**🔹 Структура занятия (90 минут)**

**0–10 мин — Разогрев/общие вводные**

* Короткий брейншторм по теме («Что вы знаете о функциях в Python?»).
* Все участвуют: новички вспоминают азы, опытные делятся знаниями (повышает их вовлечённость).

**10–40 мин — Общий блок**

* Теория и демонстрация кода, доступная для всех.
* Например, тема: *циклы и условия* или *функции*.
* Новички получают базовое понимание.
* Продвинутые могут уточнять детали («а как это работает с генераторами?»).

**40–75 мин — Разделение на уровни**

* ✏️ **Новички**: делают простые упражнения (например, написать функцию, которая проверяет чётность числа).
* 🚀 **Продвинутые**: получают расширенные задания на ту же тему (написать функцию с параметрами по умолчанию, аннотациями типов, обработкой ошибок).
* Формат: вы даёте общий «каркас» задачи, а дальше два уровня сложности.

Пример:  
Задача: «Написать программу, которая работает с массивом чисел».

* Новички: находят сумму и среднее.
* Продвинутые: реализуют через list comprehension, используют statistics, а потом оптимизируют под большие данные.

**75–90 мин — Совместное обсуждение**

* Разбор решений (новички видят, куда можно расти).
* Опытные объясняют свои подходы простыми словами → закрепляют знания.
* Короткий Q&A.

**🔹 Советы**

1. **Всегда давай "ступеньки сложности"** — одно задание, но несколько уровней.
2. **Не дели группу жёстко** — пусть новички при желании пробуют продвинутые задачи, а сильные могут помочь слабым.
3. **Держи темп**: новички не должны скучать, а опытные — простаивать.
4. Используй **pair programming**: смешивай группы для совместных решений.