

**FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY**

page1image9370432

**PROG211 – Objected Oriented Programming Methods 1**

page1image9370432

Title : Individual Assignment

Issue Date : Week 2

Due Date : Week 4

Lecturer/Examiner : Amadus Coker

Name of Student/s : Joseph Lahai

Student ID No. : 905004212

Class : BICT1101F

Year/Semester : 2/3

Academic Honesty Policy Statement

page1image9370432

I/We, hereby attest those contents of this attachment are my own work. Referenced works, articles, art, programs, papers or parts thereof are acknowledged at the end of this paper. This includes data excerpted from CD-ROMs, the Internet, other private networks, and other people’s disk of the computer system.

Student’s Signature: Date:

page1image9370432

LECTURER’S COMMMENTS/GRADE:

for office use only upon receive

Remark

DATE:  
 TIME:

RECEIVER’S NAME:

**DESIGN RATIONALE FOR LIBRARY MANAGEMENT**

**SYSTEM**

This rationale explains the selection of a dictionary for 'books', a list for 'members', and a tuple for 'genres', integrating the cultural context of Freetown while adhering to the brief's requirements.

I chose a dictionary for the 'books' data structure to efficiently manage the Freetown Library's collection, using ISBNs as unique keys for rapid access to book details such as title, author, genre, and total\_copies. Titles like "Tales of Freetown" by local author "Aminata Kamara" and "Sierra Leone History" by "Saidu Conteh" embody Sierra Leonean heritage, while the dictionary supports quick lookups, updates, and deletions through functions like add\_book() and search\_book().

For the 'members' data structure, I selected a list to accommodate the diverse community of Freetown Library patrons. Each member is stored as a dictionary within the list, with attributes like member\_id, name, email, and borrowed\_books e.g., "Fatima Bangura" with email "fatima@freetown.sl". The list's flexibility allows easy addition or removal of members via add\_member(), and it supports borrowing and returning operations through borrow\_book() and return\_book(), enforcing a three-book limit..

I opted for a tuple for the 'genres' data structure to establish a fixed, immutable set of valid genres ('Fiction', 'Non-Fiction', 'Sci-Fi'), as mandated by the brief. This immutability prevents unintended changes during execution, ensuring consistency when validating genres in add\_book() and update\_book()..

These data structure choices create a robust Freetown Library system. The dictionary ensures efficient book management, the list supports a dynamic member base, and the tuple guarantees genre consistency.

**UML DIAGRAM SKETCH (Hand Drawn)**

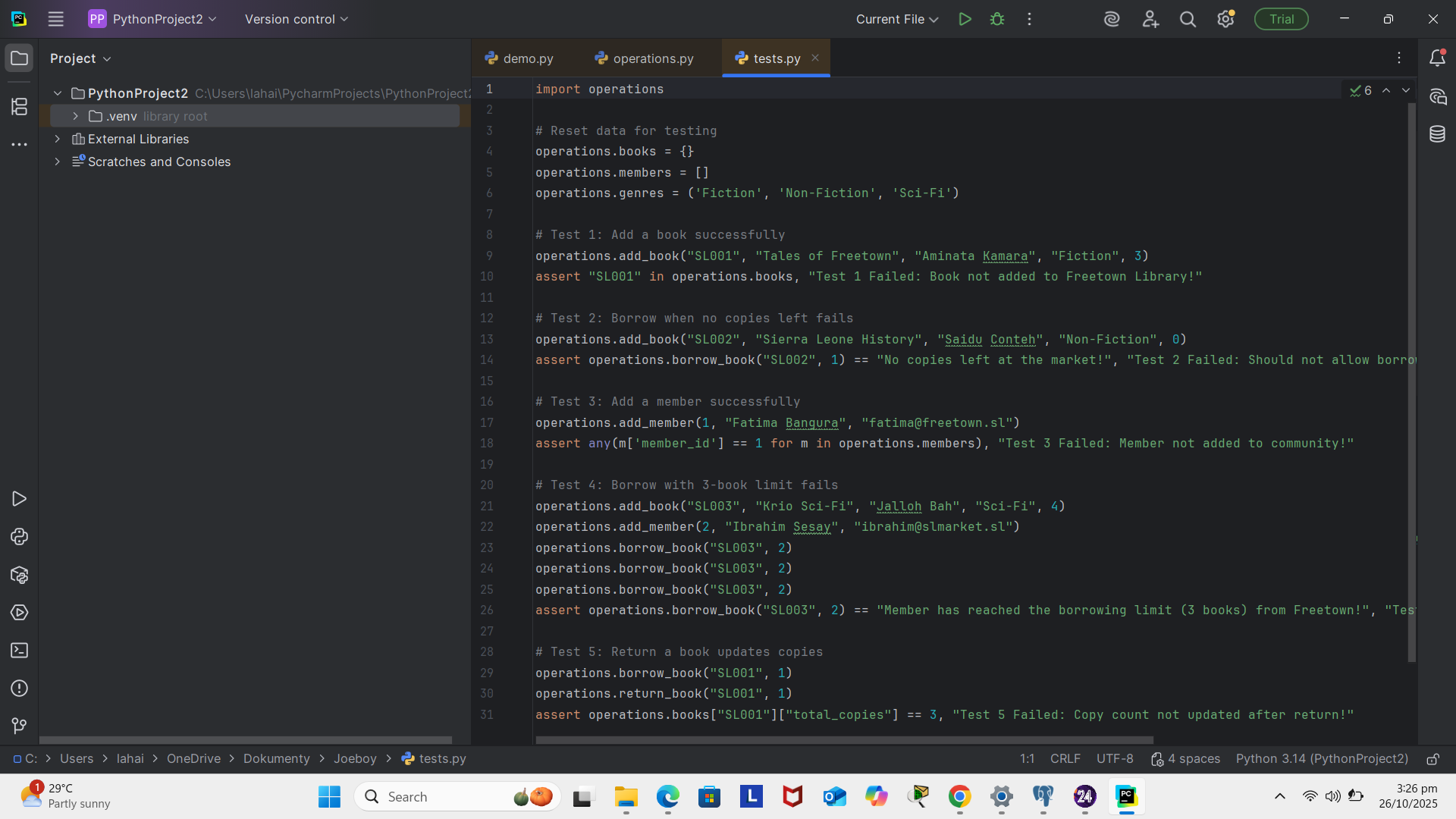
**OPERATIONS.PY CODE**

books = {}  
members = []  
genres = ('Fiction', 'Non-Fiction', 'Sci-Fi')  
  
def add\_book(ISBN, title, author, genre, total\_copies):  
 if ISBN in books:  
 return "Book already exists in Freetown Library!"  
 if genre not in genres:  
 return "Invalid genre, stick to Sierra Leone's favorites!"  
 if total\_copies < 0:  
 return "Total copies cannot be negative!"  
 books[ISBN] = {"title": title, "author": author, "genre": genre, "total\_copies": total\_copies}  
 return "Book added to Freetown Library successfully!"  
  
def add\_member(member\_id, name, email):  
 for member in members:  
 if member['member\_id'] == member\_id:  
 return "Member ID already exists in our community!"  
 members.append({'member\_id': member\_id, 'name': name, 'email': email, 'borrowed\_books': []})  
 return "Member added to Sierra Leone Library community!"  
  
def search\_book(keyword):  
 results = []  
 for ISBN, details in books.items():  
 if keyword.lower() in details['title'].lower() or keyword.lower() in details['author'].lower():  
 results.append({"ISBN": ISBN, "title": details['title'], "author": details['author'], "genre": details['genre'], "total\_copies": details['total\_copies']})  
 return results if results else "No books found in Freetown markets!"  
  
def update\_book(ISBN, \*\*details):  
 if ISBN not in books:  
 return "Book not found in Freetown Library!"  
 if 'genre' in details and details['genre'] not in genres:  
 return "Invalid genre, check Sierra Leone's list!"  
 if 'total\_copies' in details and details['total\_copies'] < 0:  
 return "Total copies cannot be negative!"  
 books[ISBN].update(details)  
 return "Book updated at Freetown Library!"  
  
def delete\_book(ISBN):  
 if ISBN not in books:  
 return "Book not found in Freetown Library!"  
 for member in members:  
 if ISBN in member['borrowed\_books']:  
 return "Cannot delete: Book is borrowed by a Sierra Leonean!"  
 del books[ISBN]  
 return "Book deleted from Freetown Library!"  
  
def borrow\_book(ISBN, member\_id):  
 if ISBN not in books:  
 return "Book not found in Freetown Library!"  
 if books[ISBN]['total\_copies'] <= 0:  
 return "No copies left at the market!"  
 member = next((m for m in members if m['member\_id'] == member\_id), None)  
 if not member:  
 return "Member not found in our community!"  
 if len(member['borrowed\_books']) >= 3:  
 return "Member has reached the borrowing limit (3 books) from Freetown!"  
 member['borrowed\_books'].append(ISBN)  
 books[ISBN]['total\_copies'] -= 1  
 return "Book borrowed from Freetown Library successfully!"  
  
def return\_book(ISBN, member\_id):  
 if ISBN not in books:  
 return "Book not found in Freetown Library!"  
 member = next((m for m in members if m['member\_id'] == member\_id), None)  
 if not member:  
 return "Member not found in our community!"  
 if ISBN not in member['borrowed\_books']:  
 return "Member did not borrow this book from Freetown!"  
 member['borrowed\_books'].remove(ISBN)  
 books[ISBN]['total\_copies'] += 1  
 return "Book returned to Freetown Library successfully!"

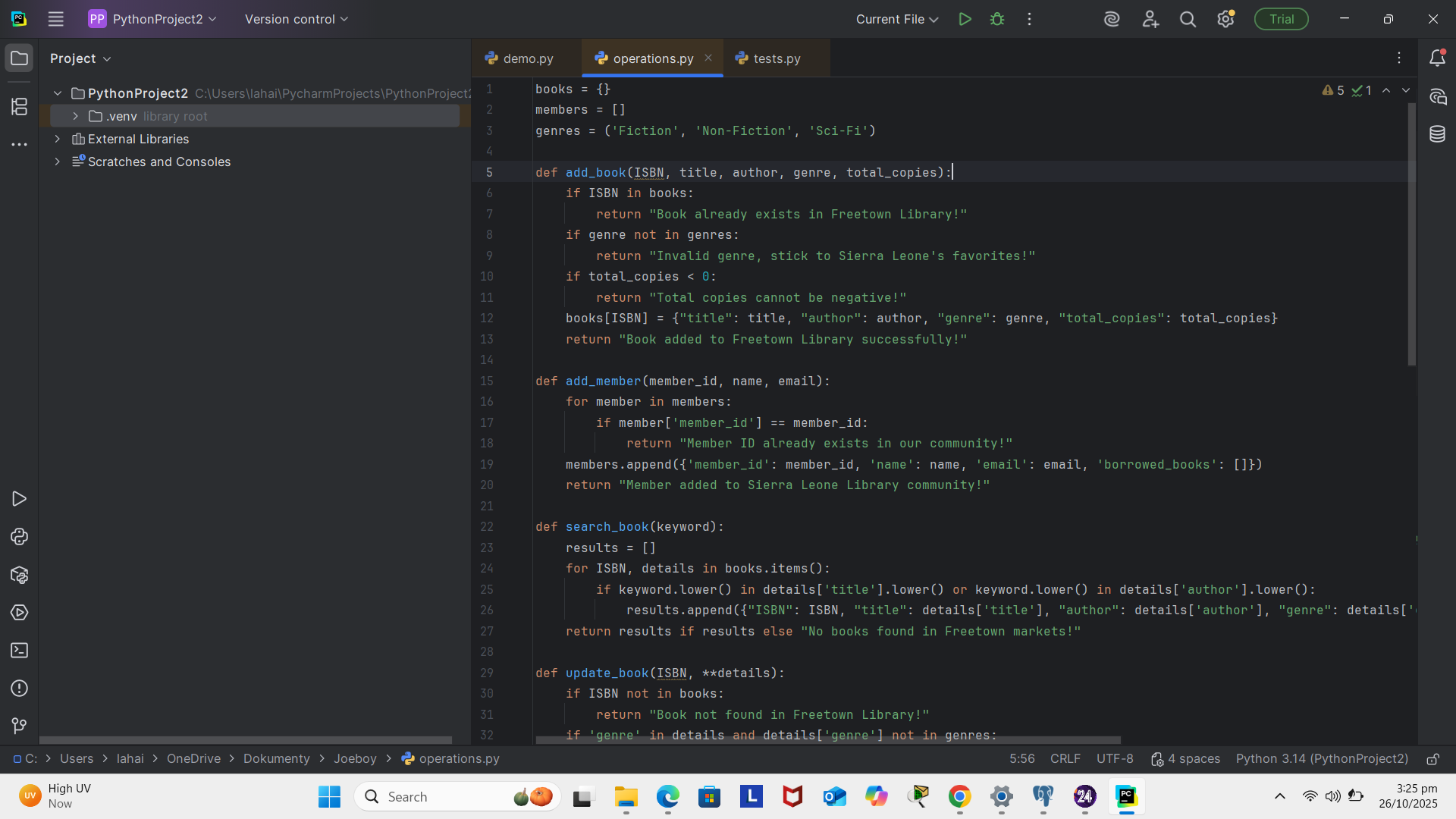
**DEMO.PY CODE**

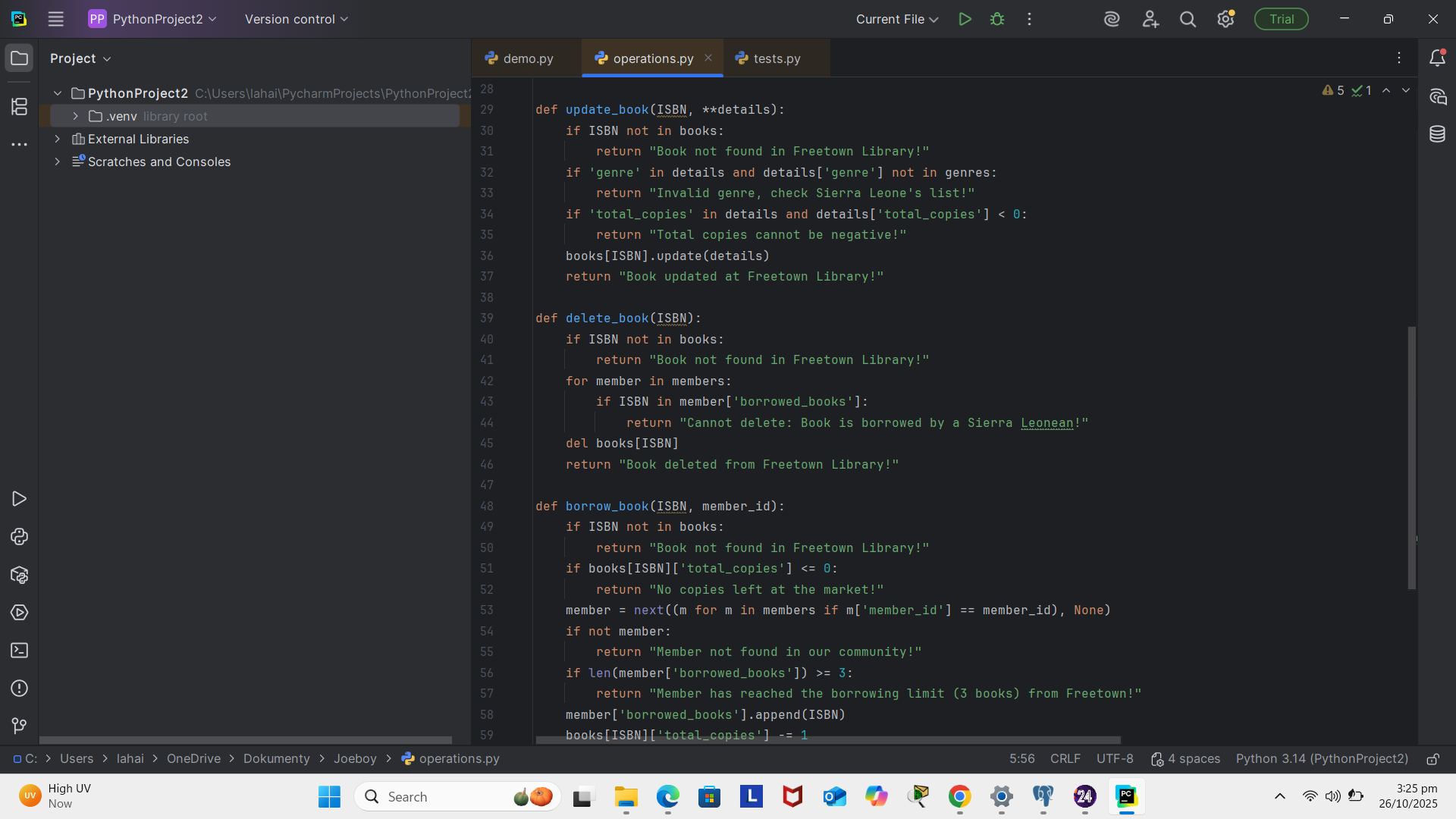
import operations  
  
# Add books  
print(operations.add\_book("SL001", "Tales of Freetown", "Aminata Kamara", "Fiction", 3))  
print(operations.add\_book("SL002", "Sierra Leone History", "Saidu Conteh", "Non-Fiction", 2))  
  
# Add members  
print(operations.add\_member(1, "Fatima Bangura", "fatima@freetown.sl"))  
print(operations.add\_member(2, "Ibrahim Sesay", "ibrahim@slmarket.sl"))  
  
# Borrow books  
print(operations.borrow\_book("SL001", 1)) # Should work  
print(operations.borrow\_book("SL002", 1)) # Should work  
print(operations.borrow\_book("SL002", 1)) # Error: No copies left  
  
# Search for a book  
print(operations.search\_book("Freetown"))  
  
# Update a book  
print(operations.update\_book("SL001", title="Tales of Freetown Market"))  
  
# Return a book  
print(operations.return\_book("SL001", 1)) # Should work  
  
# Delete a book  
print(operations.delete\_book("SL001")) # Should work after return  
print(operations.delete\_book("SL003")) # Error: Book not found  
  
# Try to borrow with max limit  
print(operations.borrow\_book("SL001", 1)) # Should work  
print(operations.borrow\_book("SL002", 1)) # Should fail: 3 books limit

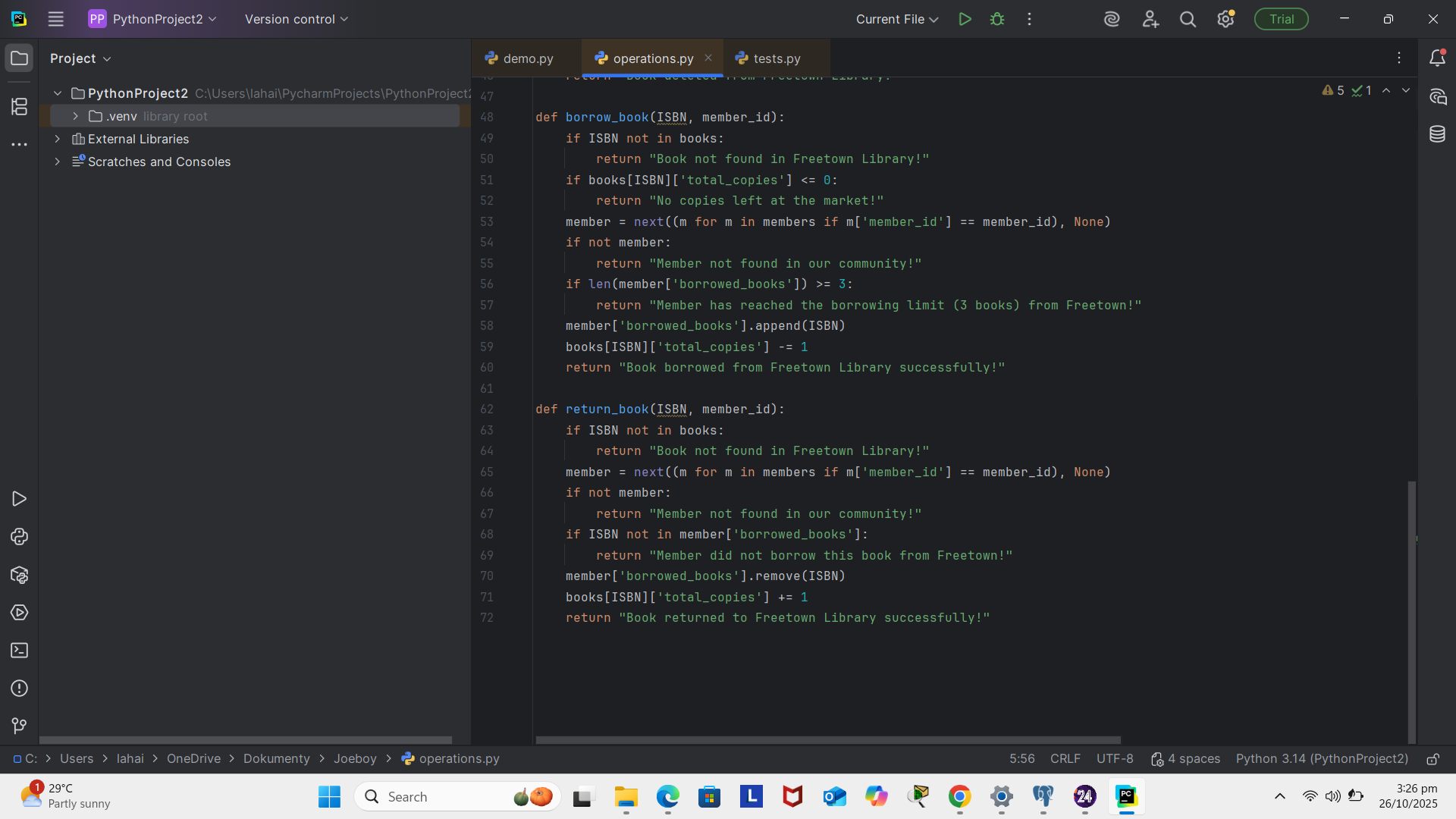
**TEST.PY**

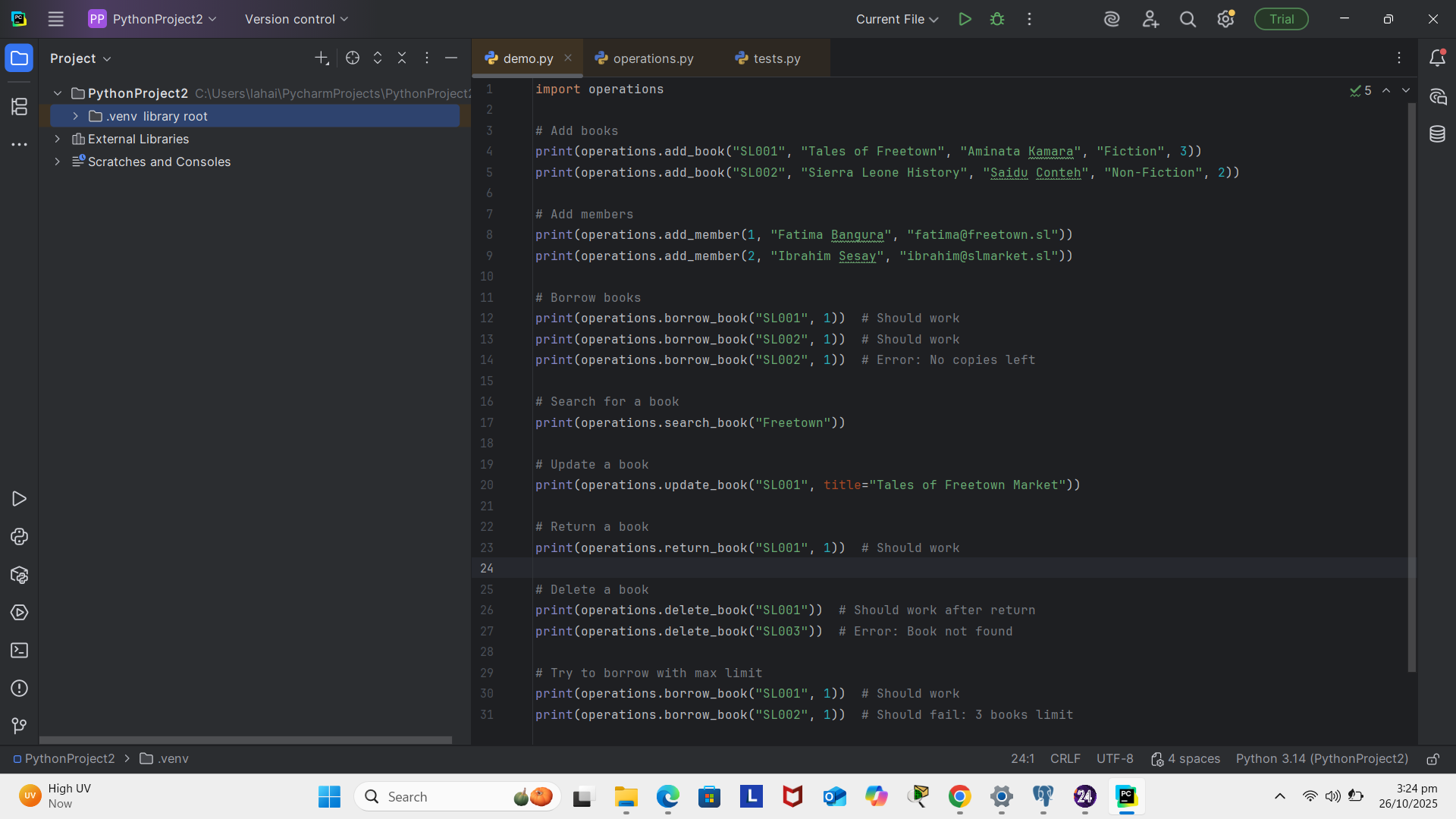


**OPERATIONS.PY**





****

**DEMO.PY**

**RESULT**