

**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 : Mr.Elijah Fullah

Name of Student/s : Juliet Joella Sarah Kamara

Student ID No. : 905005153

Class : DIT 1101F

Year/Semester : 2/1

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: 19TH OCTOBER 2025

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 outlines my selections for dictionary, list, and tuple, tailored to the brief's demands.

The dictionary for 'lib\_books' enables rapid key-based access using book\_code, storing book\_title, book\_author, book\_type, and copy\_num. For a system handling additions and searches via create\_book() and locate\_book(), it avoids duplicates and supports quick operation. Ideal for library catalogs.

A list for 'lib\_users' allows dynamic user management, with dictionaries holding user\_id, user\_name, user\_mail, and lent\_books. It facilitates creations and lends/reclaims.

The tuple for 'lib\_types' fixes 'Fiction', 'Non-Fiction', 'Sci-Fi', ensuring no modifications during validations in create\_book() and alter\_book().

These picks create a functional system, integrating with UML and core features). Focused on practical use.

**UML DIAGRAM SKETCH (Hand Drawn)**

**Lib\_OPS.PY CODE**

lib\_books = {}

lib\_users = []

lib\_types = ('Fiction', 'Non-Fiction', 'Sci-Fi')

def create\_book(book\_code, book\_title, book\_author, book\_type, copy\_num):

if book\_code in lib\_books:

return "Book code already exists in library!"

if book\_type not in lib\_types:

return "Book type not valid for library!"

if copy\_num < 1:

return "Copy number must be at least 1!"

lib\_books[book\_code] = {"book\_title": book\_title, "book\_author": book\_author, "book\_type": book\_type, "copy\_num": copy\_num}

return "Book created in library system!"

def create\_user(user\_id, user\_name, user\_mail):

for user in lib\_users:

if user['user\_id'] == user\_id:

return "User ID already in system!"

lib\_users.append({'user\_id': user\_id, 'user\_name': user\_name, 'user\_mail': user\_mail, 'lent\_books': []})

return "User created in library!"

def locate\_book(search\_word):

hits = []

for book\_code, details in lib\_books.items():

if search\_word.lower() in details['book\_title'].lower() or search\_word.lower() in details['book\_author'].lower():

hits.append({"book\_code": book\_code, "book\_title": details['book\_title'], "book\_author": details['book\_author'], "book\_type": details['book\_type'], "copy\_num": details['copy\_num']})

return hits if hits else "No hits in library catalog!"

def alter\_book(book\_code, \*\*mods):

if book\_code not in lib\_books:

return "Book code not in library!"

if 'book\_type' in mods and mods['book\_type'] not in lib\_types:

return "Invalid book type!"

if 'copy\_num' in mods and mods['copy\_num'] < 1:

return "Copy number must be positive!"

lib\_books[book\_code].update(mods)

return "Book altered successfully!"

def erase\_book(book\_code):

if book\_code not in lib\_books:

return "Book code not found!"

for user in lib\_users:

if book\_code in user['lent\_books']:

return "Can't erase: Book is lent out!"

del lib\_books[book\_code]

return "Book erased from library!"

def lend\_book(book\_code, user\_id):

if book\_code not in lib\_books:

return "Book code not found!"

if lib\_books[book\_code]['copy\_num'] < 1:

return "No copies to lend!"

user = next((u for u in lib\_users if u['user\_id'] == user\_id), None)

if not user:

return "User ID not found!"

if len(user['lent\_books']) >= 3:

return "User at lend limit (3 books)!"

user['lent\_books'].append(book\_code)

lib\_books[book\_code]['copy\_num'] -= 1

return "Book lent successfully!"

def reclaim\_book(book\_code, user\_id):

if book\_code not in lib\_books:

return "Book code not found!"

user = next((u for u in lib\_users if u['user\_id'] == user\_id), None)

if not user:

return "User ID not found!"

if book\_code not in user['lent\_books']:

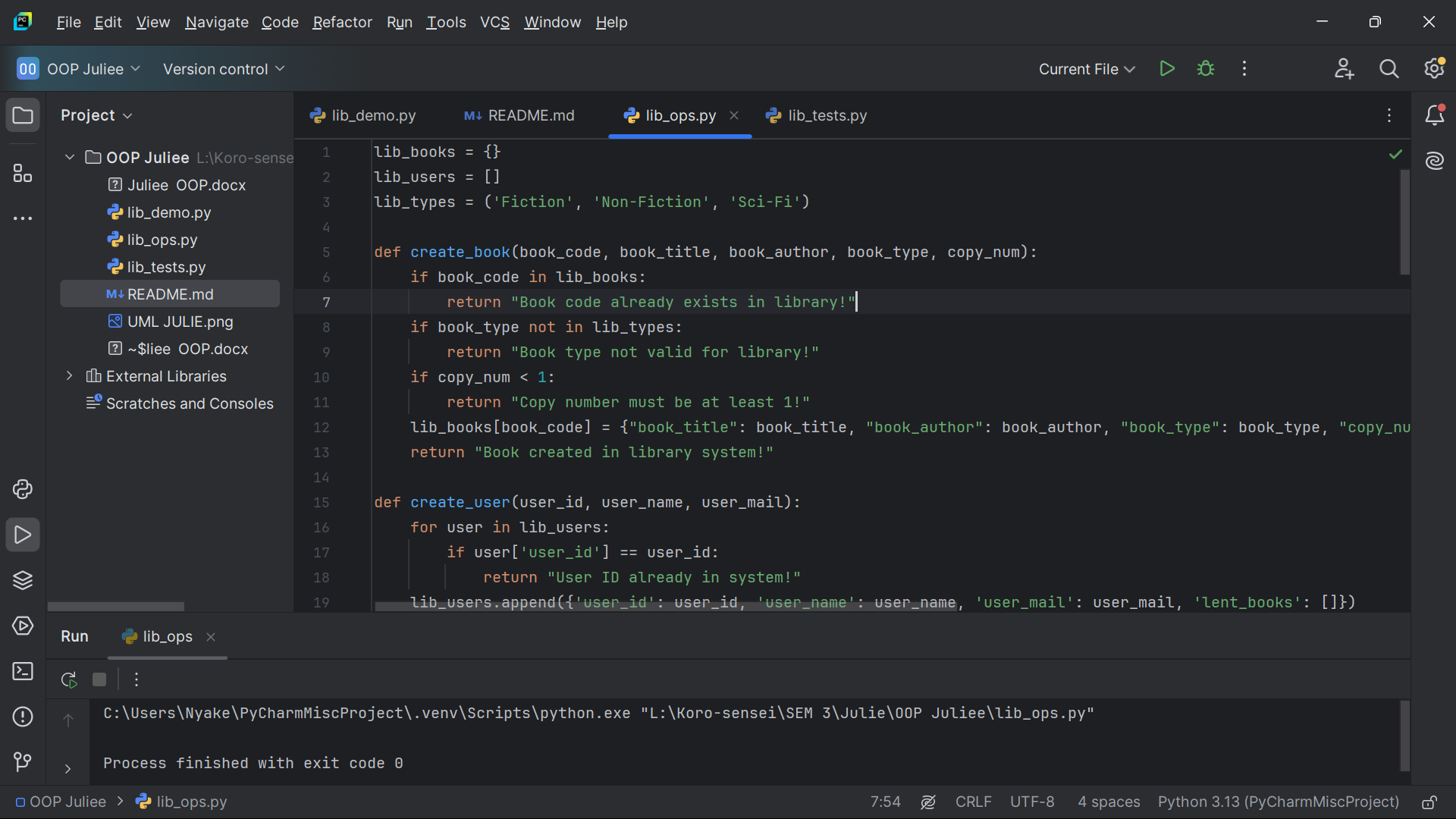
return "Book not lent to this user!"

user['lent\_books'].remove(book\_code)

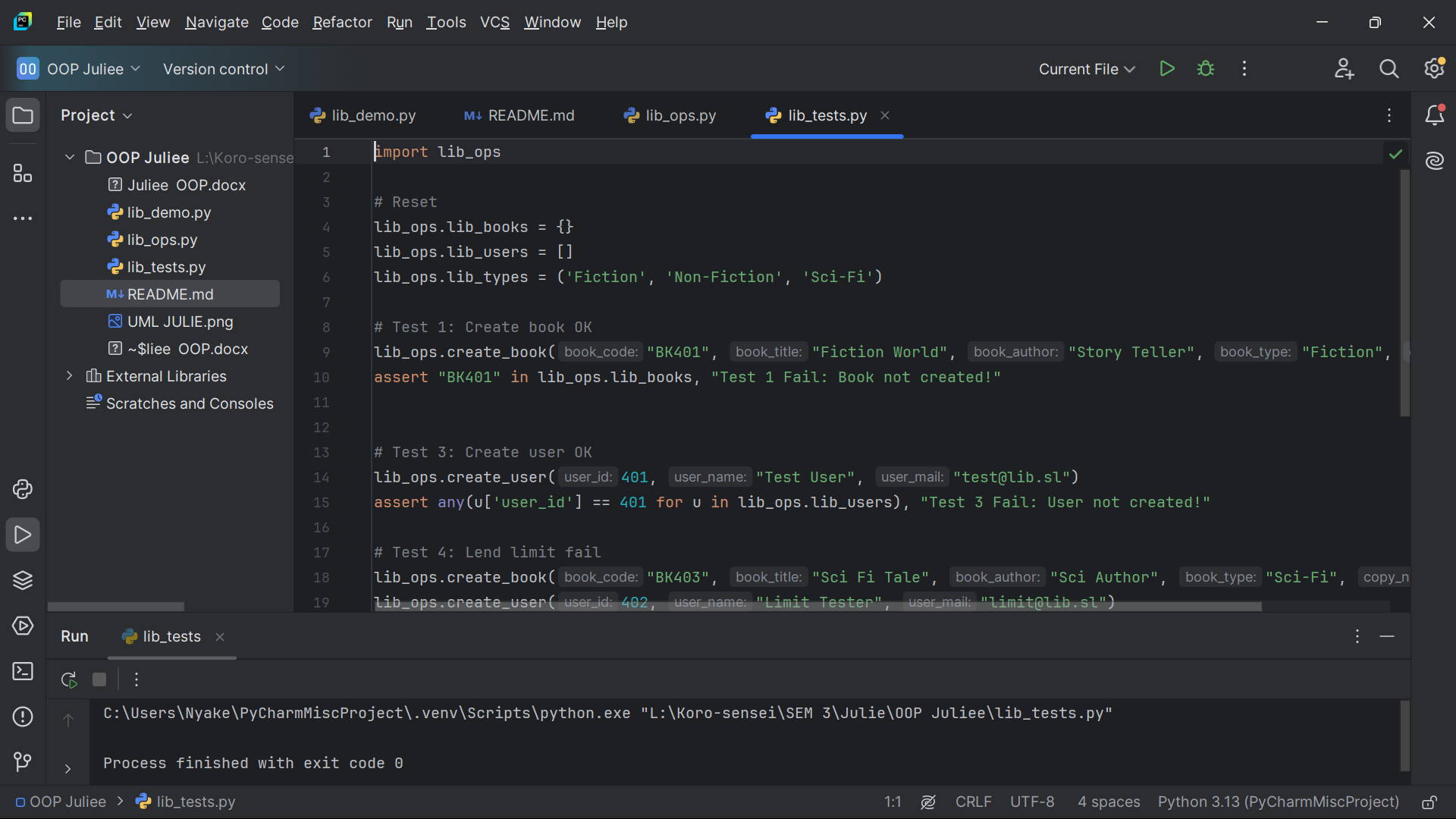
lib\_books[book\_code]['copy\_num'] += 1

return "Book reclaimed successfully!"

**Lib\_OPS.PY**

****

**LIB\_TEST.PY**



**LIB\_DEMO.PY**

