

Лабораторна Робота 8 – Класи у Python.

Мета роботи – вивчити та засвоїти базові навички роботи з класами у Python. Ознайомитися з поняттями конструктор, деструктор, метод класу, рівні доступу та отримати практичні навички їх застосування.

1.1 Базові поняття про класи

Все в Python є об'єктами. Це означає, що всі об'єкти в Python базуються на класі. Клас - це проект об'єкта. Давайте подивимося на прикладі, що це означає:

```
[1]: x = "Lorenzo"
print(dir(x))

['_add_', '__class__', '__contains__', '__delattr__', '__dir__', '__doc__', '__eq__', '__format__', '__ge__', '__getattribute__', '__getitem__', '__getnewargs__', '__gt__', '__hash__', '__init__', '__init_subclass__', '__iter__', '__le__', '__len__', '__lt__', '__mod__', '__mul__', '__ne__', '__new__', '__reduce__', '__reduce_ex__', '__repr__', '__rmod__', '__rmul__', '__setattr__', '__sizeof__', '__str__', '__subclasshook__', 'capitalize', 'casefold', 'center', 'count', 'encode', 'endswith', 'expandtabs', 'find', 'format', 'format_map', 'index', 'isalnum', 'isalpha', 'isascii', 'isdecimal', 'isdigit', 'isidentifier', 'islower', 'isnumeric', 'isprintable', 'isspace', 'istitle', 'isupper', 'join', 'ljust', 'lower', 'lstrip', 'maketrans', 'partition', 'replace', 'rfind', 'rindex', 'rjust', 'rpartition', 'rsplit', 'rstrip', 'split', 'splitlines', 'startswith', 'strip', 'swapcase', 'title', 'translate', 'upper', 'zfill']
```

У прикладі ми бачимо рядок, присвоєну змінної *x*. Це може виглядати як великий обсяг, але справа в тому, що у цього рядка багато методів. Якщо ви використовуєте ключове слово *dir*, ви отримаєте список всіх методів, які можна привласнити рядку. Ми бачимо 71 метод! Технічно, ми не можемо викликати методи, які починаються з підкреслення, так що це звужує список до 38 методів, але це все ще дуже багато! Що це означає? Це означає що, рядок заснований на класі, а змінна *x* - і є екземпляр цього класу. У Python ми можемо створювати власні класи. Приклад створення класу в Python:

```
[2]: # Python 2.x syntax
class Vehicle(object):
    """docstring"""

    def __init__(self):
        """Constructor"""
        pass
```

Цей клас не робить нічого конкретного, тим не менш, це дуже хороший інструмент для вивчення. Наприклад, щоб створити клас, ми використовуємо

ключове слово `class`, за яким слід найменування класу. У Python, конвенція вказує на те, що найменування класу має починатися з великої літери. Далі нам потрібно відкрити круглі дужки, за якими слід слово `object` і закриті дужки. «Object» - то, на чому заснований клас, або успадковується від нього. Це називається базовим класом або батьківським класом. Велика частина класів в Пайтон засновані на об'єкті. У класів є особливий метод, під назвою `__init__`. Цей метод викликається всякий раз, коли ви створюєте (або створюєте екземпляр) об'єкт на основі цього класу. Метод `__init__` викликається один раз, і не може бути викликаний знову всередині програми. Інше визначення методу `__init__` - це конструктор, до речі, цей термін рідко зустрічається в Python. Зверніть увагу на те, що кожен метод повинен мати як мінімум один аргумент. В Python 3 нам не потрібно прямо вказувати, що ми успадковуємо у об'єкта. Замість цього, ми можемо написати це в такий спосіб:

```
[ ]: # Python 3.x syntax

class Vehicle:
    """docstring"""

    def __init__(self):
        """Constructor"""
        pass
```

Зверніть увагу на те, що єдина різниця в тому, що круглі дужки нам більше не потрібні, коли ми формуємо наш клас на об'єкті. Розширимо визначення класу і дамо йому деякі атрибути і методи:

```
[ ]: class Vehicle(object):
    """docstring"""
    def __init__(self, color, doors, tires):
        """Constructor"""
        self.color = color
        self.doors = doors
        self.tires = tires
    def brake(self):
        """        Stop the car        """
        return "Braking"
    def drive(self):
        """        Drive the car        """
        return "I'm driving!"
```

В даному прикладі ми додали три атрибута і два методи. Ці атрибути:

```
[ ]: self.color = color  
      self.doors = doors  
      self.tires = tires
```

Атрибути описують автомобіль. У нього є колір, певну кількість дверей і коліс. Також у нього є два методи. Метод описує, що робить клас. У нашому випадку, автомобіль може рухатися і зупинятися. Можна помітити, що всі методи, включаючи перший, мають цікавий аргумент, під назвою `self`. Розглянемо його уважніше. Класу потрібен спосіб, що посилається на самого себе. Це спосіб сполучення між екземплярами. Слово `self` це спосіб опису будь-якого об'єкта, буквально.

```
[5]: class Vehicle(object):  
      """docstring"""  
      def __init__(self, color, doors, tires):  
          """Constructor"""  
          self.color = color  
          self.doors = doors  
          self.tires = tires  
      def brake(self):  
          """        Stop the car        """  
          return "Braking"  
      def drive(self):  
          """        Drive the car        """  
          return "I'm driving!"  
      if __name__ == "__main__":  
          car = Vehicle("blue", 5, 4)  
          print(car.color)  
          truck = Vehicle("red", 3, 6)  
          print(truck.color)  
  
blue  
red
```

Умови оператора `if` в даному прикладі це стандартний спосіб вказати Python на те, що ви хочете запустити код, якщо він виконується як автономний файл. Якщо ви імпортували свій модуль в інший скрипт, то код, розташований нижче перевірки `if` не запрацює. У будь-якому випадку, якщо ви запустите цей код, ви створите два примірника класу автомобіля (`Vehicle`): клас легкового і клас вантажного. Кожен екземпляр матиме свої власні атрибути і методи. Саме з цього, коли ми виводь кольору кожного примірника, вони і відрізняються один

від одного. Причина в тому, що цей клас використовує аргумент `self`, щоб вказати самому собі, що є що.

2. Варіанти завдання

Створити клас згідно з варіантом (табл. 1.1). У програмі необхідно створити декілька об'єктів визначеного класу та продемонструвати роботу кожного з методів класу.

Таблиця 1.1 – Варіанти завдань

№	Завдання
1.	Раціональна (нескоротна) дріб представляється парою цілих чисел (a, b) , де a — чисельник, b — знаменник. Створити клас Rational для роботи з раціональними дробами. Обов'язково повинні бути реалізовані операції: додавання $\text{add}, (a, b) + (c, d) = (ad + be, bd)$; вирахування $\text{sub}, (a, b) - (c, d) = (ad - be, bd)$; множення $\text{mul}, (a, b) \times (c, d) = (ac, bd)$; розподілу $\text{div}, (a, b) / (c, d) = (ad, be)$; порівняння $\text{equal}, \text{great}, \text{less}$.
2.	Створити клас Point для роботи із крапками на площині. Обов'язково повинні бути реалізовані: переміщення крапки по осі X, переміщення по осі Y, визначення відстані до початку координат, відстані між двома крапками, перетворення в полярні координати, порівняння на збіг і розбіжність.
3.	Створити клас Triangle для представлення трикутника. Поля даних повинні включати кути і сторони. Потрібно реалізувати операції: одержання і зміни полів даних, обчислення площі, обчислення периметра, обчислення висот, а також визначення виду трикутника (рівносторонній, рівнобедрений або прямокутний).
4.	Створіть клас planet , який містить інформацію про планету сонячної системи, що включає наступні атрибути: назва, діаметр, масу і відстань від Сонця в тисячах кілометрів у закритій частині класу. В класі повинна бути відкрита функція, яка повертає відстань від Сонця в милях і відкрита функцію, яка виводить усю інформацію про планету на екран.
5.	Створіть клас file , що описує файл на диску. Закритими елементами класу будуть: ім'я файлу, розмір у бітах, атрибути файлу, дата і час створення. Додайте до класу відкриту функцію, яка обчислює розмір

№	Завдання
	файлу в байтах, кілобайтах і мегабайтів, а також функцію виводу всіх даних про файл на екран.
6.	Створіть клас card , який підтримує каталог бібліотечних карток. Цей клас містить назву книги, ім'я автора, видане на руки число екземплярів, дату видачі книги читачеві і дату повернення книги в бібліотеку, у закритій частині класу. Додайте до класу відкриті функції підрахунку кількості днів, на протязі яких книга перебуває на руках читача, і виведення всіх даних про книгу на екран.
7.	Створіть клас nomenclature , що описує товари на складі магазину. Закритими елементами класу будуть: назва товару, оптова ціна, роздрібна націнка і кількість товарів на складі. Включите в клас відкриті функції підрахунку можливого чистого доходу при продажі цього товару і виводу всіх даних про товар на екран.
8.	Створіть клас persona , який містить інформацію про ПІБ людини, дату народження, стать і адресу людини у закритій частині класу. Додайте до класу відкриті функції підрахунку кількості днів, що залишилися до наступного дня народження і виводу даних про людей на екран.
9.	Створіть клас soft , який містить інформацію про встановлене програмне забезпечення. Закритими елементами класу будуть: назва програми, розроблювач, дисковий обсяг, дата завершення ліцензії. Додайте до класу відкриті функції підрахунку кількості днів до завершення ліцензії і виводу всіх даних про встановлене програмне забезпечення на екран.
10.	Створіть клас car , що містить інформацію про автомобілі. Закритими елементами класу будуть: назва моделі, виробник, кількість кінських сил, витрата палива на 100 км і масу автомобіля. Включите в клас відкриті функції підрахунку кількості палива необхідного для того, щоб проїхати відстань в 1000 км і виводу всіх даних про автомобіль.
11.	Створіть клас worker , що містить інформацію про співробітників підприємства. Закритими елементами класу будуть: ПІБ співробітника, табельний номер, назва відділу, посада і дата прийняття. Додайте до класу відкриті функції розрахунків стажу співробітника (кількість років, місяців і днів) і виводу всіх даних про співробітника на екран.

№	Завдання
12.	Створіть клас fluid , що містить інформацію про рідкі речовини. Закритими елементами класу будуть: назва речовини, колір, захід, щільність рідини. Додайте до класу відкриті функції розрахунків маси рідини в одному кубічному метрі і виводу всіх даних про рідину на екран.
13.	Створіть клас country , який містить інформацію про ім'я, форму правління, чисельності населення і площу країни в закритій частині класу. Додайте до класу відкриті функції розрахунків щільності населення країни і виводу всіх даних про країну на екран.
14.	Створити клас Bus , який містить прізвище та ініціали водія, номер автобуса, номер маршруту, марку, рік початку експлуатації, пробіг. Створити масив об'єктів. Вивести список автобусів для заданого номера маршруту і список автобусів, пробіг у яких більше 10 000 км.
15.	Реалізувати клас Cursor . Полями є координати курсору по гори зонталі і вертикалі — цілі позитивні числа, вид курсору — горизонтальний або вертикальний, розмір курсору – ціле від 1 до 15. Реалізувати методи зміни координат курсору, зміни виду курсору, зміни розміру курсору, метод гасіння і відновлення курсору.
16.	Реалізувати клас Rectangle . Полями є його сторони — позитивні числа із плаваючою крапкою. Потрібно реалізувати операції: одержання й зміни полів даних, обчислення площі, обчислення периметра, а також масштабування чотирикутника шляхом множення сторін на задане число.

3. Приклад

Варіант -16.

Файл програми

```
[6]: class Rectangle: # клас чотирикутник
    side_a = 0 # сторона a
    side_b = 0 # сторона b
    def __init__(self,a,b): # Constructor
        self.side_a = a
        self.side_b = b
    def perimeter(self): # метод обчислення периметру
        return (self.side_a+self.side_b)*2
    def square(self): # метод обчислення площі
        return self.side_a*self.side_b
    def scaling(self,x): # метод масштабування
        self.side_a*=x
        self.side_b*=x
    def display_self(self): # метод друку параметрів
        print("side a =", self.side_a, " side b =", self.side_b)

# демонстрація роботи з класом
a = Rectangle(3,5)
a.display_self()
print("Периметр = ",a.perimeter())
print("Площа = ", a.square())

a.scaling(2)
a.display_self()
print("Периметр = ",a.perimeter())
print("Площа = ", a.square())

a.side_a+=2
a.side_b+=3
a.display_self()
print("Периметр = ",a.perimeter())
print("Площа = ", a.square())
```

```
side a = 3    side b = 5
Периметр = 16
Площа = 15
side a = 6    side b = 10
Периметр = 32
Площа = 60
side a = 8    side b = 13
Периметр = 42
Площа = 104
```


4. Контрольні запитання

1. Класи у Python, їх види та призначення.
2. Які існують способи звернення до класів.
3. Опишіть процедуру створення власного класу.
4. Опишіть механізми створення і опису методів класу та подальшого звернення до цих методів.
5. Метод `_init_`, для чого створюється та як використовується? Наведіть приклади.