ROBOCODE
SENIOR EMBEDDED

ООП. Рівні доступу даних

LESSON 27. FOR STUDENTS

English Time

Час Англійської

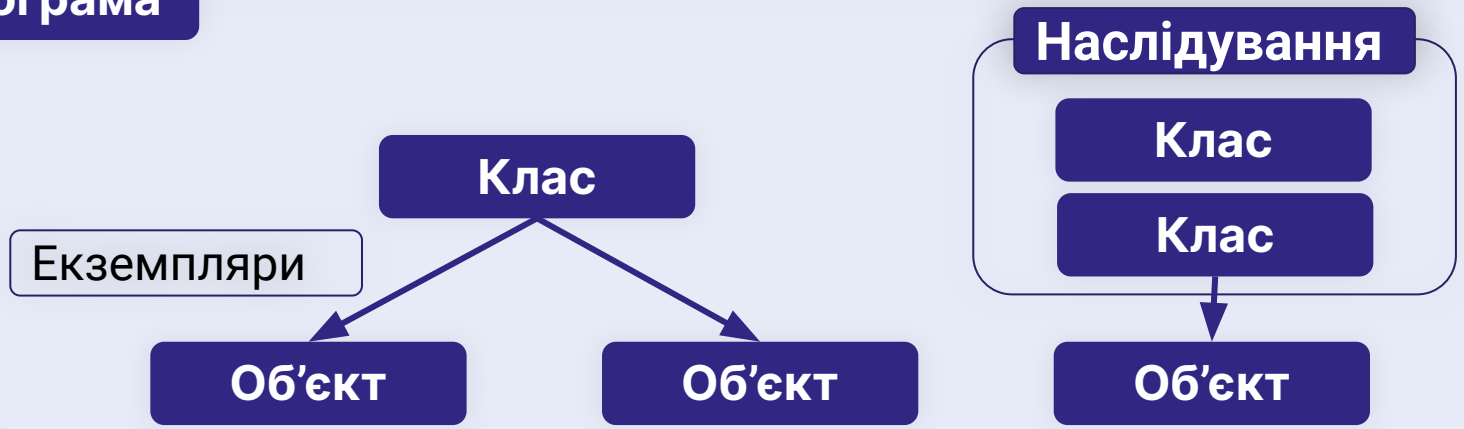

Library	Бібліотека
Header	Заголовок
Public	Публічний
Protected	Захищений

Повторюємо ООП

i

ООП - Об'єктноорієнтоване програмування - методологія програмування, заснована на представленні програм у вигляді сукупності об'єктів

Програма



i

Кожен з яких є екземпляром визначеного класу

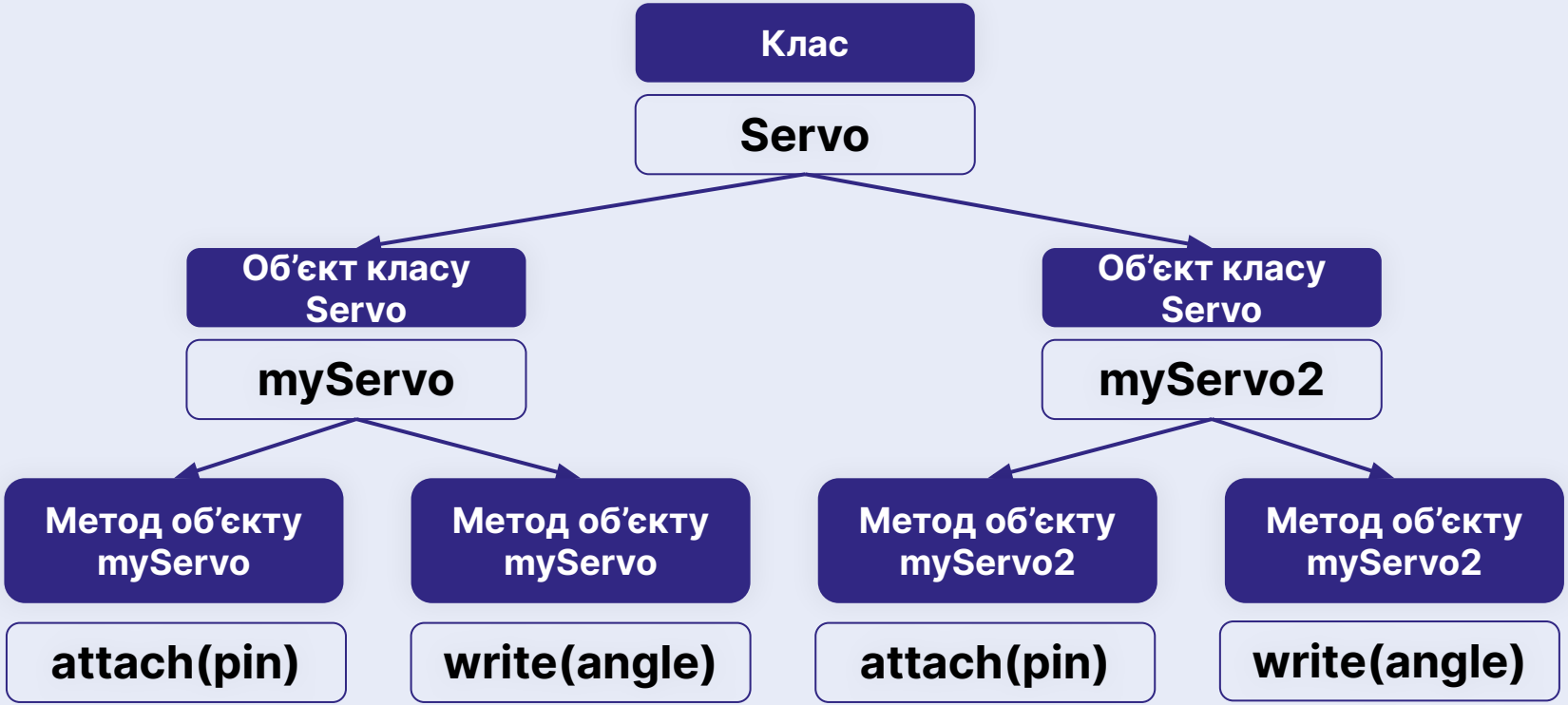
i

Класи формують ієрархію наслідування

Повторюємо ООП

i

Розглянемо структуру на прикладі класу **Servo**



Рівні доступу

i

Дані, що знаходяться в класі можуть мати різний рівень доступу. Це означає, що є методи та поля, що можна змінювати, а деякі можуть бути і зовсім недоступні

?

Для чого це?

i

Уявімо загальний клас “собака” із даними: лапи, гавкати та колір

i

У кожного виду є власні особливості, проте є дані, що змінювати *не можна* - це лапи та кожна собака вміє гавкати

i

Але є також і колір, що може бути для кожного виду різний

Рівні доступу

i Існують наступні основні рівні доступу до даних в класі

i **public** - дані, що можна отримати та змінювати поза класом

i **private** - дані, що можна отримати і змінити тільки в середині класу

i **protected** - дані, що можна змінювати та читати тільки в середині класу, або дочірньому класу

? Визначте рівень доступу даних класу “собака”: лапи, гавкати, колір

Рівні доступу

i

Як буде виглядати такі рівні доступу в вигляді коду

```
class Dog {
  public:
    int paws = 4;
    void gaw();
  private:
    String color;
};

void Dog::gaw() {
  // тут описуємо як гавкає собака
}
```

Рівні доступу

- В C++ рівні доступу можуть мати: поля, методи
- Ми можемо створювати дочірні класи, що успадковують загальні дані
- Наприклад, якщо ми маємо загальний клас “собака”, але є окремий дочірній клас “дика собака”, що має схожі характеристики, але має і відмінності

Рівні доступу

Нижче наведено код, як працює дочірній клас та поле protected

```
class Dog {
  public:
    int paws = 4;
    void gaw();
  private:
    String color;
  protected:
    bool domestic = true;
};

class WildDog: public Dog {
  protected:
    WildDog() {
      domestic = false;
    }
};
```

Рівні доступу

- Створимо власний клас для зчитування даних із сонару
- Поля із значенням портів, куди будемо підключати сонар
- Метод для налаштування сонару з визначенням режиму портів та роботи монітору порту
- Метод, що пише в монітор порту значення дистанції
- Метод, що повертає, як параметр, значення дистанції

! Завантажте код та відкрийте його