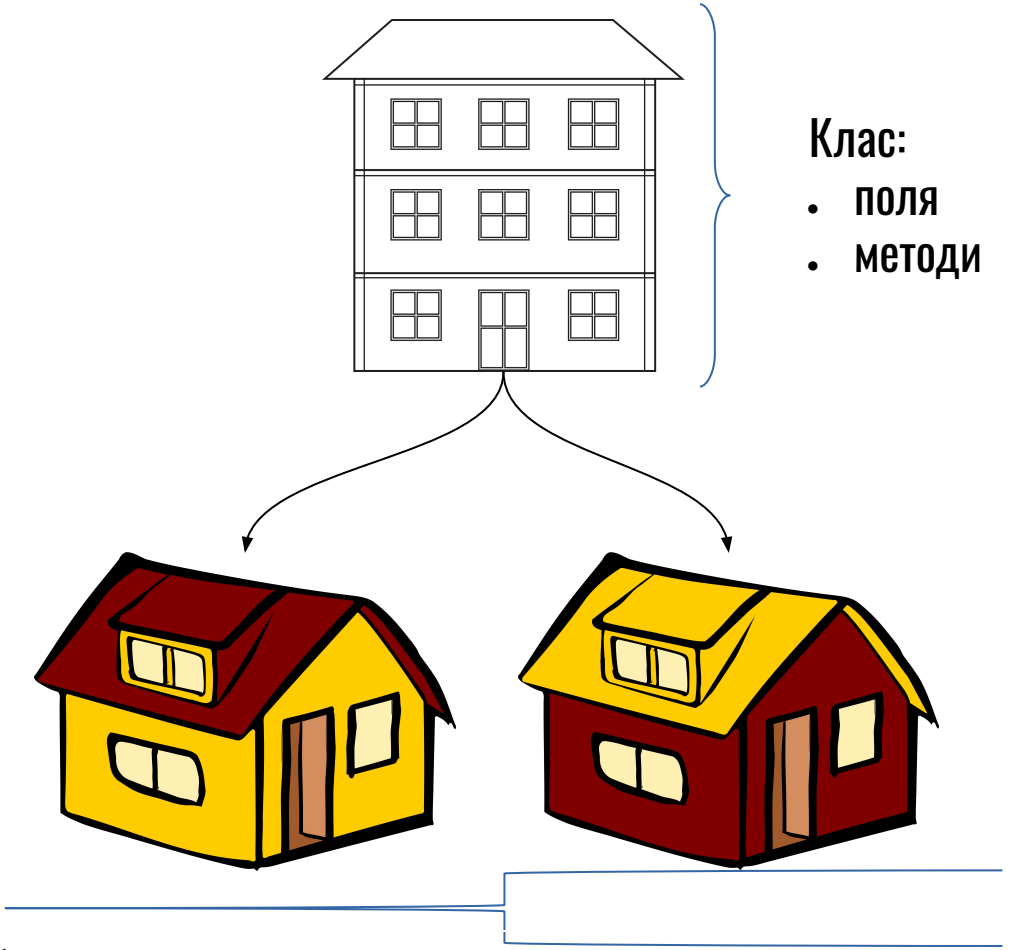
Класи та об'єкти

Класи

- Оголошення класів
- Поля класу
- Методи
- Конструктор класу
- Передача параметрів у методи

Що таке клас?

- Об'єктно-орієнтована програма будується з об'єктів.
- Клас являє собою "шаблон", який використовується для створення об'єктів.
- Клас визначає які дані може містити об'єкт і які операції можуть бути виконані



Клас:
- поля
- методи

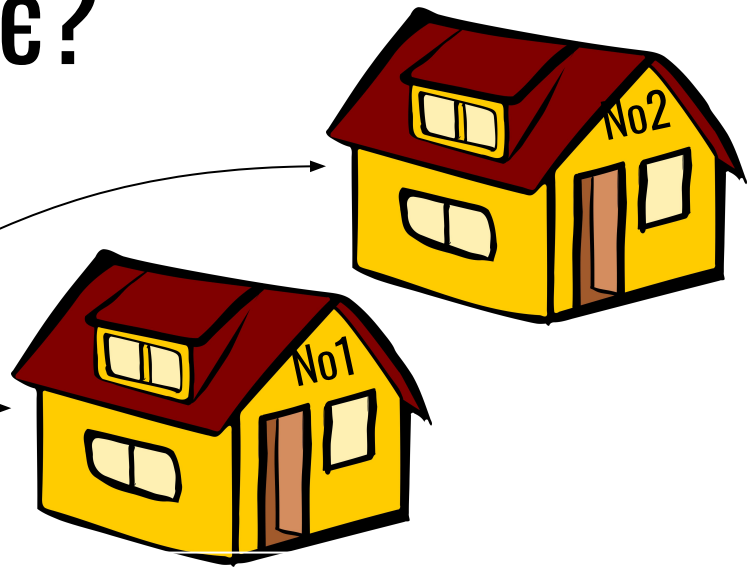
Об’єкт:
- поля — кожен об’єкт має власні значення в полях
- методи — поведінка визначена класом

І як це працює?

Клас:

- поля
  - колір
  - кількість вікон
- методи
  - перефарбувати

З класу створюється об’єкт



При створенні задаються значення полів:

- Поля зберігають стан об’єкту
- доки хтось цей стан не змінить
- Кожен об’єкт “отримує” набір методів

перефарбувати

Виклик методу **може** призвести до зміни стану **конкретного** об’єкту

Оголошення класів

```
public class SomeClassName extends ParentClassName
                                    implements SomeInterfaceName {
    variable defenition
    methods defenition
}
```

- Оголошення класу містить:
  - Модифікатори доступу: public, private ...
  - Ім'я класу
  - Ім'я батьківського класу після ключового слова extends
  - Імена інтерфейсів які реалізує даний клас після ключового слова implements
  - Тіло класу укладене в фігурні дужки {}

Оголошення полів класу

- У класі можна описати поля класу (fields or instance variable).
- Поля класу визначають, з яких даних складатимуться об'єкти цього класу.
- Поля можуть бути посиланнями на інші об'єкти або елементарними (примітивними) даними.
- Оголошення полів класу складається з:
  - Модифікатору доступу (може бути відсутнім)
  - Типу поля.
  - Імені поля.

Приклад оголошення полів класу

```
public class SomeClassName extends ParentClassName
                          implements SomeInterfaceName {
    public String variableTwo;
    double variableThree;
    private int variableOne;
}
```

Оголошення полів

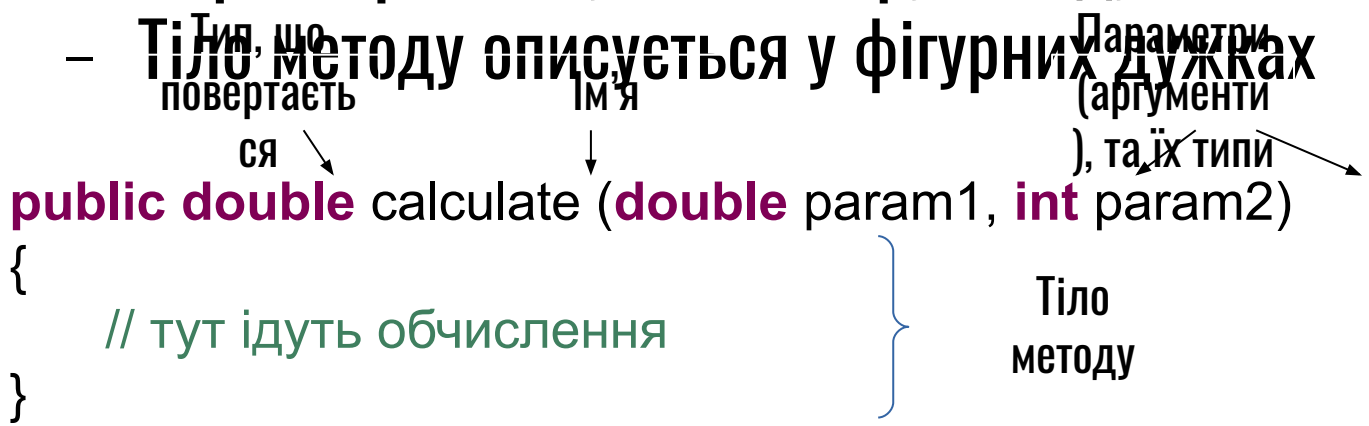
Модифікатор доступу Тип Ім’я

Модифікатори доступу

- Модифікатор визначає рівень доступу до змінної
  - Java визначає чотири рівні доступу
- Модифікатор **public** - поля доступні зі всіх класів.
- Модифікатор **private** - поля доступні тільки в межах даного класу.
  - З погляду використання інкапсуляції всі поля повинні бути помічені як private.

Методи

- Методи - це підпрограми, приєднані до конкретних визначень класів.
- При оголошенні методу задаються модифікатори доступу, тип результату що повертається, ім'я методу, список параметрів, список виключень (exceptions)
  - Параметри описуються в круглих дужках
  - Тіло методу описується у фігурних дужках



Сигнатура методу

- Ім'я методу і список його аргументів називається сигнатурою методу

Правила опису методів

- Якщо метод не повертає значення, його тип, що повертається, повинен бути **void**
  - **void** означає, що метод не має типу значення, що повертається.
  - виключення складає конструктор — він ніколи не повертає значення, при цьому **void** не застосовують.
- Описи методів розташовані всередині класу, на тому ж рівні вкладеності дужок, що й опис полів класу.
- Не може бути опису методу поза класом або всередині іншої методу чи блоку.

Перевантаження методів (overloading)

- Java дозволяє визначення всередині одного класу двох або більше методів з одним ім'ям, якщо оголошення їх параметрів різні.
  - У цьому випадку методи називають **перевантаженими**, а процес - **перевантаженням** методів.
- Перевантажені методи **повинні відрізнятися** за типом та / або кількості їх параметрів.
- Типи що повертаються у перевантажених методів можуть бути різні.
- **Java не розрізняє перевантажені методи по значенню що повертається**
- Коли Java зустрічає виклик перевантаженого методу,

Загальні правила перевантаження методів

- При перезавантаженні завжди слід дотримуватися наступних правил:
  - не використовувати складних варіантів перевантаження;
  - не використовувати перевантаження з однаковим числом параметрів;
  - замінювати при можливості перевантажені методи на кілька різних методів.

Конструктор класу

- Конструктор - це метод класу, який ініціалізує новий об'єкт після його створення.
- На відміну від методів, конструктори можуть мати модифікатори тільки доступу. Тому, конструктор не може бути abstract, final, native, static, or synchronized.
- Конструктори **не мають типу, що повертається,** вони не можуть повертати навіть тип **void**.
- Конструктори мають однакові імена з ім'ям класу в якому описані, а методи, мають імена відмінні від імені класу.

Конструктор за замовчуванням

- Ви можете не визначати конструктор класу.
- В такому випадку:
  - Компілятор сам створить конструктор для класу
  - Конструктор за замовчуванням викликає безаргументний конструктор суперкласу
  - Якщо батько класу явно не вказано то викличеться конструктор класу Object
- Якщо ж програміст створив для класу хоч один конструктор (не важливо з параметрами або без) — безаргументний конструктор за замовчуванням не створюється!

Передача параметрів у методи

- Параметри використовуються в тілі методу і під час виконання приймають значення переданих аргументів
- Параметри - список змінних використаних при декларації методу
- Аргументи - фактичні значення параметрів

Типи параметрів

- Можливе використання будь-яких типів даних для передачі в якості параметрів
- Ви можете передавати примітивні (byte, int, double і т.п.) і типи посилання (на масиви та об'єкти)

Змінна кількість параметрів

- Змінна кількість параметрів описується за допомогою конструкції … (три крапки)
-
```
public PrintStream printf (String format, Object ... args)
```
- усередині методу такий аргумент доступний як масив
-

```
System.out.println(args.length);
```