ЛАБОРАТОРНА РОБОТА №8.
КЛАСИ. Ч. 2

Мета роботи: ознайомитися з ООП в мові Python

Хід роботи:

Напишіть програми у середовищі програмування для розв'язування таких завдань:

1. Напишіть клас Bank для опису простих операції з вашим банківським рахунком: покласти на рахунок, зняти з рахунку, переглянути рахунок. При створенні екземпляру класу, екземпляр отримує атрибут __balance з певним значенням. Клас повинен містити методи для додавання коштів на рахунок і знімання з рахунку, за умови, що на рахунку достатньо коштів.

2. Напишіть клас Coin, який описує монету, яку можна підкидати. При створенні екземпляру класу, екземпляр отримує атрибут __sideup зі значенням heads або tails. У класі визначте метод toss, який випадково визначає результат підкидання монети - орел чи решка. Створіть екземпляр класу і виведіть на екран n підкидань монети.

3. Напишіть клас Car, який надає для створених екземплярів такі атрибути даних автомобіля: марку виготовлення автомобіля, модель автомобіля, рік автомобіля, швидкість (початкове значення 0). Клас також повинен мати наступні методи: accelerate (метод повинен щоразу додавати 5 до значення атрибуту даних про швидкість), brake (метод повинен віднімати 5 від значення атрибута даних швидкості кожного разу, коли він викликається), get_speed (метод повинен повернути поточну швидкість). Створіть екземпляр класу Car і викличте метод accelerate п'ять разів. Після кожного виклику методу accelerate отримайте поточну швидкість автомобіля і надрукуйте її значення. Потім викличте метод brake п'ять разів. Після кожного виклику методу brake отримайте поточну швидкість автомобіля та надрукуйте її значення.

4. Напишіть клас Dog, який має три атрибути класу: mammal (ссавець), nature (характер) і breed (порода), та два атрибути екземпляра: name (кличка) і age (вік). Створіть екземпляри трьох нових собак, кожна з яких різного віку. Визначте у класі Dog метод для виведення значень атрибутів екземпляру - імені та віку конкретної собаки. За потреби, додайте кілька інших методів, які визначають поведінку собаки (подавання голосу тощо). Напишіть кілька класів, які унаслідуються від батьківського класу Dog, що описують конкретні породи собак. Визначте для цих класів

атрибути nature і breed відповідно, включіть у класи по одному методу, що визначає поведінку конкретної породи собаки. Створіть батьківський клас Pets, що створює список ваших домашніх улюбленців. У підсумку, надрукуйте інформацію про ваших домашніх тварин, на зразок, як у вихідних даних.

5. Дано послідовність цілих чисел. Необхідно її обробити і вивести на екран суму першої п’ятірки чисел із цієї послідовності, потім суму другої п’ятірки, і т. д. Але послідовність не дається відразу загалом. З плином часу до вас надходять її послідовні частини. Наприклад, спочатку перші три елементи, потім наступні шість, потім наступні два і т. д. Реалізуйте клас Buffer, який буде накопичувати в собі елементи послідовності і виводити суму п’ятірок послідовних елементів у міру їх накопичення. Однією з вимог до класу є те, що він не повинен зберігати в собі більше елементів, ніж йому дійсно необхідно, тобто, він не повинен зберігати елементи, які вже увійшли в п’ятірку, для якої була виведена сума. Клас повинен мати наступний вигляд

```
class Buffer:
    def __init__(self):
        # конструктор без аргументів

    def add(self, *a):
        # додати наступну частину послідовності

    def get_current_part(self):
        # повернути збережені в поточний момент часу елементи
послідовності в порядку, в якому вони були додані
```

Зверніть увагу, що під час виконання методу add виводити суму п’ятірок може знадобитися кілька разів до тих пір, поки в буфері не залишиться менше п’яти елементів.

6. Напишіть клас-виняток, на основі вбудованого в Python класу ValueError(). Клас буде представляти перевірку певного імені на основі його довжини. Якщо довжина введеного імені є меншою 10, то має генеруватися виняток як у вихідних даних. У інших випадках нічого не виводиться.

7. Напишіть один клас для перетворення десяткового числа на число в римській системі числення. І ще один клас для перетворення числа з римської системи числення у десяткове число.

8. Онлайн-магазин.

a. Створіть клас з ім’ям Shop(). Клас Shop() повинен містити два атрибути: shop_name і store_type. Створіть метод describe_shop(), який виводить два атрибути, і метод open_shop(), який виводить повідомлення про те, що онлайн-магазин відкритий. Створіть на основі класу екземпляр з ім’ям store. Виведіть два атрибути окремо, потім викличте обидва методи.

b. Створіть три різних екземпляри класу, викличте для кожного екземпляру метод describe_shop().

c. Додайте атрибут number_of_units зі значенням за замовчуванням 0; він представляє кількість видів товару у магазині. Створіть екземпляр з ім’ям store. Виведіть значення number_of_units, а потім змініть number_of_units і виведіть знову.

d. Додайте метод з ім’ям set_number_of_units(), що дозволяє задати кількість видів товару. Викличте метод з новим числом, знову виведіть значення. Додайте метод з ім’ям increment_number_of_units(), який збільшує кількість видів товару на задану величину. Викличте цей метод.

e. Напишіть клас Discount(), що успадковує від класу Shop(). Додайте атрибут з ім’ям discount_products для зберігання списку товарів, на які встановлена знижка. Напишіть метод get_discounts_ptoducts, який виводить цей список. Створіть екземпляр store_discount і викличте цей метод.

f. Збережіть код класу Shop() у модулі. Створіть окремий файл, що імпортує клас Shop(). Створіть екземпляр all_store і викличте один з методів Shop(), щоб перевірити, що команда **import** працює правильно.

9. Облік користувачів на сайті.

a. Створіть клас з ім’ям User. Створіть два атрибути first_name і last_name, а потім ще кілька атрибутів, які зазвичай зберігаються у профілі користувача (поштова адреса, нікнейм, що відображається на сайті, згода на розсилку новин з форуму). Напишіть метод describe_user який виводить повне ім’я користувача. Створіть ще один метод greeting_user() для виведення персонального вітання для користувача. Створіть кілька примірників, які представляють різних користувачів. Викличте обидва методи для кожного користувача.

b. Додайте атрибут login_attempts у клас User. Напишіть метод increment_login_attempts(), що збільшує значення login_attempts на 1. Напишіть інший метод з ім’ям reset_login_attempts(), обнуляє значення login_attempts. Створіть екземпляр класу User і викличте increment_login_attempts() кілька разів. Виведіть значення login_attempts, щоб переконатися у тому, що значення було змінено правильно, а потім викличте reset_login_attempts(). Знову виведіть login_attempts і переконайтеся у тому, що значення обнулилося.

c. Адміністратор - користувач з повними адміністративними привілеями. Напишіть клас з ім’ям Admin, що успадковує від класу User. Додайте атрибут privileges для зберігання списку рядків виду «Allowed to add message», «Allowed to delete users», «Allowed to ban users» і т. д. Напишіть метод show_privileges() для виведення набору привілеїв адміністратора. Створіть екземпляр Admin і викличте метод.

d. Напишіть клас Privileges. Клас повинен містити всього один атрибут privileges зі списком, який треба забрати із класу Admin. Водночас, необхідно перемістити метод show_privileges() у клас Privileges із класу Admin. Створіть екземпляр priv як атрибут класу Admin. Створіть новий екземпляр admin і використайте метод для виведення списку привілеїв.

e. Збережіть клас User в одному модулі, а класи Privileges і Admin у іншому модулі. В окремому файлі створіть екземпляр admin і викличте метод show_privileges(), щоб перевірити, що все працює правильно.

Звіт по лабораторній роботі має містити:
- лістинг програми з коментарями;
- висновки по проведеній роботі.