Вариант 1

Задача 1

Напишите функцию `deepPropertyCount`, которая принимает объект и возвращает количество всех свойств, включая вложенные. Если свойство также является объектом, необходимо рекурсивно считать его вложенные свойства.

Например:
const data = { name: 'Alice', details: { age: 25, address: { city: 'New York', zip: 10001 } } };

Корректный подсчет свойств на всех уровнях вложенности — 10 баллов.
Корректная работа с вложенными объектами через рекурсию — 5 баллов.

Задача 2

Напишите функцию `groupBy`, которая принимает массив объектов и строку `property`. Функция должна группировать объекты по значению указанного свойства и возвращать объект, где ключи — это уникальные значения этого свойства, а значения — массивы объектов с соответствующими значениями.

Например:
const users = [{ name: 'Alice', group: 'admin' }, { name: 'Bob', group: 'user' }, { name: 'Charlie', group: 'admin' },];

Корректное создание объекта с уникальными значениями ключей — 10 баллов.
Правильное распределение объектов по массивам на основе свойства — 10 баллов.

Задача 3

Напишите асинхронную функцию `fetchWithTimeout`, которая принимает URL и максимальное время ожидания (в миллисекундах) для выполнения запроса. Если запрос выполняется дольше указанного времени, функция должна отклоняться с сообщением "Время ожидания истекло". Используйте `fetch` и `Promise.race`. URL любой на Ваш выбор.

Использование `Promise.race` для обработки времени ожидания — 15 баллов.
Обработка ошибки и возврат корректного сообщения при превышении времени — 10 баллов.

Задача 4

Напишите функцию `sequentialFetch`, которая принимает массив URL и загружает данные по каждому URL с задержкой в 500 миллисекунд между запросами. Если какой-либо запрос завершился ошибкой, остальные запросы не должны выполняться, а функция должна возвращать сообщение "Ошибка в загрузке данных".

Корректная последовательная загрузка с задержкой — 10 баллов.
Остановка выполнения и возврат ошибки при неудачном запросе — 10 баллов.

Задача 5
Создайте асинхронный генератор `paginatedFetch`, который принимает URL и максимальное количество страниц `maxPages`. Генератор должен последовательно загружать данные с каждой страницы до достижения `maxPages` или пока данные не перестанут приходить. Если ответ пустой, генератор должен завершить выполнение.

Корректная реализация асинхронного генератора для загрузки данных постранично — 10 баллов.
Завершение генератора при пустом ответе или достижении `maxPages` — 10 баллов.