**1: функціональність (фільми та сеанси)**

**test\_01\_initial\_movies\_loaded\_successfully**:

Перевіряє, чи при створенні менеджера в нього автоматично завантажується 10 початкових фільмів.

*Чому?* Система стартує з очікуваним, коректним станом. Якщо тест падає - є проблема з початковою ініціалізацією даних.

**test\_02\_add\_unique\_movie\_successfully**:

Додає новий, унікальний фільм і перевіряє, що загальна кількість фільмів збільшилася на 1.

*Чому?* Тестує базову функцію додавання нових даних.

**test\_03\_add\_duplicate\_movie\_is\_ignored**:

Намагається додати фільм, який вже існує, і перевіряє, що кількість фільмів не змінилася.

*Чому?*  Забезпечує цілісність даних, не створювати дублікати.

**test\_04\_find\_movie\_by\_full\_title** та **test\_05\_find\_movies\_by\_keyword**:

Перевіряють, чи правильно працює пошук фільмів як за повною назвою, так і за ключовим словом (частиною назви).

*Чому?* Тестує одну з ключових функцій для користувача — пошук.

**test\_06\_find\_non\_existent\_movie\_returns\_empty**:

Шукає фільм, якого точно немає, і перевіряє, що результат — порожній список.

*Чому?* Перевіряє коректну обробку "нульового" результату, щоб програма не видавала помилку, коли нічого не знайдено.

**test\_07\_add\_screening\_successfully** та **test\_09\_get\_screening\_by\_valid\_id**:

Перевіряють успішне створення нового сеансу для існуючого фільму та його подальший пошук за ID.

*Чому?* Тестують базову логіку роботи з сеансами.

**test\_08\_add\_screening\_for\_non\_existent\_movie\_fails** та **test\_10\_get\_screening\_by\_invalid\_id\_returns\_none**:

Намагаються створити сеанс для неіснуючого фільму або знайти сеанс за недійсним ID. Очікується, що функція поверне None.

*Чому?* Це тести на відмову. Вони гарантують, що система коректно обробляє некоректні вхідні дані і не "падає".

**2: Логіка бронювання квитків**

**test\_11\_book\_tickets\_successfully**:

Бронює кілька квитків на сеанс і перевіряє, що кількість вільних місць зменшилась правильно.

*Чому?* Тестує основний сценарій бронювання.

**test\_12\_book\_tickets\_for\_non\_existent\_screening\_fails**:

Намагається забронювати квитки на сеанс, якого не існує.

*Чому?* Запобігає помилкам, коли користувач намагається взаємодіяти з неіснуючими даними.

**test\_13\_book\_more\_tickets\_than\_available\_fails**:

Намагається забронювати більше квитків, ніж є вільних місць, і перевіряє, що бронювання не відбулося, а кількість місць не змінилася.

*Чому?* Не дозволяє "продати" більше квитків, ніж є в залі.

**test\_14\_cancel\_booking\_successfully**:

Створює бронювання, а потім скасовує його. Перевіряє, що місця повернулися у вільний доступ.

*Чому?* Тестує логіку "зворотної" операції — скасування.

**test\_15\_cancel\_non\_existent\_booking\_fails**:

Намагається скасувати неіснуюче бронювання.

*Чому?* Перевіряє, що система не ламається від спроби скасувати те, чого немає.

**3: Тести, що впали**

**test\_16 \_get\_screenings\_for\_movie\_uses\_substring**:

Функція пошуку сеансів працює неточно. Коли шукають сеанси для фільму "Батько", вона також знаходить "Хрещений батько", бо "Батько" є частиною цієї назви.

*Чому?* Вказує на необхідність робити точне порівняння назв, а не пошук за підрядком.

**test\_17\_ screenings\_are\_not\_sorted\_chronologically**:

Коли для фільму є кілька сеансів, вони повертаються не в хронологічному порядку (раніший сеанс має йти першим).

*Чому?* Зручність для користувача вимагає логічного сортування.І

**test\_18\_cancel\_booking\_can\_result\_in\_negative\_seats**:

Критичну помилку в логіці. Тест штучно імітує ситуацію, коли після скасування бронювання кількість заброньованих місць стає від'ємною.

*Чому?* Демонструє, що логіка скасування не надійна і може призвести до некоректного стану системи.

**test\_20\_ add\_screening\_is\_ambiguous\_for\_same\_titles**:

Система не може розрізнити два фільми з однаковою назвою, але різними роками випуску (наприклад, "Соляріс" 1972 і 2002 року). Незрозуміло, для якого з них створюється сеанс.

*Чому?* Вказує на те, що для додавання сеансу потрібно використовувати унікальний ідентифікатор фільму, а не лише назву.

**test\_21\_ time\_is\_string\_not\_datetime**:

Час сеансу зберігається як звичайний рядок (string), а не спеціальний об'єкт datetime. Це дозволяє створювати сеанси з некоректним часом, як-от "це не дата".

*Чому?* Наголошує на важливості використання правильних типів даних для валідації та подальших операцій (наприклад, сортування).

**test\_22\_ booking\_with\_non\_integer\_crashes**:

Функція бронювання квитків "падає" з помилкою TypeError, якщо їй передати не число, а рядок (напр., "два").

*Чому?* Хороша програма не повинна падати від некоректних даних. Вона має їх обробити (наприклад, повернути None) і продовжити роботу. Цей тест перевіряє відмову.

**movie\_\_manager\_update: що змінилось?**

*Некоректний формат часу (test\_20\_time\_is\_string\_not\_datetime)*

У методі add\_screening додано блок try-except, який перевіряє, чи відповідає рядок screening\_time формату 'YYYY-MM-DD HH:MM'. Якщо ні, функція повертає None.

*Неоднозначність при додаванні сеансу (test\_19\_add\_screening\_is\_ambiguous\_for\_same\_titles)*

У методі add\_screening змінено логіку. Тепер, якщо пошук за назвою повертає не рівно один фільм (тобто 0 або більше 1), сеанс не створюється і повертається None.

*Неточний пошук сеансів (test\_166\_get\_screenings\_for\_movie\_uses\_substring)*

**Рішення:** Перевірку in замінено на точне порівняння == (movie\_title.lower() == s.movie\_title.lower()).

*Відсутність сортування сеансів (test\_17\_screenings\_are\_not\_sorted\_chronologically)*

**Рішення:** Результат роботи методу get\_screenings\_for\_movie тепер сортується за полем screening\_time перед поверненням.

*Бронювання нецілої кількості квитків (test\_21\_booking\_with\_non\_integer\_crashes)*

**Рішення:** На початку методу book\_tickets додано перевірку isinstance(num\_tickets, int). Якщо тип неправильний, функція повертає None.

*Від'ємна кількість місць при скасуванні (test\_18\_cancel\_booking\_can\_result\_in\_negative\_seats)*

**Рішення:** У методі cancel\_booking при поверненні квитків тепер використовується max(0, ...), що гарантує, що кількість заброньованих місць ніколи не опуститься нижче нуля.