>> тест pytest для asyncio домашки, добавил только принты, но код виснет.

принты захватываются и выводятся после теста. решение: pytest -s

Мысли по решению самого задания 3-блока курсов kts по основам асинхронного программирования (asyncio):

Стоит еще раз про воркеры проговорить!

chain-ы из домашнего задания

должны постоянно работать

если у вас внутри нет какого то бесконечного цикла

который еще раз всю логику внутри воркера повторит - функция закончит свое выполнение

Например если мы вызвали обычную функцию и она дошла до конца - она закончила выполнения

если мы создали такску asyncio.create_task() - то же самое

наша функция (короутина) просто так не заработает еще раз

Вот пример того как любой chain из домашки должен работать

твой chain - воркер как из примера лекций они запускаются как таски у них внутри while True например они берут из какой то очереди данные await inbound.get() и подвисают на этом месте пока мы в эту очередь ничего не положим дальше они делают что то с этой порцией данных обрабатывают ее делают походы в сеть, что угодно и в конце например в другую очередь кладут ответ, чтобы другие элементы нашей программы получили эти данные и как-то их обработали у нас While True переходим на новую итерацию цикла и снова лочимся на await inbound.get() пока не получим данные. Из-за того, что есть while True - у нас функция бесконечная и именно поэтому она всегда у нас в будет шуршать в event loop

(только если мы ее специально не отменим).

create_task как раз запускает выполнение в event_loop. Просто когда тест проходит, вызывается loop.close() и все незавершённые таски падают в исключение, казалось бы, таски отмены бронирования просто не успевают отработать....

...но таки можно их всех обработать

например кто-то первый закончил работу

и мы получаем незавершенные таски отменяем их

+ мы можем перехватывать исключения

исключение asyncio.CancelledError вызывается при отмене

try: делаем что-то
    return что-то
except asyncio.CancelledError:
    в этом случае делаем что-то