



# ЛЕКЦІЯ №1 ФУНДАМЕНТАЛЬНІ КОНЦЕПЦІЇ ООП

Клас, екземпляр

# ТЕМА 1. ФУНДАМЕНТАЛЬНІ КОНЦЕПЦІЇ ООП

1. Об'єкти
2. Клас
3. Імена класів
4. Синтаксис визначення класу
5. Конструктор класу
6. Створення екземпляру класу
7. Доступ до атрибутів

# ОПП

Об'єктно-орієнтоване програмування (ООП) - це метод структурування програми шляхом пакування пов'язаних властивостей та поведінки в окремі об'єкти.

У об'єктно-орієнтованому програмуванні використовують **класи**, що описують реально існуючі предмети і ситуації, а потім створюють **об'єкти** на основі цих описів. При написанні класу визначається **загальна поведінка** для цілої категорії об'єктів.

При створенні конкретних об'єктів (**екземплярів**) на основі класів, кожен об'єкт автоматично наділяється загальною поведінкою; після цього можна наділити кожен об'єкт **унікальними особливостями** на свій вибір.

# ОБ'ЄКТИ

Об'єкт має дві характеристики:

- атрибути;
- поведінка.

Розглянемо приклад. Припустимо, наш об'єкт – це папуга. Папуг має такі властивості:

Ім'я, вік, колір. Це **атрибути** .

Те, як папуга співає та танцює. Це **поведінка** .

# КЛАС

***Клас – це шаблон об'єкта.***

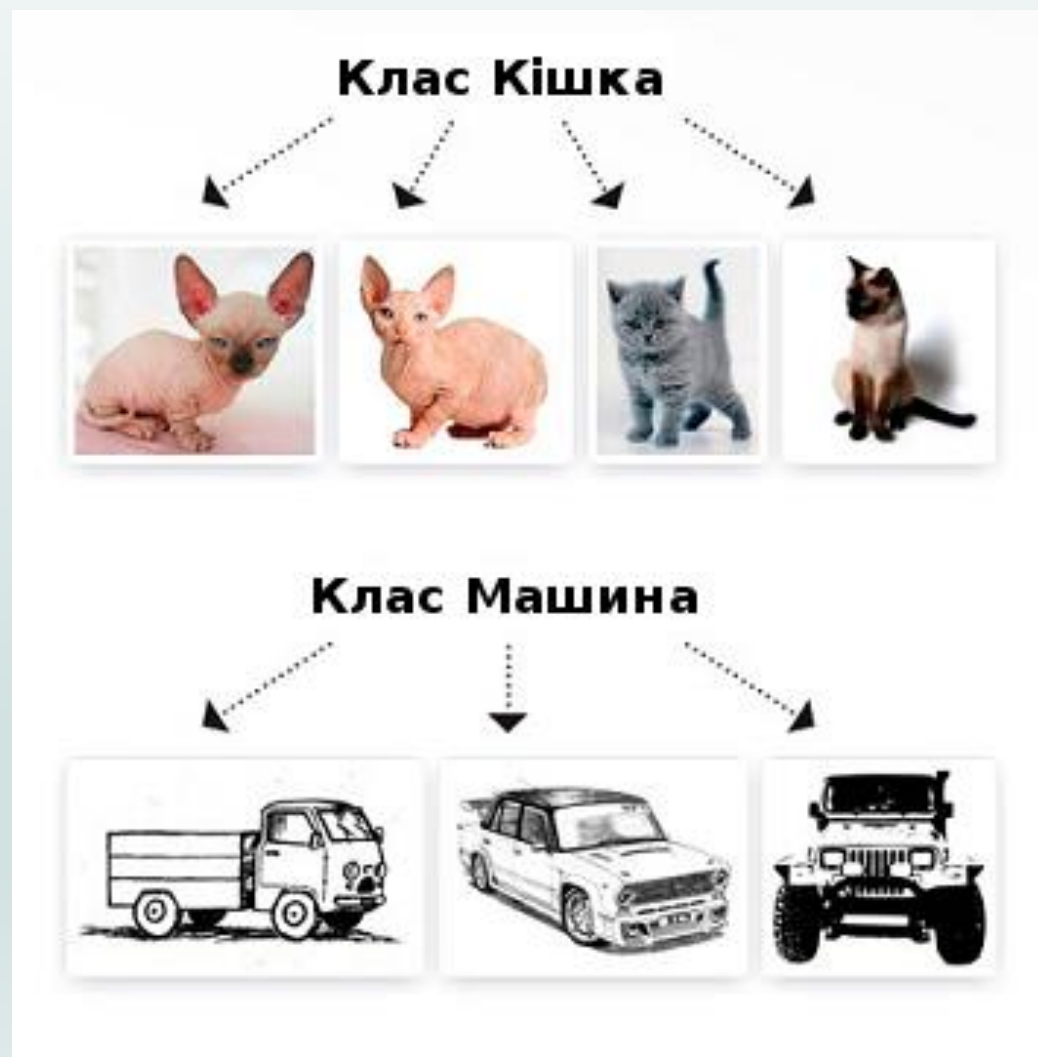
Клас в ОПП виступає ролі креслення об'єкта. Клас можна розглядати як карту будинку. Ви можете зрозуміти, як виглядає будинок, просто глянувши на його карту.

Сам собою клас не представляє нічого.

Наприклад, не можна сказати, що карта є будинком, вона тільки пояснює як справжній будинок має виглядати.

Відношення між класом та об'єктом можна уявити більш наочно, поглянувши на ставлення між машиною та Audi. Так, Audi – це машина. Проте немає такої речі, як просто машина. Машина – це абстрактна концепція, яку також реалізують у Toyota, Honda, Ferrari та інших компаніях.

# КЛАС – ОБ'ЄКТ



# Об'єктно-орієнтоване програмування(ООП)

ООП – це програмування за допомогою класів та об'єктів.

Давайте спочатку розберемося що таке об'єкт. А потім плавно перейдемо до такого поняття як клас.

Все навкруги нас являється  
**об'єктом.**

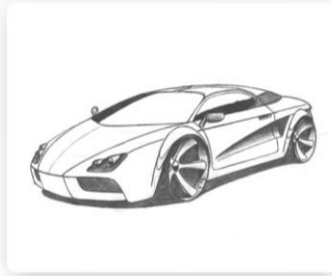


У об'єкта є  
**властивості**  
(ще називають параметри)



У об'єкта є  
**методи**  
(методи – це дії,  
тобто що може  
робити об'єкт)

Наприклад, машина – це об'єкт.



У будь-якої машини є такі  
властивості: модель, колір, розмір і  
т.п.

Методи машини:  
затормозити ();  
натиснутиНаГаз ();  
відчинитиДвері ();  
закритиДвері ();  
і т.д.

Наприклад, кішка – це об'єкт.



У будь-якої кішки є властивості:  
порода, ім'я, колір, довжина хутра, вік і т.п.

Методи кішки:  
спати();  
їсти();  
грати ();  
шкодити ();  
і т.д.

# КЛАС. З ЧОГО ПОЧАТИ СТВОРЕННЯ КЛАСУ?

Клас може бути корисний для опису фізичних або концептуальних об'єктів.

Щоб вирішити, чи варто моделювати якусь сутність за допомогою класу, задайте собі наступні питання:

- У сутності є ім'я?
- Якими властивостями сутність володіє?
- Ці властивості притаманні усім екземплярам класу? А саме:
  - Які властивості є загальними для класу загалом?
  - Які з цих властивостей є унікальними для кожного екземпляра?
- Які операції сутність виконує?

# ІМЕНА КЛАСІВ

В Python імена класів мають починатися з великої літери і не можуть починатися з цифр. Зазвичай, в якості імен використовуються іменники.

Якщо імена класів складаються з кількох слів, то вони записуються у «ВерблюжомуРегістрі».

На відміну від функцій, в іменах яких слова з'єднуються символами підкреслення, у «ВерблюжомуРегістрі» кожне слово починається з великої літери, а самі слова з'єднуються без розділювачів.

# СИНТАКСИС ВИЗНАЧЕННЯ КЛАСУ

```
class ClassName:
```

*Визначення значень*

```
    statement-value 1
```

```
    .
```

```
    statement- value n
```

*Визначення поведінки*

```
    statement- def 1
```

```
    .
```

```
    statement- def n
```

Порожній клас

```
Class Person:
```

```
    pass
```

# СТВОРЕННЯ КЛАСУ

Усі визначення класів починаються з ключового слова `class`, за яким слідує ім'я класу та двокрапка. Будь-який код, розташований з відступом нижче за визначення класу, вважається частиною тіла класу

```
class Car:
    # атрибути класе
    name = "C200"
    make = "Mercedes"
    model = 2008
    # створюємо методи класу
    def start(self):
        print ("Заводимо двигун")
    def stop(self):
        print («Виключаємо двигун»)
```

*Створено клас Car з трьома атрибутами: ім'я name, марка make та модель model. Наш клас також містить два методи: start() і stop().*

# ОБ'ЄКТИ

Клас надає креслення об'єкту.

Щоб насправді використовувати об'єкти та методи класу, потрібно створити об'єкт із цього класу .

У Python, щоб створити об'єкт класу, нам просто потрібно вписати назву класу, з наступними дужками, що відкриваються і закриваються.

```
# Створення об'єкту класу Car з назвою car_a  
car_a = Car()
```

```
# Створення об'єкту класу Car з назвою car_b  
car_b = Car()
```

# КОНСТРУКТОР КЛАСУ

**Конструктор** - це спеціальний метод, який викликається за замовчанням, коли ви створюєте об'єкт класу.

Для створення конструктора вам потрібно створити метод із ключовим словом `__init__`.

Параметр **self** є обов'язковим у визначенні методу, він повинен бути першим у списку усіх параметрів.

*При кожному виклику методу, пов'язаного з класом, автоматично передається **self** - посилання на екземпляр. Посилання надає конкретному екземпляру доступ до атрибутів і методів класу.*

# ПРИКЛАД

```
1 class Dog():
2     # Проста модель собаки.
3     def __init__(self, name, age):
4         # Ініціалізація атрибутів name і age
5         self.name = name
6         self.age = age
7     def sit(self):
8         # Собака сідає по команді
9         print(self.name.title() + " зараз сидить.")
10    def roll_over(self):
11        # Собака перекинується по команді.
12        print(self.name.title() + " перекинувся!")
```

# СТВОРЕННЯ ЕКЗЕМПЛЯРУ КЛАСУ

Можна вважати, що клас - це, свого роду, інструкція по створенню екземплярів. Відповідно, клас Dog - інструкція по створенню екземплярів, які представляють конкретних собак.

Створимо екземпляр, який представляє конкретну собаку, використовуючи клас Dog:

```
my_dog = Dog('jessie', 5)
print("My dog's name is " + my_dog.name.title() + ".")
print("My dog is " + str(my_dog.age) + " years old.")
```

# ДОСТУП ДО АТРИБУТІВ

Для звернення до атрибутів екземпляра використовується крапковий запис.

```
my_dog = Dog('jessie', 5)
```

Звернення до атрибутів

```
my_dog.name
```

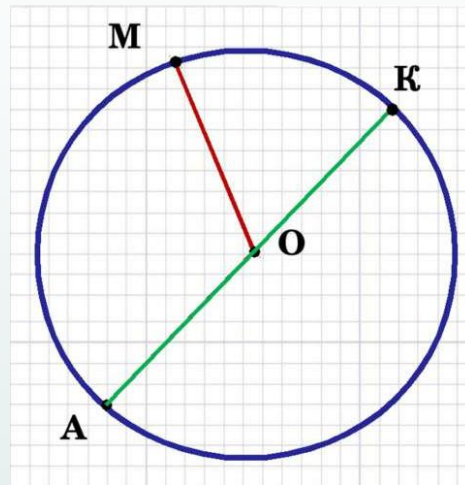
Зміна значення

```
my_dog.name="Jack"
```

Звернення до поведінки

```
my_dog.sit()
```

# ПРИКЛАД



Напишіть клас з назвою Circle для обчислення площі круга та довжину кола за введеним радіусом.

Клас Circle має містити методи, які обчислює площу круга та довжину кола

## Атрибути класу

Радіус кола  $R$

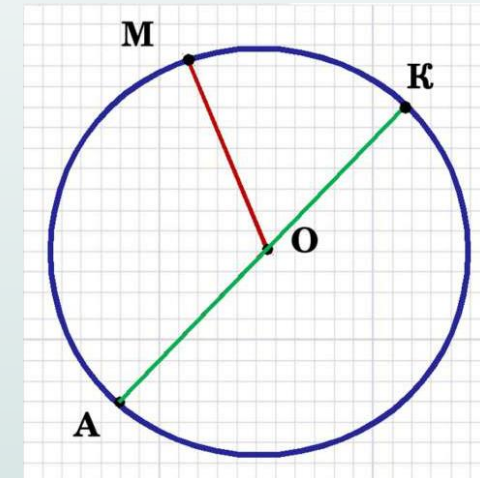
## Поведінка класу

Обчислення площі та вивід

Обчислення довжини кола та вивід

# КОД ТА РЕАЛІЗАЦІЯ

```
import math
class Circle:
    def __init__(self, radius):
        self.r=radius
    def square(self):
        S=math.pi*self.r**2
        print("Площа кола",S)
    def length(self):
        L=2*math.pi*self.r
        print("Довжина кола",L)
my_circle=Circle(10)
my_circle.square()
my_circle.length()
```



Площа кола 314.1592653589793  
Довжина кола 62.83185307179586