Державний торговельно-економічний університет  
Кафедра інженерії програмного забезпечення та кібербезпеки

**ЛАБОРАТОРНА РОБОТА №4**

**З ДИСЦИПЛІНИ «АЛГОРИТМИ ТА СТРУКТУРИ ДАНИХ»**

**НА ТЕМУ «МНОЖИНИ, ФУНКЦІЇ У МОВІ ПРОГРАМУВАННЯ PYTHON»**

**Виконала**: студентка факультету

інформаційних технологій

групи\_курсу 3-4

Авєріна Наталія Ігорівна

**Перевірила**: Палагута К. О.

Київ 2024

**Лабораторна робота №4**

**Тема**: Множини, функції у мові програмування Python.

**Мета**: навчитись працювати з множинами, застосовувати декоратори функцій.

**Програмне забезпечення**: PyCharm, MS Word.

**1 варіант**

**Завдання:**

Створити програму, під час виконання якої:

* створюються множини зі списками студентів групи, учасників конференції, учасників вебінару;
* за допомогою окремих функцій сформувати списки студентів групи, які:
  + прийняли участь в обох заходах,
  + прийняти участь хоча б в одному заході,
  + не приймали участь у жодному заході;
* до функцій формування списків застосувати декоратор, який виводить дані у відформатованому вигляді як нумерований список.

**Хід роботи:**

1. Створюємо декоратор print\_formatted\_list, який приймає іншу функцію як аргумент та викликає її. Результат виводиться через розділювач та нумерованим списком

def print\_formatted\_list(func):  
 def wrapper(\*args, \*\*kwargs):  
 result = func(\*args, \*\*kwargs)  
 for i, item in enumerate(result, 1):  
 print(f"{i}. {item}")  
 print("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_")  
 return wrapper

1. Визначаємо функції для роботи зі списками учасників:
   1. common\_participants приймає три множини (групу, конференцію, вебінар) і повертає спільних учасників усіх трьох заходів.
   2. any\_participants приймає ті ж множини і повертає усіх учасників хоча б одного з заходів.
   3. no\_participants приймає ті ж множини і повертає учасників, які не брали участь ні в одному з заходів.

@print\_formatted\_list  
def common\_participants(group, conference, webinar):  
 common = group.intersection(conference, webinar)  
 return common  
  
  
@print\_formatted\_list  
def any\_participants(group, conference, webinar):  
 any\_part = group.union(conference, webinar)  
 return any\_part  
  
  
@print\_formatted\_list  
def no\_participants(group, conference, webinar):  
 no\_part = group.difference(conference, webinar)  
 return no\_part

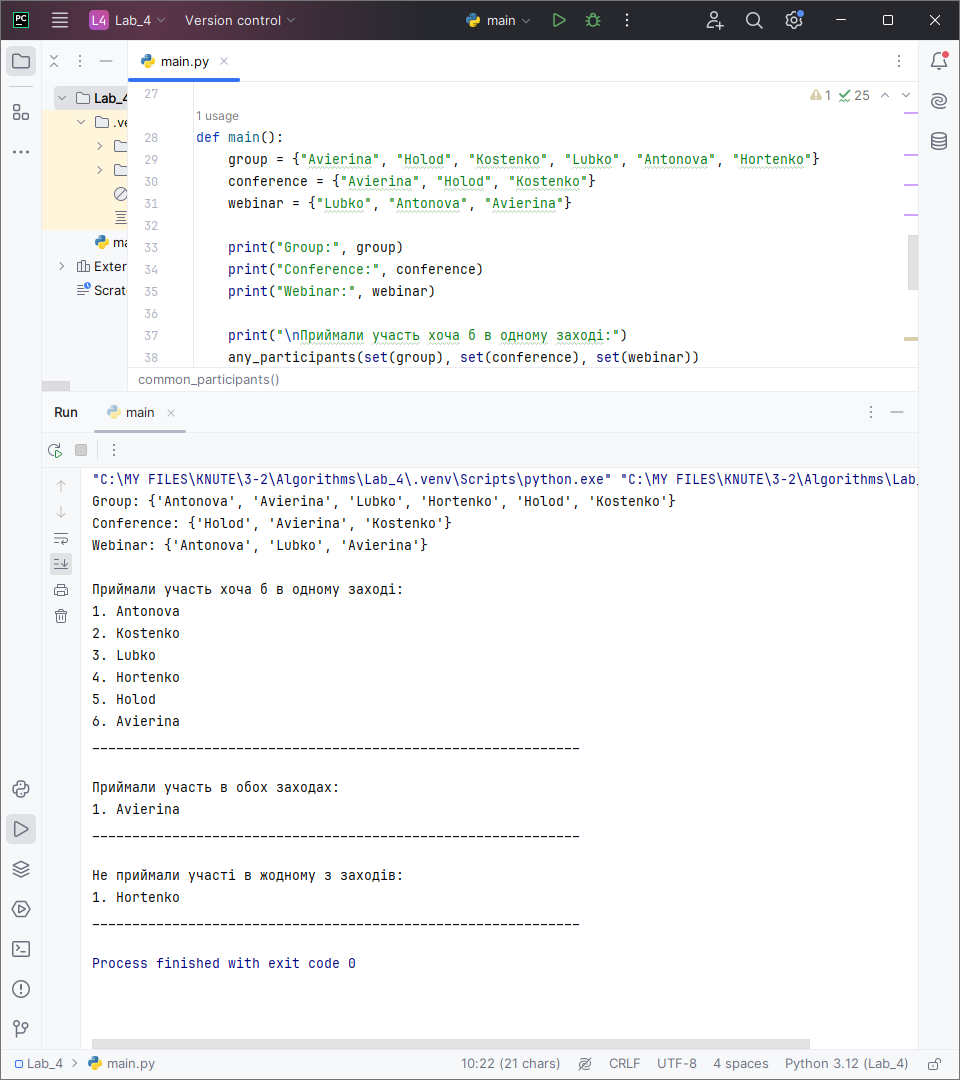
1. Основна функція main вміщає в себе визначення множин учасників групи, конференції та вебінару. Викликається кожна функція для аналізу учасників та виводяться результати у відповідному форматі.

def main():  
 group = {"Avierina", "Holod", "Kostenko", "Lubko", "Antonova", "Hortenko"}  
 conference = {"Avierina", "Holod", "Kostenko"}  
 webinar = {"Lubko", "Antonova", "Avierina"}  
  
 print("Group:", group)  
 print("Conference:", conference)  
 print("Webinar:", webinar)  
  
 print("\nПриймали участь хоча б в одному заході:")  
 any\_participants(set(group), set(conference), set(webinar))  
  
 print("\nПриймали участь в обох заходах:")  
 common\_participants(set(group), set(conference), set(webinar))  
  
 print("\nНе приймали участі в жодному з заходів:")  
 no\_participants(set(group), set(conference), set(webinar))

1. В кінці викликається функція main

if \_\_name\_\_ == '\_\_main\_\_':  
 main()

Результат виконання коду:

****

**Висновки**: в ході виконання практичної роботи було набуто практичних навичок роботи з множинами та декораторами функцій. Було створено програму, яка створювала множини зі списками студентів та формувала за допомогою методів перетину, об’єднання та різниці результат.