Day-22 Morning Assessment

1. Auto-save data on every operation:

 def addBook(self, book):

    self.books[book.isbn] = book

[self.save](http://self.save)()

2. Search books by partial title/author (case-insensitive)

 def searchBooks(self, query):

    query = query.lower()

     return [book for book in self.books.values()

if query in book.title.lower() or query in [book.author](http://book.author).lower()]

 3. Sorting books

title = sorted(self.books.values(), key=lambda b: b.title)

author = sorted(self.books.values(), key=lambda b: [b.author](http://b.author))

availability = sorted(self.books.values(), key=lambda b: b.is\_available)

 4. List comprehension for borrowed books

 borrowed\_books = [book for book in self.books.values() if not book.is\_available]

 5. import csv

  def export\_to\_csv(self, filename="library\_data.csv"):

     with open(filename, mode="w", newline="") as file:

         writer = csv.writer(file)

         writer.writerow(["ISBN", "Title", "Author", "Available"])

for book in self.books.values():

writer.writerow([book.isbn, book.title, [book.author](http://book.author), book.is\_available])

 6. Use dataclasses instead of normal classes

 from dataclasses import dataclass

 @dataclass

class Book:

     isbn: str

     title: str

     author: str

     is\_available: bool = True

 @dataclass

class Member:

     member\_id: str

     name: str

     borrowed\_books: list = None

 7. Use zip() to pair members with borrowed books

 def borrowed\_report(self):

    for member in self.members.values():

paired = list(zip([[member.name](http://member.name)] \* len(member.borrowed\_books), member.borrowed\_books))

print(paired)

 8. Regex for ISBN validation

 import re

 def validate\_isbn(isbn):

    pattern = r"^(97(8|9))?\d{9}(\d|X)$"

    return bool(re.match(pattern, isbn))

Advanced OOP Concepts

9. StaffMember subclass with extra permissions

 class Member:

    def \_\_init\_\_(self, member\_id, name):

        self.member\_id = member\_id

[self.name](http://self.name) = name

class StaffMember(Member):

    def remove\_book(self, library, isbn):

        if isbn in library.books:

            del library.books[isbn]

10. Operator overloading for book comparison

 class Book:

    def \_\_init\_\_(self, isbn, title, author):

        self.isbn = isbn

        self.title = title

[self.author](http://self.author) = author

   def \_\_eq\_\_(self, other):

        return self.isbn == other.isbn

  def \_\_lt\_\_(self, other):

        return self.isbn < other.isbn