

- стиль програмування — спосіб мислення про код
- визначає:
  - ▶ як формується ідея
  - ▶ де зосереджена складність
  - ▶ що бачить читач

## Імперативний стиль:

- фокус: як виконати обчислення
- послідовність команд
- явне керування станом
- цикли, умови, змінні
- побічні ефекти — норма
- приклад:
  - ▶ `s=0; for x in data: if x>0: s+=x`

## Декларативний стиль:

- фокус: що треба отримати
- опис властивостей результату
- мінімум явного стану
- алгоритм прихований
- менше побічних ефектів
- приклад:
  - ▶ `s = sum(x for x in data if x>0)`

- імперативний стиль:
  - ▶ контроль виконання
  - ▶ прозорий порядок дій
  - ▶ більше коду
  - ▶ часто шумний код
- декларативний стиль:
  - ▶ читабельність
  - ▶ оптимізованість
  - ▶ менше деталей

## Що таке шумний код

- шумний код — код без нової семантики
- описує механіку, не сенс
- змушує думати про виконання
- приховує інваріанти задачі

### Види шуму:

- шум керування:
  - ▶ for, while, індекси
- шум стану:
  - ▶ `sum = sum + x`
- синтаксичний шум:
  - ▶ `std::vector<double>::const_iterator`
- церемоніальний шум:
  - ▶ ритуальні конструкції без логіки

# Чому шум — проблема

- знижує читабельність
- маскує логічні помилки
- ускладнює модифікації
- збільшує когнітивне навантаження
- імперативний стиль:
  - ▶ більше контролю
  - ▶ більше шуму
- декларативний стиль:
  - ▶ менше деталей
  - ▶ шум прихований

# Boilerplate code

- boilerplate — повторюваний стандартний код
- потрібен не для задачі
- диктується мовою або фреймворком
- не несе власної логіки

## Приклад:

- задача: вивести один рядок "hello world"
- Python: `print("hello world")`
- C++: `#include<cstdio> int main() {printf("hello world\n"); return 0;}`
- логіка: `print()/printf()`
- бойлерплейт: усе інше

## Ознаки бойлерплейту

- повторюється у багатьох файлах
- копіюється без розуміння
- не змінюється при зміні задачі
- легко винести в шаблон

# Шум vs Boilerplate

- шум:
  - ▶ будь-яка зайва механіка
  - ▶ не обов'язково повторювана
- boilerplate:
  - ▶ повторюваний шум
  - ▶ придатний до автоматизації
- шум і бойлерплейт не завжди зло
- часто плата за:
  - ▶ контроль
  - ▶ продуктивність
  - ▶ низький рівень

## Гарна практика:

- ідея має бути видна з першого погляду
- шум слід мінімізувати
- бойлерплейт слід ховати або генерувати
- стиль — інструмент, не догма