Q1. Unittest가 객체지향이라서 가독성/유지보수 측면에서 더 좋은 것 아닌가?

A1. 우선 pytest는 def를 사용한 더 단순한 문법으로 작성되므로 가독성 측면에서 더 낫다고 할 수 있고, Fixture과 파라미터화 기능을 제공하기 때문에, 나중에 초기화 로직이 변경되더라도 Fixture만 수정하면 되고 하나의 테스트함수에서 파라미터화를 이용해 반복해서 처리할 수 있으므로 유지보수성 측면에서 더 뛰어나다고 할 수 있다.

Q2. fixture도 플러그인인가?

A2. fixture는 pytest의 플러그인이 아니라, pytest내의 한 기능이다. 플러그인은 pip install을 통해 설치한 후 사용할 수 있다.

참고: <https://docs.pytest.org/en/7.1.x/how-to/writing_plugins.html>

Q3. 파라미터화를 여러 함수에 적용할 수 있는가?

A3. 파이썬에서 데코레이터가 하나의 함수에만 적용되듯이, 데코레이터로 정의되는 파라미터화 또한 하나의 함수에만 적용된다고 할 수 있다.

Q4. Verbosity\_test\_cases에서의 상세도가 의미하는 것이 정확히 무엇인지?

A4. pytest의 Verbosity(상세도)는 테스트 실행 시 출력되는 정보의 양을 제어하는 옵션을 의미한다. 상세도를 조절함으로써, 테스트 실행 결과를 더 간략하게 또는 더 자세하게 볼 수 있다.

Q5. Pytest를 쓰지 않아도 27p 코드 돌아가는 거 아닌가?

A5. 실행은 되지만 결과는 보이지 않음. pytest는 test\_로 쓰여있는 함수를 자동으로 실행해주고 결과를 보여준다.

Q6. 플러그인 사용사례는 무엇이 있는지?

A6. pytest의 핵심내용 부분에서 ‘데이터베이스, 웹 및 API 테스트와 같은 복잡한 테스트 사례에서도 유용’하다고 한 것이 바로 플러그인을 사용하여 가능한 것인데, pytest-django, pytest-flask와 같은 플러그인을 사용하여 데이터베이스와의 상호작용을 테스트할 수 있다.