



# Об'єкти, їх властивості та методи.



Продовжуємо знайомство з популярною сучасною мовою програмування **Python**, яка застосовується для розв'язування різних задач: написання прикладних програм, створення ігор, розробки вебсайтів.







**Python** — інтерпретована об'єктно-орієнтована мова програмування високого рівня



### ЦІКАВІ ФАКТИ

Мова програмування **Python** була створена в 1991 році нідерландським програмістом *Гвідо ван Россумом* (рис. 21.1) і названа ним на честь скетч-серіалу «Літаючий цирк Монті Пайтона» (англ. *Monty Python's Flying Circus*).



Рис. 21.1

Мова **Python** підтримується всіма операційними системами. Існують версії для Linux, Windows. Наразі використовують дві версії Python: 2.x і 3.x.

**Мова Python** є об'єктно-орієнтованою, тобто кожна величина є об'єктом певного класу. Добре спроектовані класи – це «будівельні блоки», з яких легше будувати складні комп'ютерні програми.



**Клас** – це опис об'єктів певного типу.

**Об'єкт** – це екземпляр певного класу.

Об'єкти мають різні властивості. Значення властивостей у кожного об'єкта свої.







**Атрибут класу** – це імена змінних, у яких зберігаються значення властивостей об'єктів.

**Методи класу** – дії, які можуть виконувати об'єкти цього класу.

### Синтаксис опису класу:

```
class <назва класу>(<базовий клас>):  
    <атрибут класу> = <значення>  
    def __init__(self, <інші параметри>):  
        self.<атрибут екземпляра> = значення
```

**Методи \_\_init\_\_** – називається конструктором класу.

### Синтаксис команди створення об'єкта:

```
[змінна = ] <назва класу>([<перелік значень параметрів>])
```

Доступ до створеного об'єкта можна отримати через змінну. Якщо такий доступ не потрібен, назву змінної не вказують.

2 Опишемо клас, що моделює транспортний засіб (рис. 26.1).

```
class Transport():  
    def __init__(self, type_vehicle, motor):  
        self.type_vehicle = type_vehicle    # Тип транспортного засобу  
        self.motor = motor                  # Тип двигуна  
tr1 = Transport('вантажівка', 'дизельний') # Створення екземпляра класу
```

Рис. 26.1

В останньому рядку коду викликається конструктор класу Transport і створюється екземпляр класу, відповідні атрибути якого мають значення вантажівка і дизельний. Посилання на цей об'єкт зберігається в змінну tr1.



## Методи класу

Синтаксис заголовку методу класу:

`def <назва методу>(self, [параметри]):`

Синтаксис виклику методу для об'єкта:

`<об'єкт>.<метод>([значення параметрів])`

3

Додамо до опису класу `Transport()` опис методу `fuel()` для виведення повідомлення про вид пального (рис. 26.2):

```
class Transport():
    <...>
    def fuel(self):                # Визначення виду пального
        print(self.type_vehicle, ':", self.motor, 'двигун')
tr1 = Transport('вантажівка', 'дизельний')
tr1.fuel()
```

Рис. 26.2

Буде надруковано: вантажівка: дизельний двигун.



**Успадкування** — це можливість створення класу-нащадка на основі наявного (базового) класу.

Транспортний засіб має такі властивості: тип засобу, тип двигуна, вартість. Автобус крім цих властивостей має ще кількість місць, призначення, маршрут. З точки зору успадкування клас Автобус є нащадком класу Транспортний засіб.

5

Створимо на основі базового класу Transport два класи-нащадки: Truck і Bus, що мають власні методи (рис. 26.3).

```
transport.py x
7  class Truck(Transport):
8      def cargo(self, crg):
9          print(self.type_vehicle, ' перевозить', crg)
10 class Bus(Transport):
11     def passenger(self, destination):
12         print(self.type_vehicle, ' везе учнів', destination)
13     tr1 = Truck('вантажівка', 'дизельний')
14     tr2 = Truck('автоцистерна', 'бензиновий')
15     tr1.cargo('гравій')
16     tr2.cargo('воду')
17     tr3 = Bus('міжміський', 'дизельний')
18     tr3.passenger('на екскурсію')
```





Домашнє  
завдання

## Домашня робота

### Вправа 26



- Створити програму, що моделює облік користувачів на сайті.
- 1) Завантажте середовище PyCharm. Створіть файл типу Python file із назвою Вправа 26.
  - 2) Створіть клас із назвою User і атрибутом `privileges = []` (порожній список; буде використано в класах-нащадках). Конструктор класу має містити атрибути `first_name`, `last_name`, `age`. Опишіть методи класу User:
    - `describe_user`, який виводить повне ім'я користувача;
    - `greeting_user()` для виведення вітання для користувача;
    - `show_privileges()` для виведення списку привілеїв користувача.

```
def show_privileges(self):  
    print(self.first_name, self.last_name)  
    print(self.privileges)
```
  - 3) Напишіть клас Admin, що успадковує від класу User. У список `privileges` помістіть рядки з описом привілеїв адміністратора (Дозвіл на блокування користувачів, Дозвіл на додавання повідомлень тощо):

```
class Admin(User):  
    privileges = ['Дозвіл на блокування користувачів', 'Дозвіл на  
                додавання повідомлень']
```
  - 4) Напишіть клас Visitor, що успадковує від класу User. Задайте привілеї для користувачів цього класу (Дозвіл на перегляд вмісту, Заборона додавання повідомлень тощо).
  - 5) Створіть екземпляр класу User. Викличте для цього об'єкта методи `describe_user()`, `greeting_user()`:

```
user = User('Іванна', 'Репецька', 25)  
user.describe_user()  
user.greeting_user()
```
  - 6) Створіть екземпляри `admin` класу Admin, `visit` класу Visitor. Викличте для цих об'єктів методи `describe_user()`, `show_privileges()`. Запустіть і випробуйте програму Вправа 26.py.





## Домашня робота

---

Файл, з назвою Vprava 26, прикріпити в Human або надіслати у вигляді файлу на пошту вчителя [anton.kuropiatnickoff2016@gmail.com](mailto:anton.kuropiatnickoff2016@gmail.com)







## Домашня робота (допомога)

```
*Vprava 26.py - C:\Users\User\Desktop\Vprava 26.py (3.11.1)*
File Edit Format Run Options Window Help
1 class User():
2     privileges = []
3     def __init__(self, first_name, last_name, age):
4         self.first_name = first_name
5         self.last_name = last_name
6         self.age = age
7         print(self.first_name, ',', self.last_name, ',', self.age)
8     def describe_user(self, first_name, last_name):
9         self.describe_user()
10        print(self.first_name, self.last_name)
11        print(self.describe_user)
12    def greeting_user(self, greeting):
13        self.greeting_user()
14        print(self.greeting)
15        print(self.greeting_user)
16    def show_privileges(self):
17        print(self.first_name, self.last_name)
18        print(self.privileges)
19 class Admin(User):
20     def __init__(self, admin, privileges):
21         self.admin = Admin
22         self.privileges = ['Дозвіл на блокування користувачів', 'Дозвіл на додавання повідомлень']
23         print(self.admin, ',', self.privileges)
24 class Visitor(User):
25     def __init__(self, visitor, privileges):
26         self.visitor = Visitor
27         self.privileges = ['Дозвіл на перегляд вмісту', 'Заборона додавання повідомлень']
28         print(self.visitor, ',', self.privileges)
29 user = User('Іванна', 'Репецька', 25)

Ln: 1 Col: 0
```



До нових зустрічей!

