PYTHON IS FOR EVERYONE

Tutorial 14:

PYTHON PROGRAMMING - FINAL PROJECT IN GOOGLE COLAB

Jeff Gentry

@www.linkedin.com/in/jefferycharlesgentry

Project Overview

In this project, you will create a library management system that allows users to:

- Add new books to the library.
- View all available books.
- Borrow a book.
- Return a book.
- Search for a book by title or author.

Objectives

- Apply Object-Oriented Programming (OOP) principles.
- Use file handling to store and retrieve data.
- Implement user input and command-line interaction.

Define the Classes

Book Class: This class will represent a book with attributes like title, author, and availability status.

utorial_14.ipynb

Edit View Insert Runtime Tools Help All changes saved

+ Text

```
class Book:
    def __init__(self, title, author):
        self.title = title
        self.author = author
        self.available = True

    def __str__(self):
        return (f"{self.title} by {self.author} - "
                f"{'Available' if self.available else 'Not Available'}")
```

Define the Classes

Library Class: This class will manage the collection of books and provide methods to add, borrow, return, and search for books.

```python
class Library:
    def __init__(self):
        self.books = []

    def add_book(self, book):
        self.books.append(book)

    def view_books(self):
        for book in self.books:
            print(book)

    def borrow_book(self, title):
        for book in self.books:
            if book.title == title and book.available:
                book.available = False
                print(f"You have borrowed '{title}'.")
                return
        print(f"'{title}' is not available.")

    def return_book(self, title):
        for book in self.books:
            if book.title == title and not book.available:
                book.available = True
                print(f"You have returned '{title}'.")
                return
        print(f"'{title}' was not borrowed.")

    def search_book(self, search_term):
        found_books = [
        book for book in self.books
        if (search_term.lower() in book.title.lower() or
            search_term.lower() in book.author.lower())
    ]
        if found_books:
            for book in found_books:
                print(book)
        else:
            print("No books found.")
```

Implement File Handling

Save Books to File:

```python
def save_books(self, filename):
    with open(filename, 'w') as file:
        for book in self.books:
            status = '1' if book.available else '0'
            file.write(f"{book.title},{book.author},{status}\n")
```

It’s important that this is part of the Library Class (it’s in the same Colab Code cell).

Implement File Handling

Load Books from File:

```python
def load_books(self, filename):
    try:
        with open(filename, 'r') as file:
            for line in file:
                title, author, status = line.strip().split(',')
                book = Book(title, author)
                book.available = status == '1'
                self.add_book(book)
    except FileNotFoundError:
        print("No previous library data found.")
```

It’s important that this is part of the Library Class (it’s in the same Colab Code cell).

class Library (Full)

```python
class Library:
    def __init__(self):
        self.books = []

    def add_book(self, book):
        self.books.append(book)

    def view_books(self):
        for book in self.books:
            print(book)

    def borrow_book(self, title):
        for book in self.books:
            if book.title == title and book.available:
                book.available = False
                print(f"You have borrowed '{title}'.")
                return
        print(f"'{title}' is not available.")

    def return_book(self, title):
        for book in self.books:
            if book.title == title and not book.available:
                book.available = True
                print(f"You have returned '{title}'.")
                return
        print(f"'{title}' was not borrowed.")

    def search_book(self, search_term):
        found_books = [
        book for book in self.books
        if (search_term.lower() in book.title.lower() or
            search_term.lower() in book.author.lower())
    ]
        if found_books:
            for book in found_books:
                print(book)
        else:
            print("No books found.")

    def save_books(self, filename):
      with open(filename, 'w') as file:
          for book in self.books:
              status = '1' if book.available else '0'
              file.write(f"{book.title},{book.author},{status}\n")

    def load_books(self, filename):
      try:
          with open(filename, 'r') as file:
              for line in file:
                  title, author, status = line.strip().split(',')
                  book = Book(title, author)
                  book.available = status == '1'
                  self.add_book(book)
      except FileNotFoundError:
          print("No previous library data found.")
```

Create the Main Program

Now, you will create a simple command-line interface to interact with the library management system.

```python
def main():
    library = Library()
    library.load_books('library.txt')

    while True:
        print("\nLibrary Management System")
        print("1. Add Book")
        print("2. View Books")
        print("3. Borrow Book")
        print("4. Return Book")
        print("5. Search Book")
        print("6. Save and Exit")
        choice = input("Choose an option: ")

        if choice == '1':
            title = input("Enter book title: ")
            author = input("Enter book author: ")
            library.add_book(Book(title, author))
            print(f"Book '{title}' added.")
        elif choice == '2':
            library.view_books()
        elif choice == '3':
            title = input("Enter the title of the book to borrow: ")
            library.borrow_book(title)
        elif choice == '4':
            title = input("Enter the title of the book to return: ")
            library.return_book(title)
        elif choice == '5':
            search_term = input("Enter title or author to search: ")
            library.search_book(search_term)
        elif choice == '6':
            library.save_books('library.txt')
            print("Library data saved. Exiting...")
            break
        else:
            print("Invalid option. Please try again.")

if __name__ == "__main__":
    main()
```

Conclusion

In this final project, you have created a simple library management system using Object-Oriented Programming principles. You implemented classes for books and the library, added functionality for managing books, and utilized file handling to persist data. This project serves as a practical application of the concepts learned in previous tutorials and can be expanded with additional features as you continue to develop your programming skills.

Next Steps

Python Is For Everyone: Intermediate Tutorials

FOLLOW ME

for more tips you didn’t know you needed

Jeff Gentry

@www.linkedin.com/in/jefferycharlesgentry