Python Project: Movie Theatre

Description of commands in the project:

- list: Lists all available movies along with the number of seats available for each.
- add: Adds a new movie to the list of available movies. Takes the movie title and the number of available seats as arguments.
- book: Books a specified number of seats for a given movie. Takes the movie title and the number of seats to book as arguments.

Scripts that can be used from the command line:

- To list all movies: python script.py list
- To add a new movie: python script.py add "New Movie" 20
- To book seats: python script.py book "New Movie" 2
- Adding Another New Movie: python script.py add "Titanic" 50
- Booking Seats for Another Movie: python script.py book "Titanic" 5

Key Python Concepts include:

- classes
- file I/O
- command-line arguments
- error handling

```
import sys

import json
```

<u>- Class definition for Movie</u>

```
class Movie:

    Constructor initializes movie title and available seats

    def __init__(self, title, available_seats):

        self.title = title

        self.available_seats = available_seats
```

<u>-Function to save the current state of movies to a JSON file</u>

```
def save_state(movies):

    with open('movies.json', 'w') as f:

        # Serialize the list of Movie objects into JSON and write to file

        json.dump({m.title: m.available_seats for m in movies}, f)
```

```
-Function to load the current state of movies from a JSON file
def load_state():
    try:
        with open('movies.json', 'r') as f:
            Deserialize the JSON from the file into a Python object
            data = json.load(f)
            Create a list of Movie objects based on the deserialized data
            return [Movie(title, seats) for title, seats in data.items()]
    except FileNotFoundError:
         If file is not found, return an empty list.
        return []

Main function to handle various commands
def main(args):
    # Load the list of movies from file
    movies = load_state()

    Check if there are enough arguments
    if len(args) < 2:
        print("Usage: python script.py <command> <arguments>")
        sys.exit(1)  # Exit the program with an error code

Retrieve the command from command-line arguments
    command = args[1].lower()

    List available movies
    if command == "list":
        if not movies:
            print("No movies available.")
        else:
            for movie in movies:
                print(f"{movie.title}: {movie.available_seats} seats available")
```

<u>Add a new movie</u>

```
elif command == "add":
    if len(args) != 4:
        print("Usage: python script.py add <movie_title> <available_seats>")
    else:
        title = args[2]
        try:
            seats = int(args[3])
```

<u>Add a new Movie object to the list.</u>

```
            movies.append(Movie(title, seats))
            # Save the updated list to file
            save_state(movies)
            print(f"Added movie {title} with {seats} seats.")
        except ValueError:
            print("Invalid number of seats.")
```

<u>Book seats for a movie</u>

```
elif command == "book":
    if len(args) != 4:
        print("Usage: python script.py book <movie_title> <seats_to_book>")
    else:
        title = args[2]
        try:
            seats_to_book = int(args[3])
```

<u>Look for the movie by title</u>

```
            for movie in movies:
                    if movie.title == title:
                        # Check if enough seats are available
                        if movie.available_seats >= seats_to_book:
                            movie.available_seats -= seats_to_book
                            print(f"Booked {seats_to_book} seats for {title}")
                            Save the updated list to file
                            save_state(movies)
                            break
                        else:
                            print("Not enough seats available.")
                            break
                else:
                    print("Movie not found.")
            except ValueError:
                print("Invalid number of seats to book.")

    Invalid command entered
    else:
        print("Invalid command. Available commands are: list, add, book")

Entry point for the script
if __name__ == "__main__":
    main(sys.argv)
```