МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ОДЕСЬКА ПОЛІТЕХНІКА»

Кафедра інформаційних технологій

Звіт

Модульна контрольна робота №1

З дисципліни: «Прототипування»

Варіант 1

Виконав:

здобувач групи АД-222

Буберенко С.О.

Перевірив:

Рудніченко М.М.

Оцінка\_\_\_\_\_\_\_\_\_\_\_\_

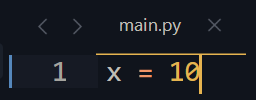
Одеса 2024

**Теоретична частина**

1. Що таке змінна і як її оголосити в Python? Наведіть приклад.
2. Які існують основні типи даних у Python?
3. Чим відрізняється список кортежу в Python?
4. Як використовувати умовні оператори if, elif, та else у Python?
5. Поясніть, що таке цикли for та while. Наведіть приклади їхнього використання.
6. Що таке функція? Як оголосити функцію в Python?
7. Чим функція відрізняється від методу?
8. Що таке аргументи та параметри функцій? Наведіть приклад.
9. Як працює оператор return у функціях?
10. Як імпортувати модуль у Python? Наведіть приклад використання вбудованого модуля.
11. Поясніть різницю між локальною та глобальною змінною.
12. Що таке lambda-функція в Python і коли її використовувати?
13. Як працювати з файлами в Python (читання та запис)? Наведіть приклад коду.
14. Поясніть, що таке винятки та як їх обробляти за допомогою try, except у Python.
15. Що таке об'єкт у контексті ОВП?
16. Що таке клас та об'єкт? Наведіть приклад створення класу та об'єкта в Python.
17. Як оголосити конструктор у класі Python і яку функцію він виконує?
18. Чим відрізняється атрибут класу від об'єкта?
19. Що таке інкапсуляція в ОВП? Як її реалізувати в Python?
20. Що таке спадкування? Наведіть приклад успадкування класів у Python.
21. Що таке поліморфізм? Наведіть приклад поліморфізму в Python.
22. Що таке абстракція у ОВП? Як її реалізувати в Python?
23. Як у Python реалізується множинне спадкування? Наведіть приклад.
24. Що таке магічні методи (наприклад, \_\_init\_\_, \_\_str\_\_, \_\_repr\_\_)? Наведіть приклад їхнього використання.
25. Що таке статичні методи в Python? Як їх оголосити та використати?
26. Що таке декоратори класів та методів у Python? Наведіть приклад.
27. Чим відрізняється метод екземпляра класу від методу класу?
28. Що таке навантаження методів (method overloading)? Наведіть приклад коду.
29. Як організувати обробку помилок у ОВП? Наведіть приклад.
30. Поясніть, як використовувати ООП для вирішення реального завдання на Пітоні. Наведіть приклад із практики.

**Відповіді**

1. Змінна — це контейнер для зберігання даних, які можуть змінюватися. Для оголошення змінної в Python достатньо вказати її ім'я та присвоїти значення:



2. Основні типи даних у Python:

- `int` (цілі числа): `x = 5`

- `float` (числа з плаваючою комою): `x = 5.5`

- `str` (рядки): `x = "Hello"`

- `bool` (логічний тип): `x = True`

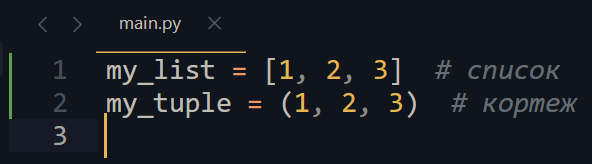
- `list` (список): `x = [1, 2, 3]`

- `tuple` (кортеж): `x = (1, 2, 3)`

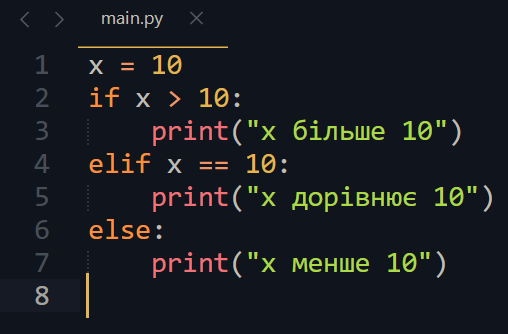
- `set` (множина): `x = {1, 2, 3}`

- `dict` (словник): `x = {"key": "value"}`

3. Список (`list`) — це змінюваний (може змінюватися після створення) тип даних, де можна додавати або видаляти елементи. Кортеж (`tuple`) — це незмінюваний тип даних.

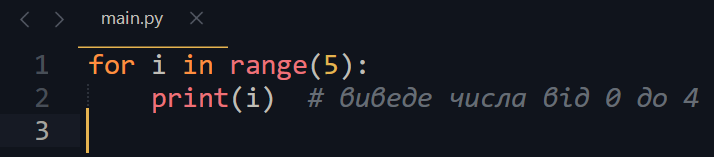


4. Умовні оператори `if`, `elif`, `else` використовуються для виконання коду в залежності від умови:

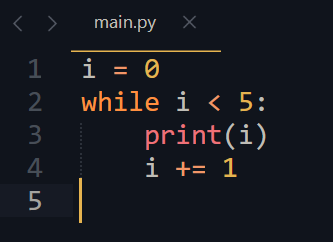


5. Цикли:

- `for` використовується для ітерації по колекціях (наприклад, списках):



- `while` виконує цикл, поки умова є істинною:



6. Функція — це блок коду, який виконує певну дію. Її можна оголосити так:

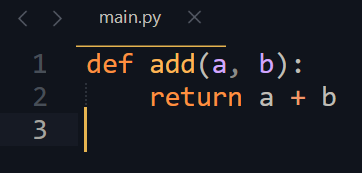


7. Функція — це окремий блок коду, який можна викликати в будь-якому контексті, тоді як метод викликається через об'єкт і зазвичай є частиною класу.

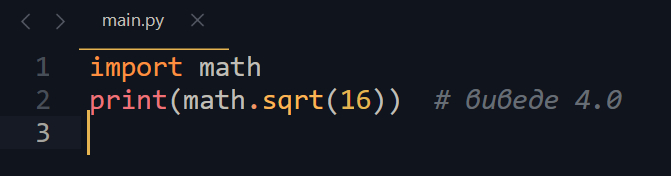
8. Параметри — це змінні, які функція приймає при виклику, а аргументи — це значення, які передаються в параметри:



9. Оператор `return` використовується для повернення значення з функції:

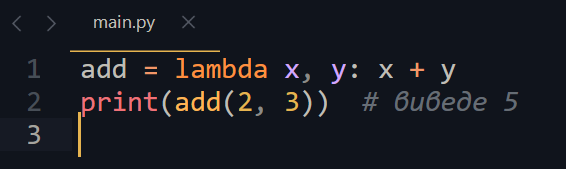


10. Для імпорту модуля використовується `import`:

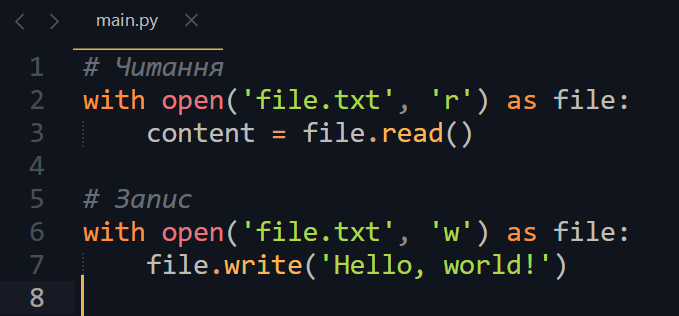


11. Локальні змінні визначені всередині функції та доступні тільки в її межах, тоді як глобальні змінні доступні в усьому коді.

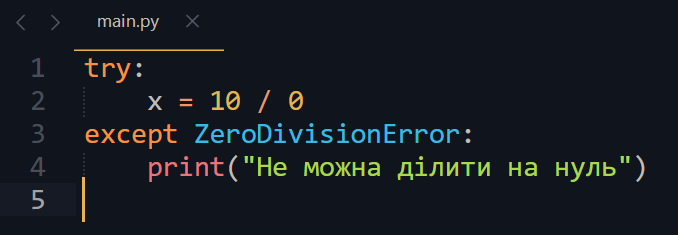
12. Lambda-функція — це анонімна функція, яка використовується для простих операцій:



13. Читання і запис файлів:

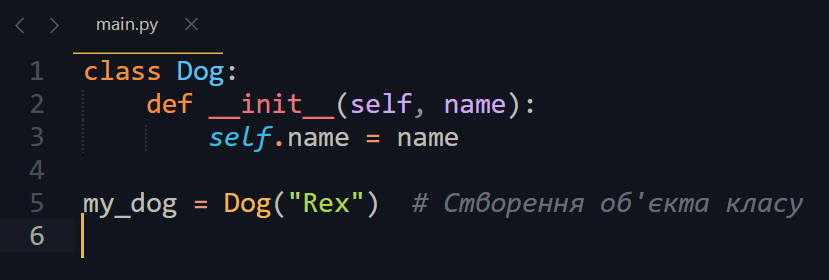


14. Винятки дозволяють обробляти помилки:

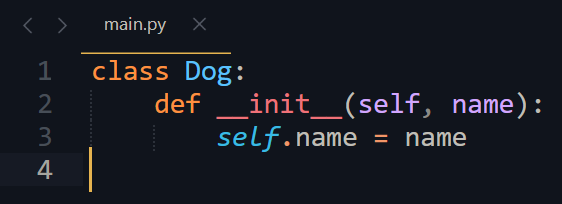


15. Об'єкт — це екземпляр класу, що поєднує дані та методи для їх обробки.

16. Клас — це шаблон для створення об'єктів:



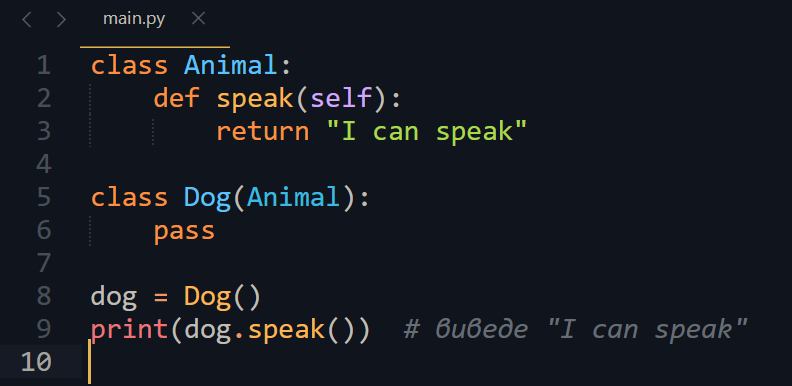
17. Конструктор — це метод `\_\_init\_\_`, який викликається при створенні об'єкта:



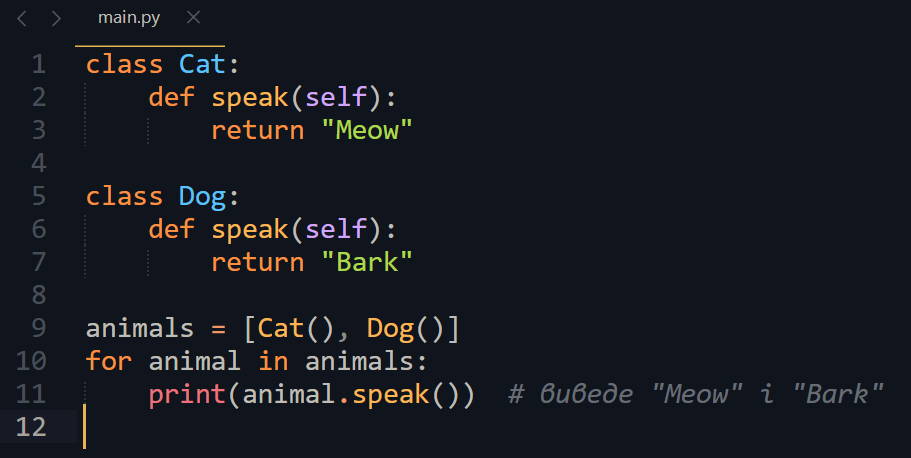
18. Атрибути класу — це змінні, спільні для всіх об'єктів класу. Атрибути об'єкта унікальні для кожного об'єкта.

19. Інкапсуляція приховує деталі реалізації об'єкта. Її реалізують через приватні змінні та методи (з префіксом `\_` або `\_\_`).

20. Спадкування дозволяє класу успадковувати властивості іншого класу:

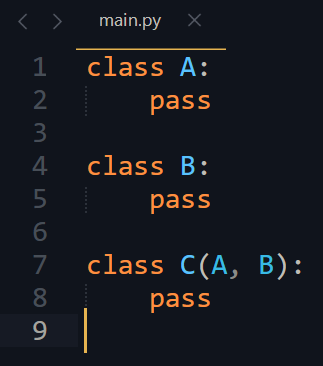


21. Поліморфізм дозволяє використовувати об'єкти різних класів однаково:

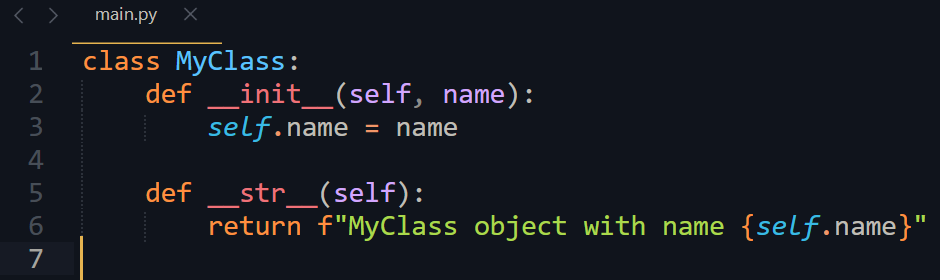


22. Абстракція приховує складні деталі реалізації. Можна реалізувати через абстрактні класи.

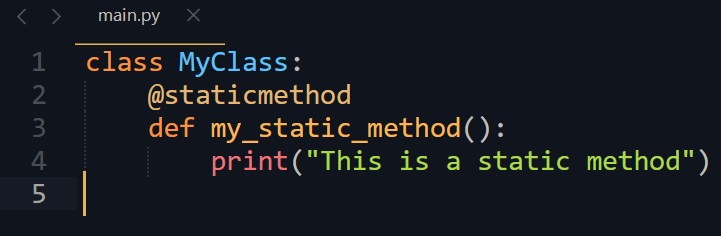
23. Множинне спадкування дозволяє класу успадковувати кілька класів:



24. Магічні методи використовуються для налаштування поведінки об'єкта:



1. Статичні методи використовують для методів, які не потребують доступу до атрибутів класу:



26. Декоратори дозволяють модифікувати методи та класи:



27. Метод екземпляра класу працює з об'єктом через `self`, тоді як метод класу працює з самим класом через `cls`.

28. Навантаження методів не підтримується безпосередньо в Python, але можна використовувати аргументи за замовчуванням або args/kwargs.

29. Для обробки помилок в ООП можна використовувати `try`, `except` або спеціальні обробники в методах.

30. Приклад ООП: створення класу для обробки платежів, де є класи `CreditCardPayment`, `PayPalPayment` зі спільним інтерфейсом `PaymentProcessor`.

**Практичная частина**

Розробити програмний додаток у вигляді консольної програми мовою Python, що є довідковою інформаційною системою обліку інтересів (хобі) учнів, що включає наступні можливості (кожен з пунктів має бути реалізований у вигляді окремої функції):

1.

• Введення особистих даних (ПІБ, стать, вік, номер телефону).

• Введення інтересів (3-5 різних інтересів/хобі, що складаються з назви хобі та текстового опису у вигляді 3х пропозицій, наприклад «Шахмати» та «Вчився з 5 років. Першу партію на змаганнях зіграв у 9 років. Відвідую шаховий клуб раз на дві тижнів за адресою: ... »). Подумайте, яку структуру даних доцільно використовувати.

• Виведення всіх введених значень у консоль.

2.

• Завдання пріоритетів введеним хобі (наприклад, за шкалою від 1 до 5).

• Сортування пріоритетів за вибраним порядком (за спаданням чи зростанням) вбудованими засобами мови та виведення даних з хобі (у відсортованому вигляді).

• Сортування чисел у номері телефону за зростанням (одним із розглянутих алгоритмів на вибір, тобто бульбашки, вставками тощо) та виведення їх у консоль.

3.

• З опису кожного хобі вивести в консоль лише першу пропозицію (наприклад, використовуючи зрізи).

• Порахувати та вивести в консоль кількість символів назв усіх хобі.

• Порахувати та вивести в консоль кількість слів в описі хобі.

• Якщо введений вік є парним числом – вивести в консоль свій рік народження; у разі, якщо введений вік є непарним числом – вивести у консоль місяць свого народження.

\* Для зручності введення даних (щоб при кожному запуску не вводити текст руками через консоль) можна використовувати прийоми роботи з файлами, наприклад з форматом \*.txt

Кожен набір даних, що вводяться, повинен бути збережений у відповідній структурі даних (список, кортеж або словник)

Виклик усіх функцій здійснювати через головну функцію main()

4. Розробити статичний (мокап) протип інтерфейсу системи

**Реалізація**



Рисунок 1. Код input\_data.py

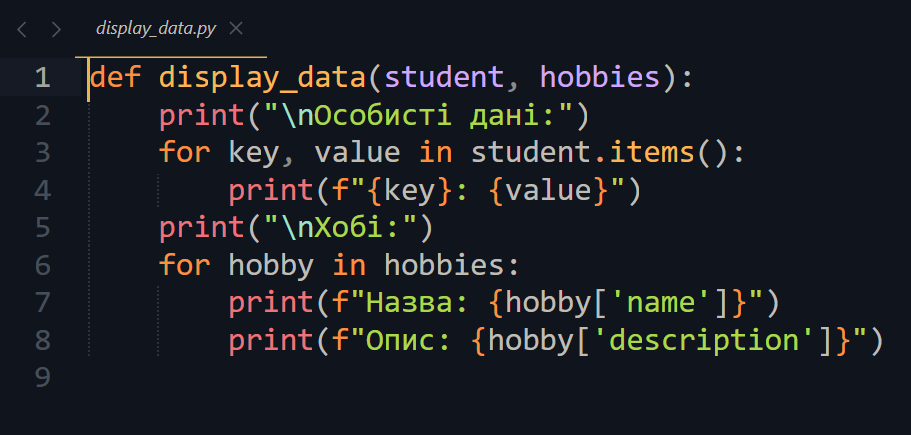


Рисунок 2. Код display\_data.py

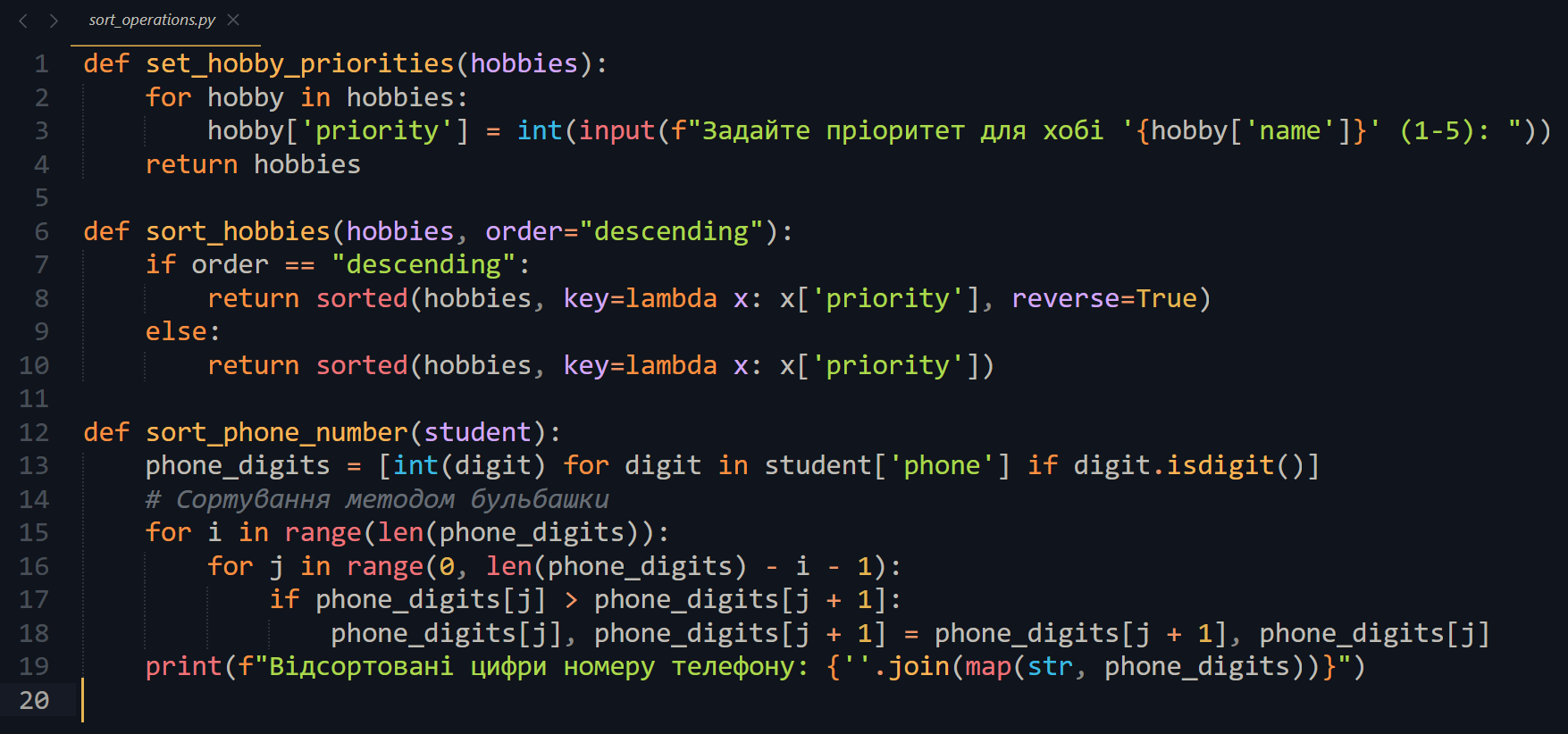


Рисунок 3. Код sort\_operations.py

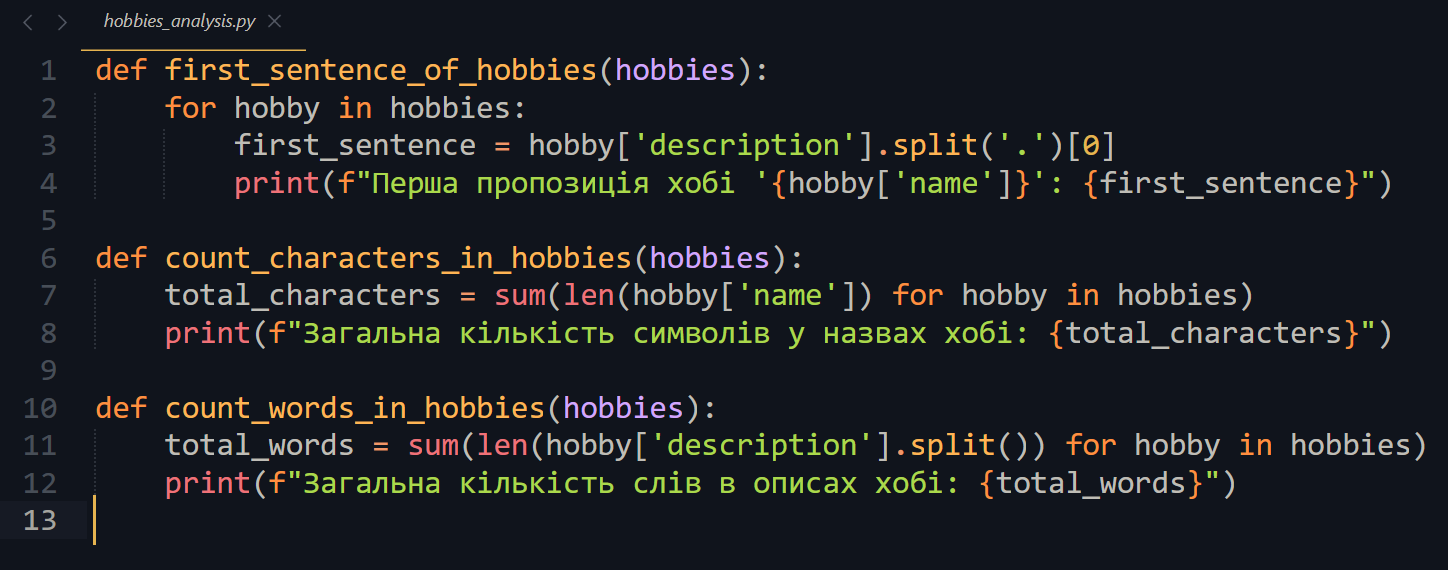


Рисунок 4. Код hobbies\_analysis.py

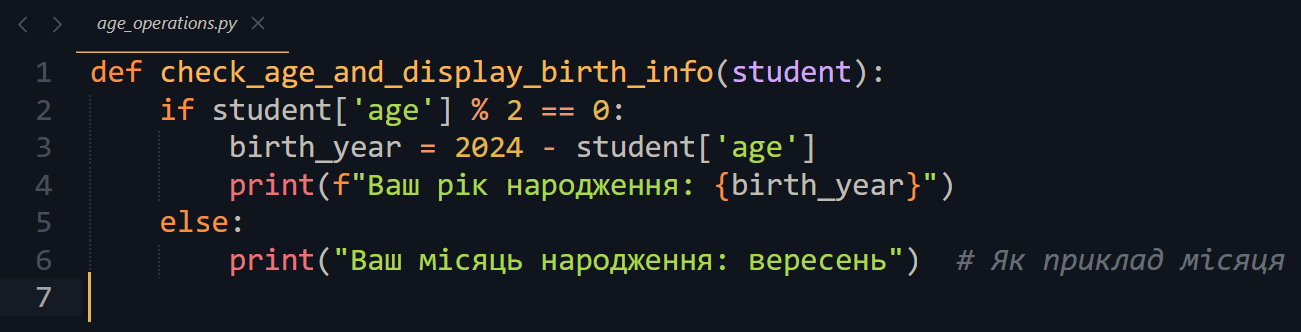


Рисунок 5. Код age\_operations.py



Рисунок 6. Код головного файлу main.py

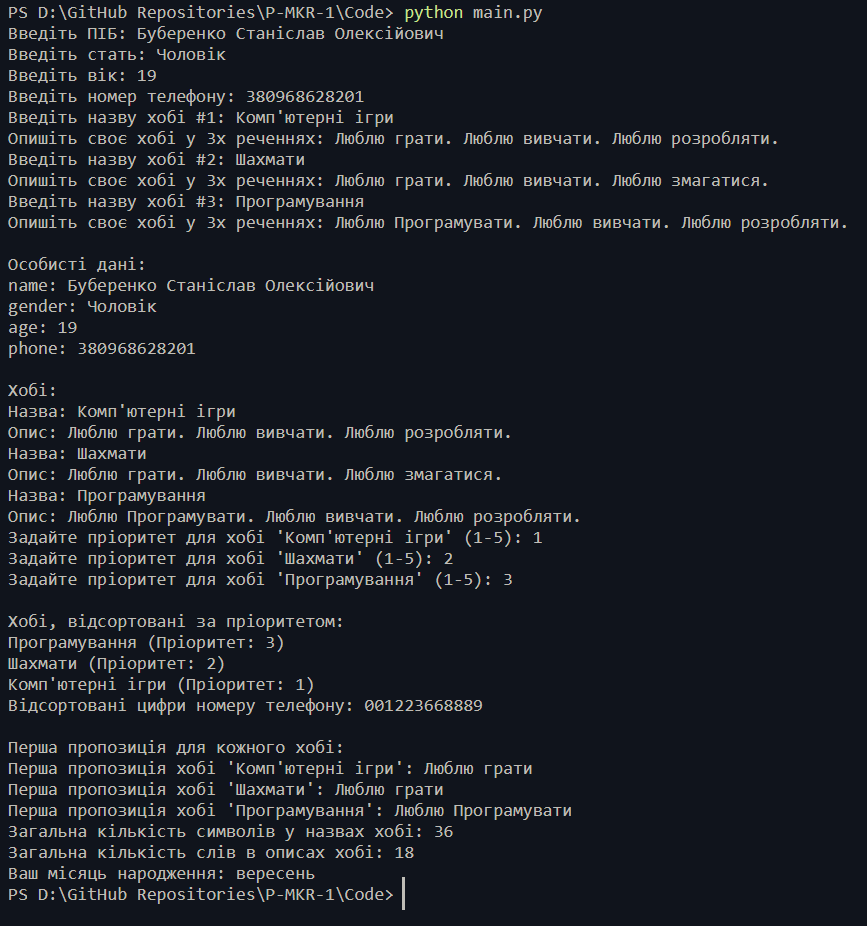


Рисунок 7. Результат