CSC1005: Introduction to Computer Engineering

Programming and Applications

Assignment 3

Assignment description:
This assignment will be worth 9% of the final grade.

You should write your code for each question in a .py file (please name it using the question name, e.g. q1.py). Please pack all your .py files into a single .zip file, name it using your student ID (e.g. if your student ID is 123456, then the file should be named as 123456.zip), and then submit the .zip file via Blackboard.

Please create a Word document that provides a detailed explanation of the code for each question. This document should clearly outline the logic behind your code, specify the expected inputs, and describe the anticipated outputs. Make sure the explanations are easy to understand and provide enough context for anyone reviewing your work. Include this Word document in the .zip file along with your code.

Please be aware that the teaching assistant may ask you to explain your code to verify that it was written by you. Additionally, your submission may be checked for similarities with other students' work using Blackboard to ensure academic integrity.

This assignment is due on 5:00PM, 28 Nov (Friday). For each day of late submission, you will lose 10% of your mark in this assignment. If you submit more than three days later than the deadline, you will receive zero in this assignment.

Question 1: Matrix Class (25% of this assignment)
Write a Python class called Matrix that represents a 2x2 matrix and implements matrix addition, subtraction, and multiplication. The class should accept two 2x2 matrices as inputs and output the results as new matrices in the same format.

For example, if matrix1 = [[1, 2], [3, 4]] and matrix2 = [[5, 6], [7, 8]], your program should be able to perform:

matrix1 + matrix2 resulting in [[6, 8], [10, 12]]
matrix1 - matrix2 resulting in [[-4, -4], [-4, -4]]
matrix1 * matrix2 resulting in [[19, 22], [43, 50]]
Note: Handle cases where the input matrices do not have the correct dimensions by displaying an appropriate error message.

Question 2: Complex Number Class (30% of this assignment)

Create a Python class ComplexNumber that represents a complex number. Implement methods for addition, subtraction, multiplication, and division of complex numbers. The complex numbers should be inputted in the form of strings (e.g., "3+4i" or "-2-5i") and the output should also be in the form of strings.

For example, if the input is:

num1 = "3+4i" and num2 = "1-2i", your program should output:
Addition: "4+2i"
Subtraction: "2+6i"
Multiplication: "11-2i"
Division: "-1.4+2.2i"
Note: Handle all potential invalid inputs and provide an appropriate error message if attempted.

Question 3: Library Management System (45% of this assignment)

Design and implement a simple library management system that handles books and library members.

1. Create a class Book with the following specifications:

Data fields:
ISBN (string) - unique identifier
title (string)
author (string)
is_available (boolean) - tracks if book is currently available

Methods:
check_out() - marks book as unavailable
return_book() - marks book as available

All data fields should be private with appropriate getters/setters

2. Create a class Member with:

Data fields:
member_id (string) - unique identifier
name (string)
borrowed_books (list of Book objects)

Methods:
borrow_book(book) - adds a book to borrowed_books (maximum 3 books allowed)
return_book(book) - removes a book from borrowed_books

3. Create a class Library that:

Data fields:

books (list of Book objects)

members (list of Member objects)

Methods:

add_book(book)

add_member(member)

check_out_book(isbn, member_id) - processes a book checkout

return_book(isbn, member_id) - processes a book return

get_available_books() - returns list of available books

get_member_books(member_id) - returns books borrowed by a member

Example Usage:

```
# Create library
library = Library()

# Create and add books
book1 = Book("123-456", "Python Programming", "John Smith")
book2 = Book("789-012", "Data Structures", "Jane Doe")
library.add_book(book1)
library.add_book(book2)

# Create and add member
member = Member("M001", "Alice Brown")
library.add_member(member)

# Check out a book
library.check_out_book("123-456", "M001")
```

Your solution must:

- Implement proper error handling for:
  - ✓ Checking out an unavailable book
  - ✓ Exceeding maximum borrowed books limit
  - ✓ Invalid ISBN or member ID
- Include input validation
- Use appropriate data types