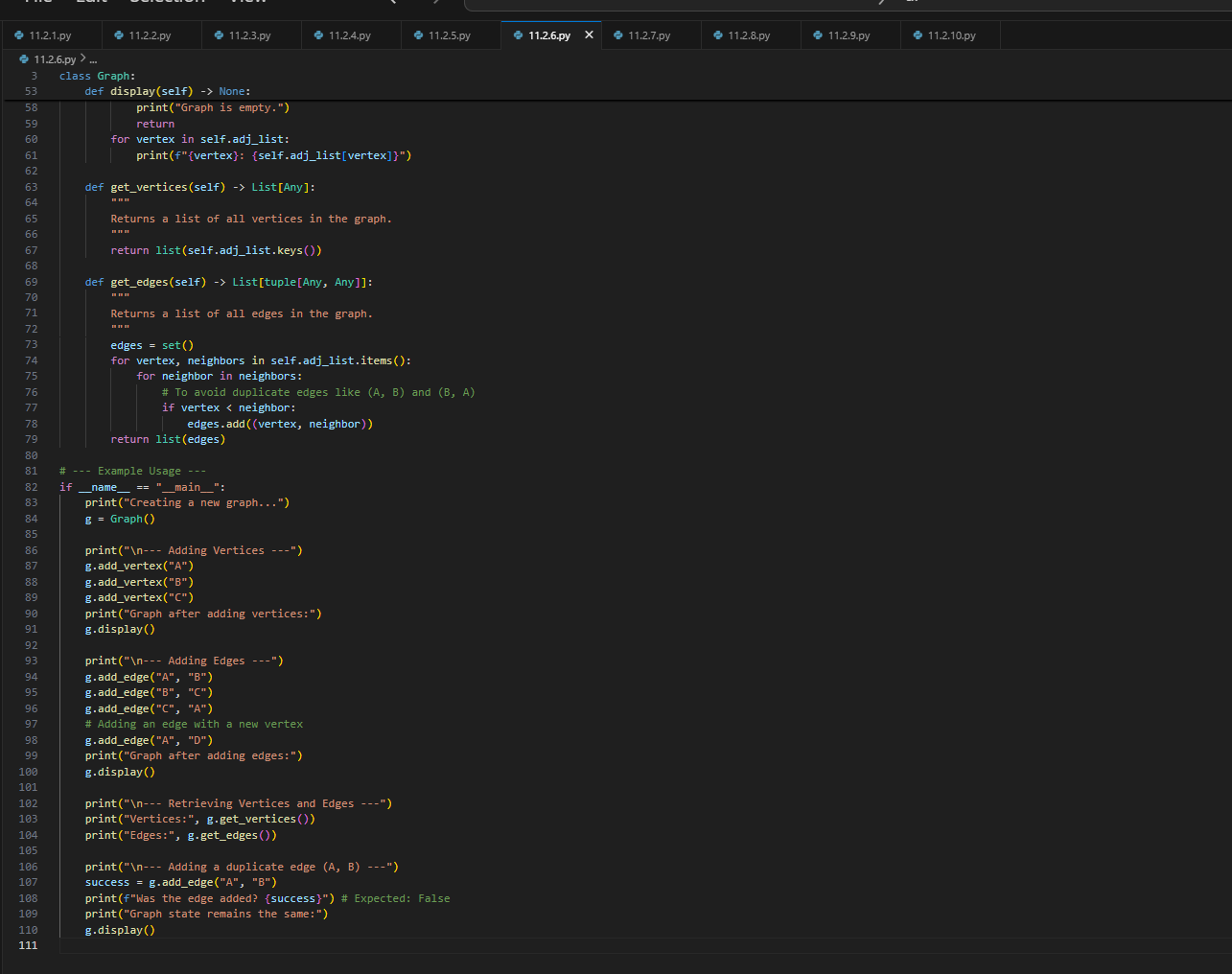
Use AI to implement a graph using an adjacency list.  
Sample Input Code:  
class Graph:  
pass

**PROMPT:**

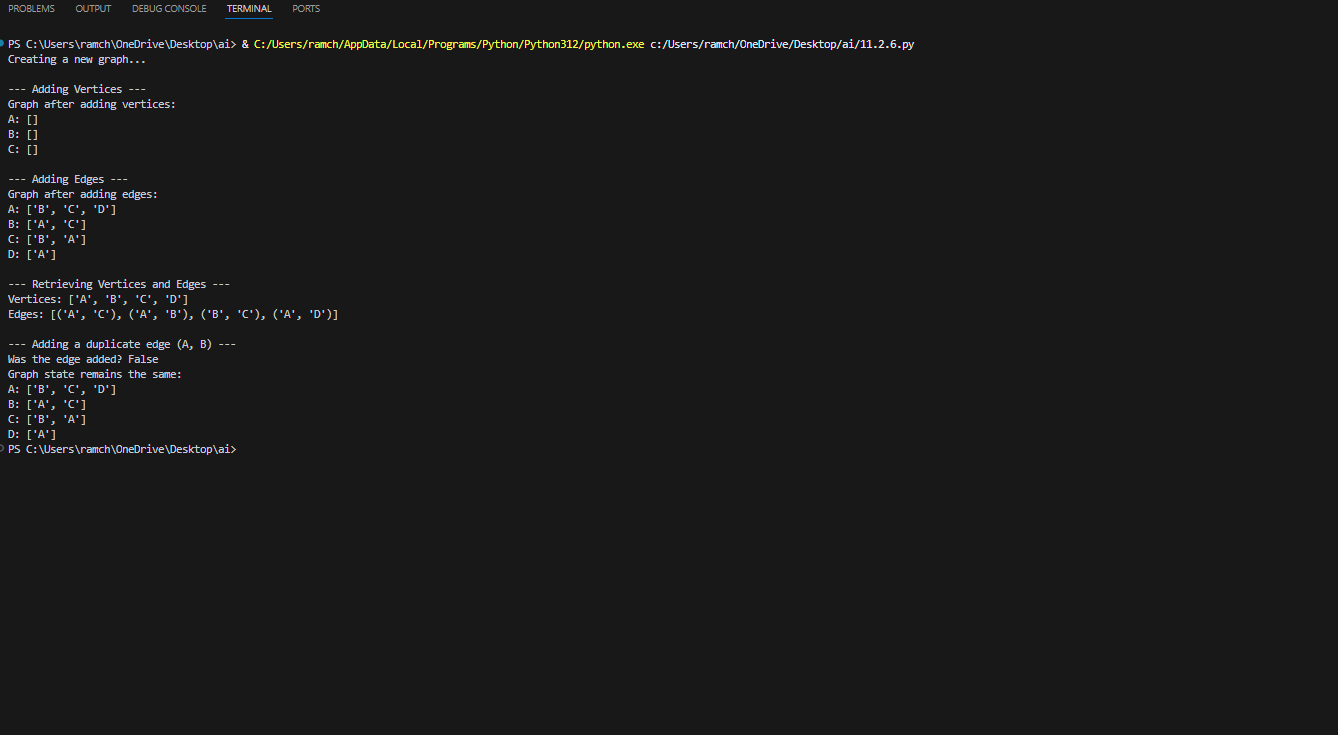
Generate a python code which implements the graph using an adjacency list in data structures.

**CODE:**

****

****

**OUTPUT:**

****

**OBSERVATION:**

Implementation of graph with the adjacency list is generated by the AI which makes us know about the graph in an efficient way and easy to understand.

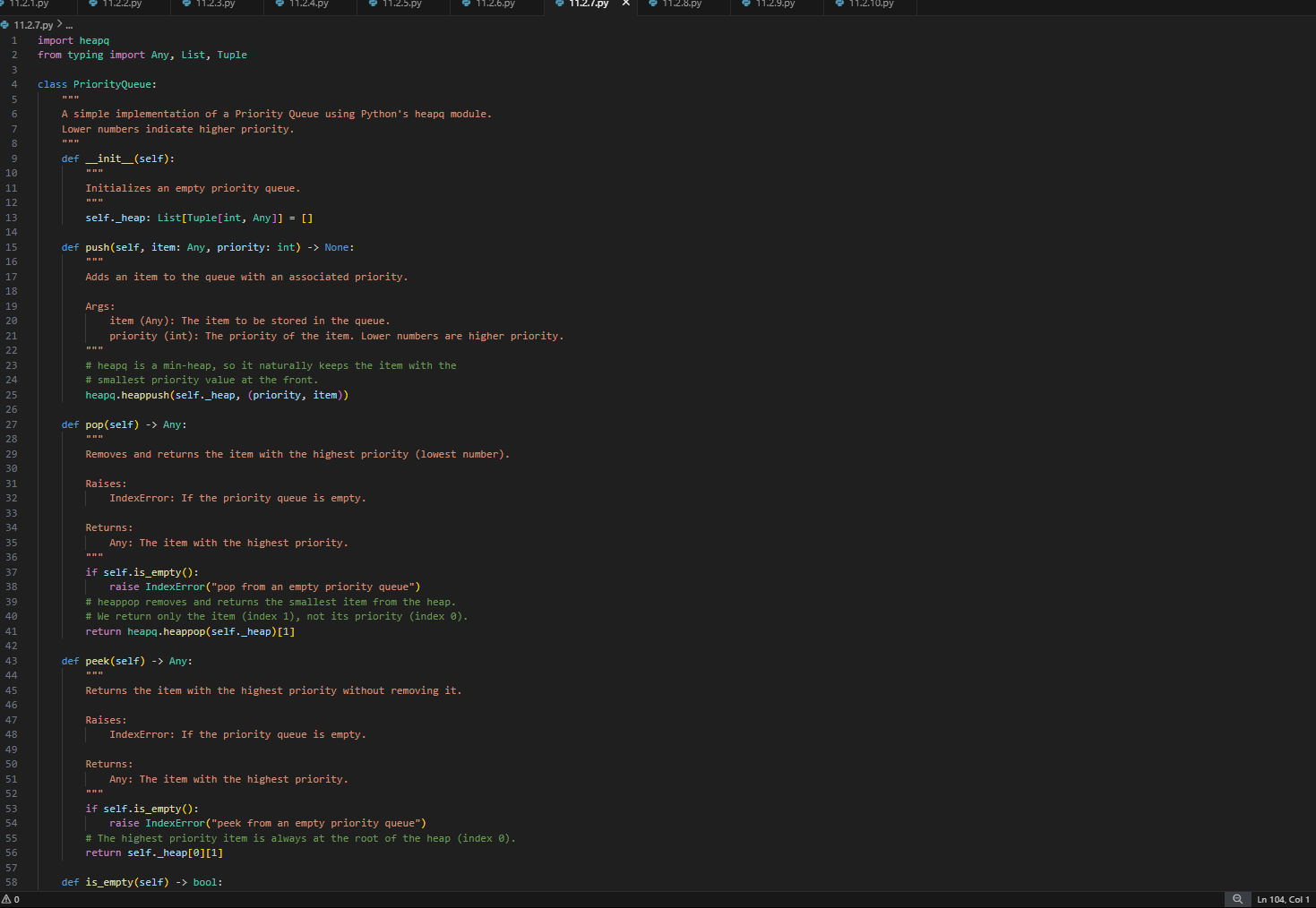
**TASK-07:**

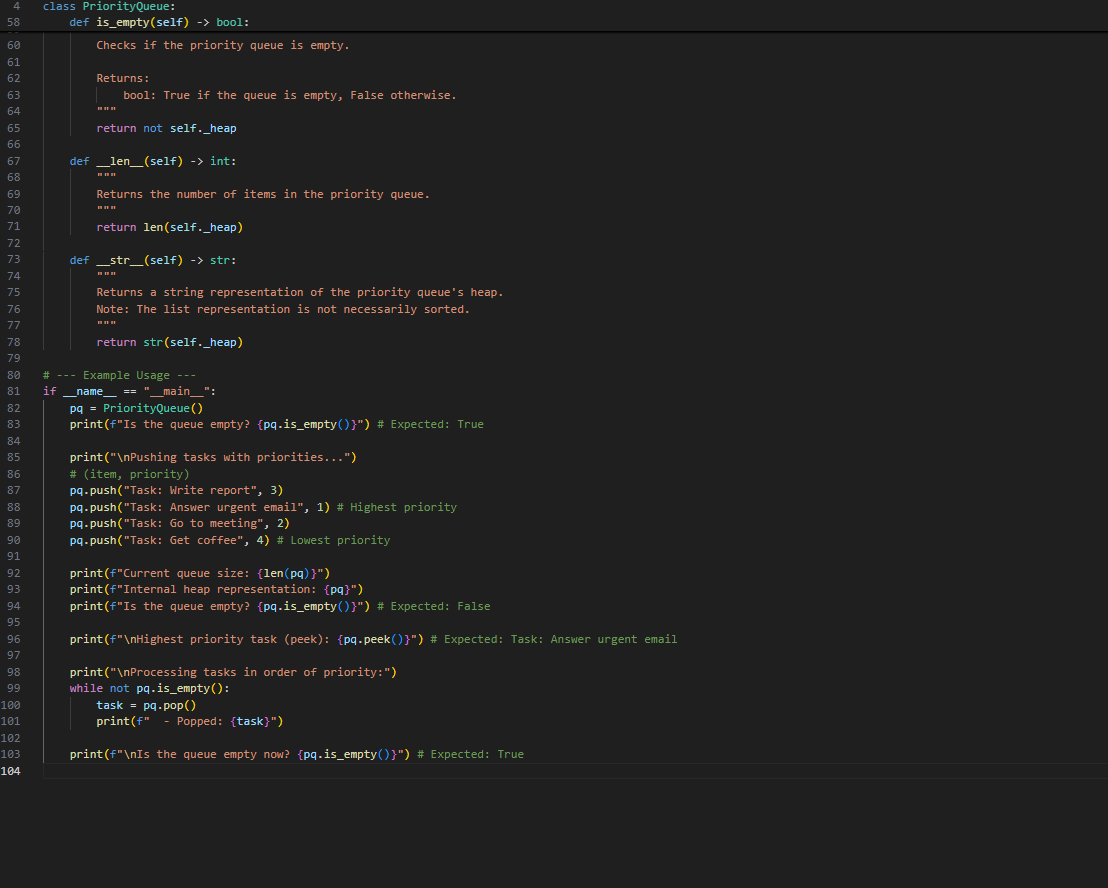
Use AI to implement a priority queue using Python’s heapq module.  
Sample Input Code:  
class PriorityQueue:  
pass

**PROMPT:**

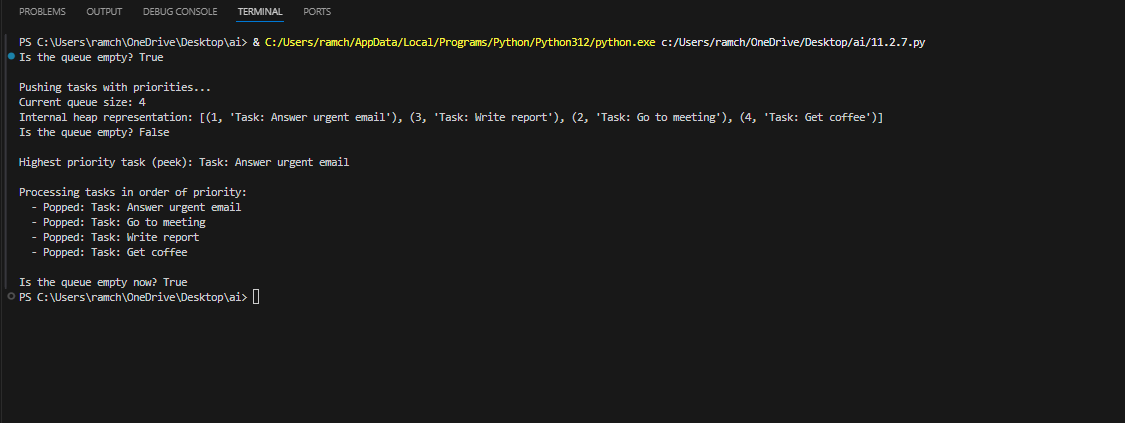
Write a python code which implement the priority queue using the heapq module.

**CODE:**

****

****

**OUTPUT:**

****

**OBSERVATION:**

The priority queue is generated using the heaqp module of python which implements them accurately.

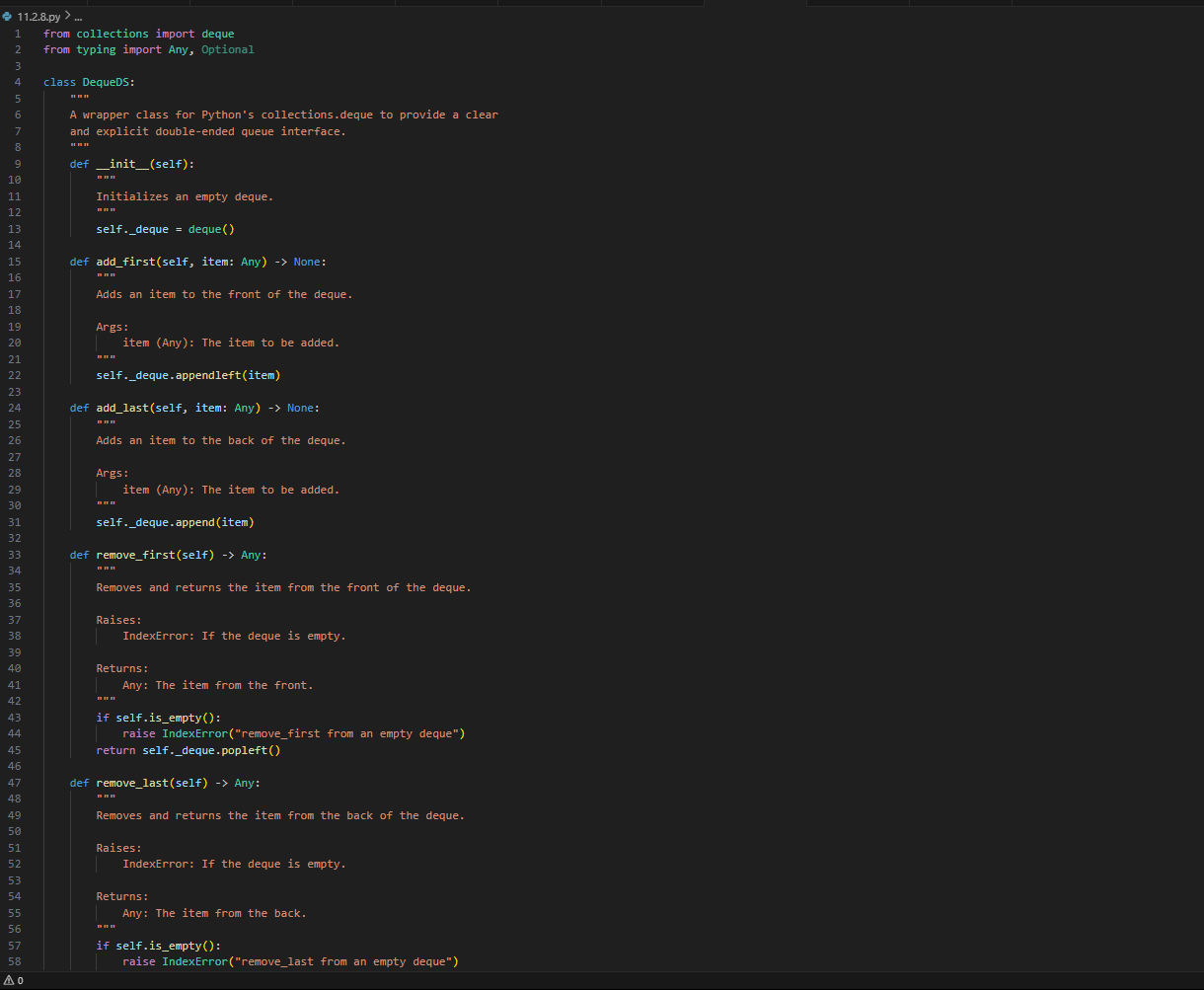
**TASK-08:**

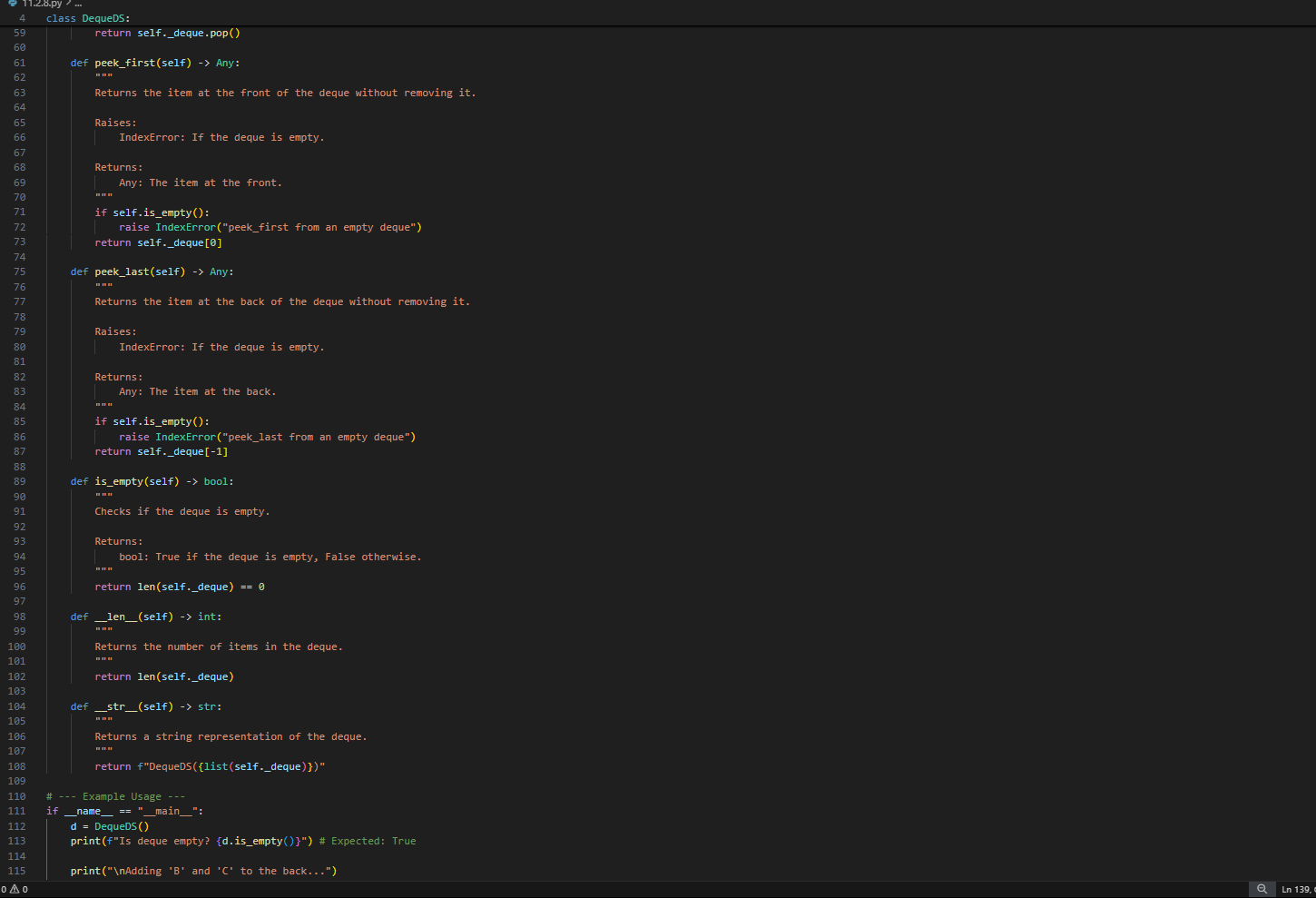
Use AI to implement a double-ended queue using collections.deque.  
Sample Input Code:  
class DequeDS:  
pass

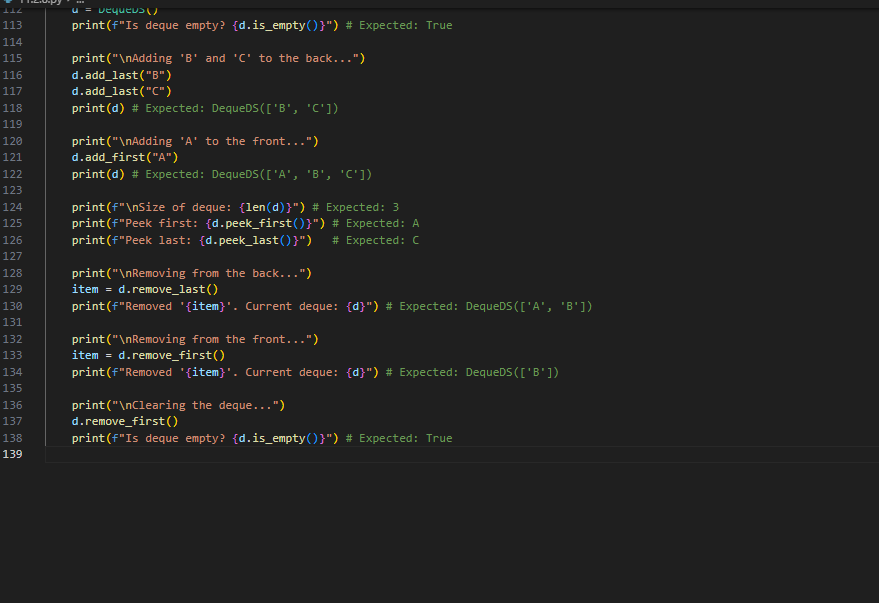
**PROMPT:**

Write a python code to implement the double-ended queue using collections .deque

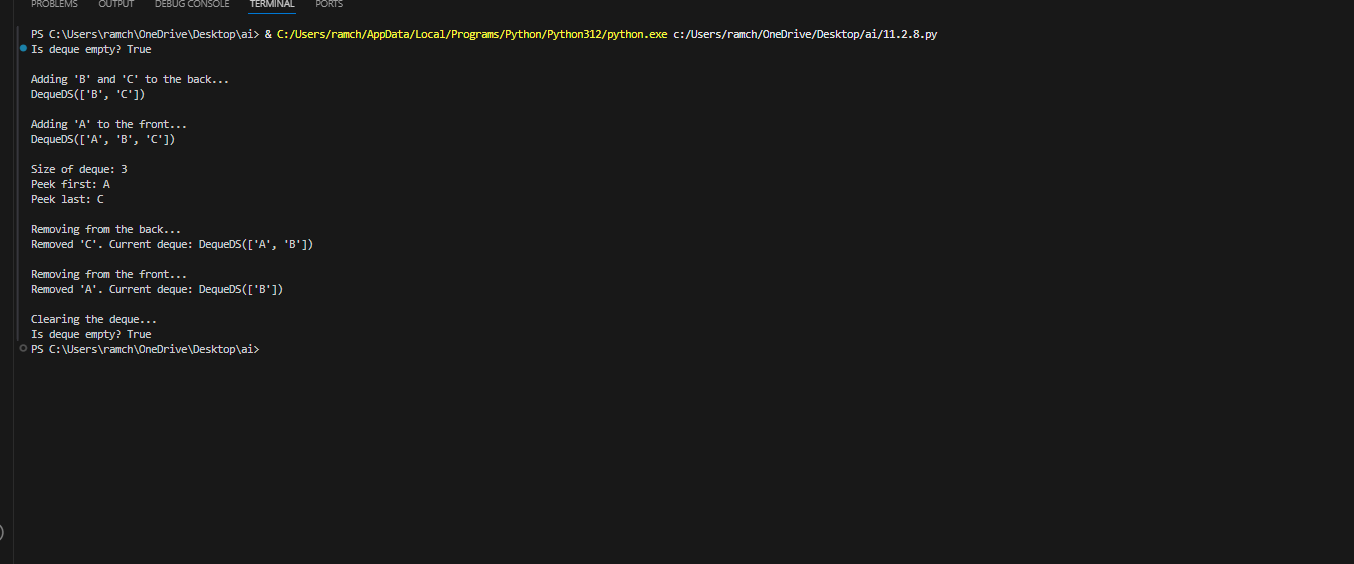
**CODE:**

****

****

****

**OUTPUT:**

****

**OBSERVATION:**

The code generated by the AI of data structures topic of double-ended queue with the help of python to easily understand the topic accurately.

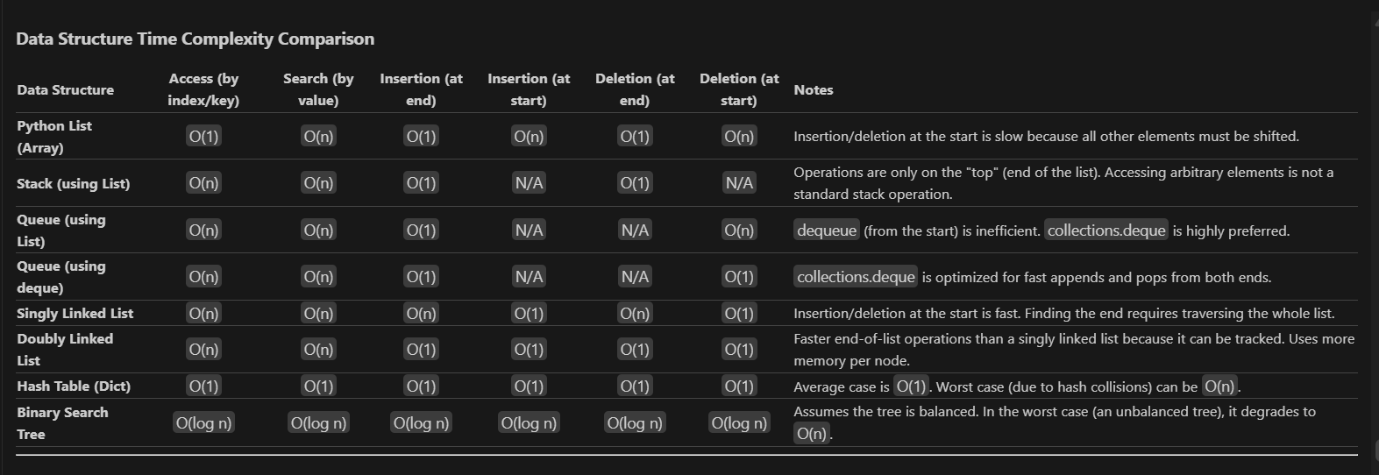
**TASK-09:**

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 a comparative table of different data structures including time complexities .

**TABLE:**

****

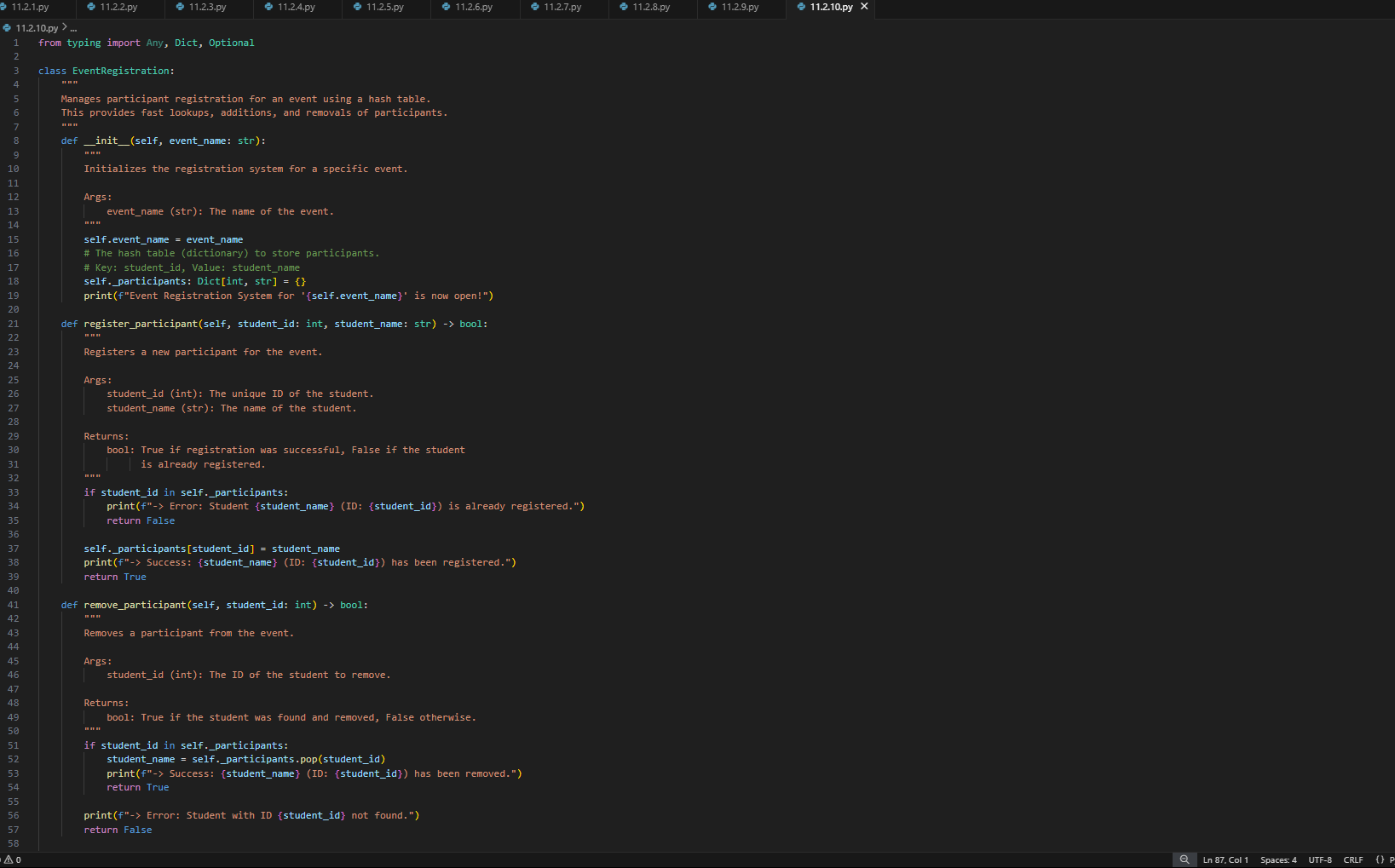
**TASK-10:**

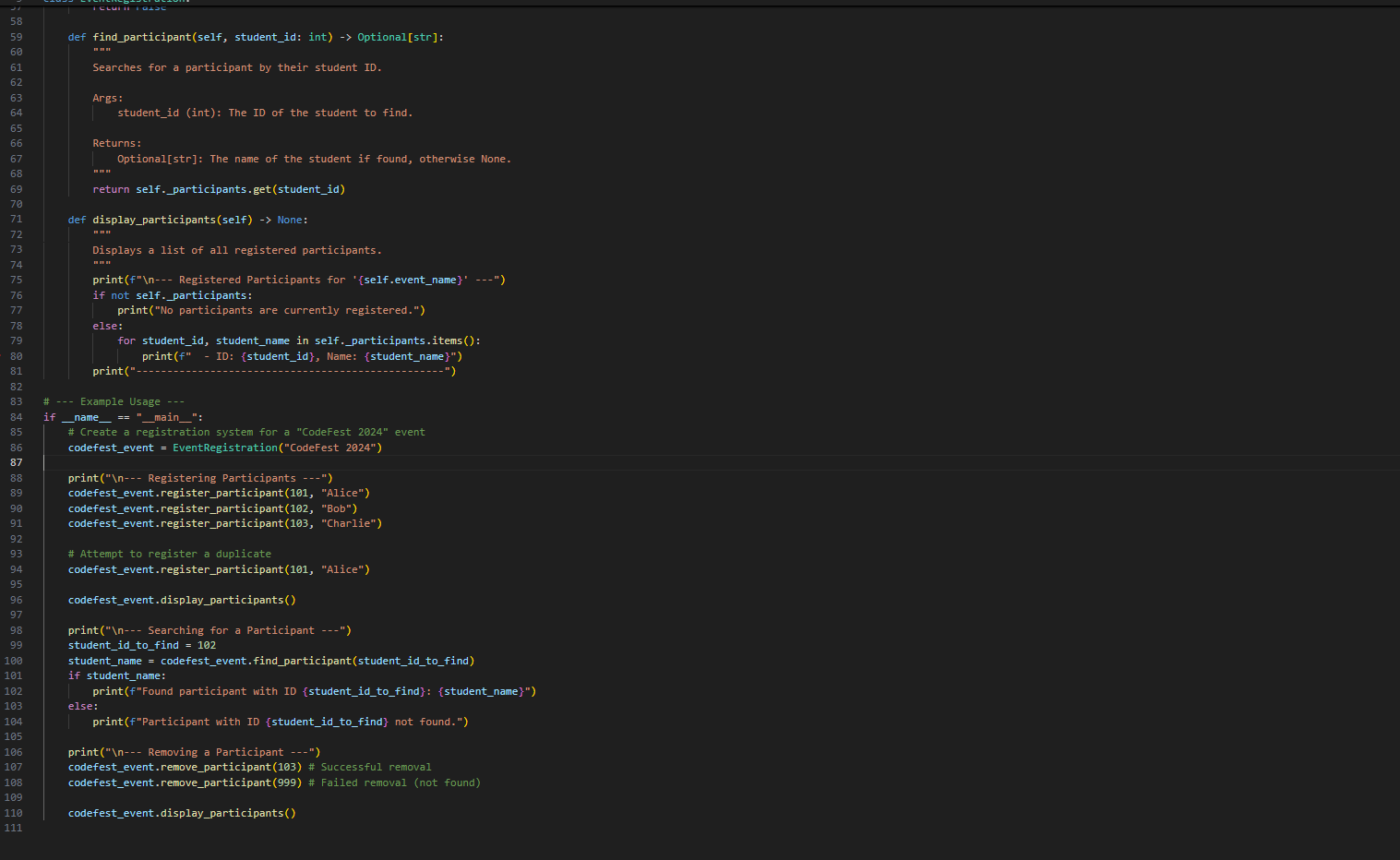
Your college wants to develop a Campus Resource Management System that  
handles:  
1. Student Attendance Tracking – Daily log of students entering/exiting  
the campus.  
2. Event Registration System – Manage participants in events with quick  
search and removal.  
3. Library Book Borrowing – Keep 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:  
o Stack  
o Queue  
o Priority Queue  
o Linked List  
o Binary Search Tree (BST)  
o Graph  
o Hash Table  
o Deque  
• Justify your choice in 2–3 sentences per feature.  
• Implement one selected feature as a working Python program with AI-  
assisted code generation.

**PROMPT:**

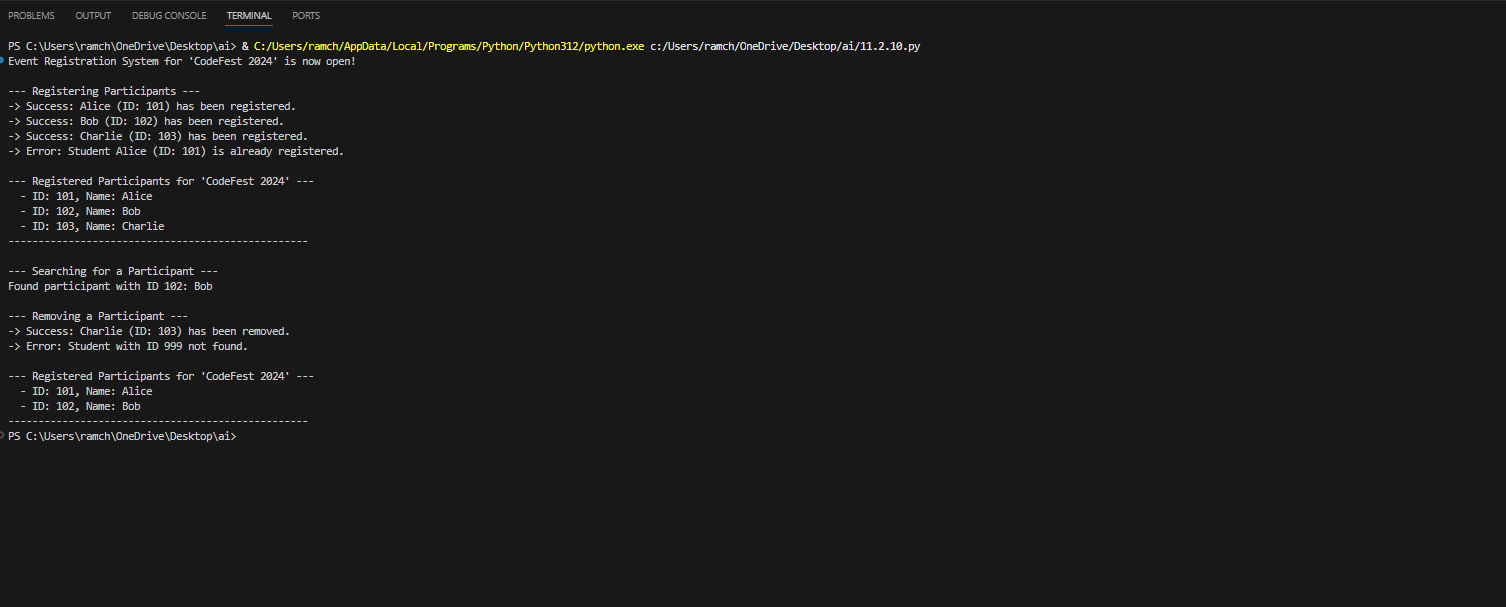
Generate the python code according to the given above details.

**CODE:**

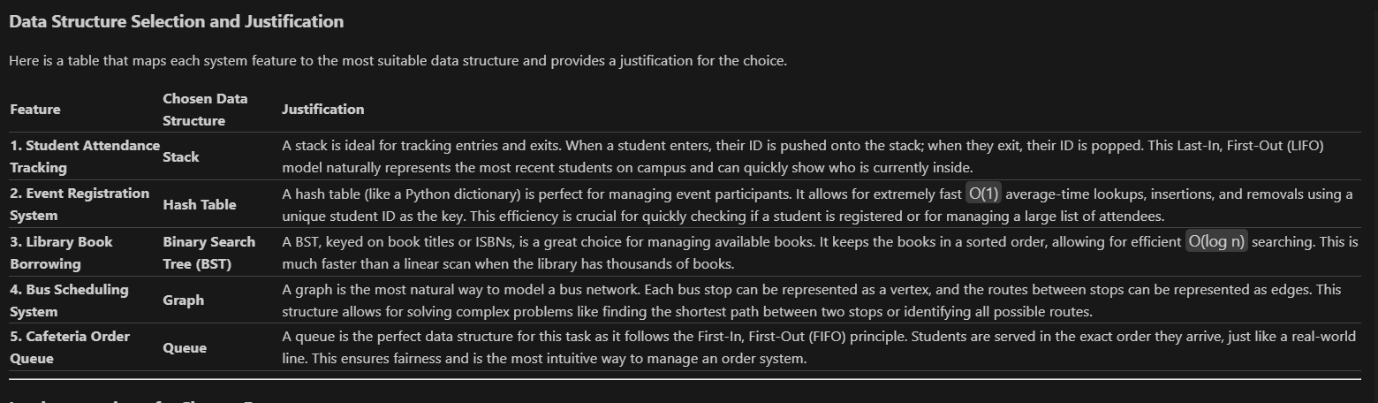
****

****

**OUTPUT:**

****

**TABLE:**

****

**OBSERVATION:**

The AI generated the code in an efficient way according to the details given in which it include all the data structures concepts to make all easily understand . As the task contains much more information it should be handled in an efficient way.