НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

імені Ігоря Сікорського»

Факультет прикладної математики

Кафедра прикладної математики

Звіт

із лабораторної роботи №5

із дисципліни «Основи програмування»

на тему

СТРУКТУРИ ДАНИХ: СПИСКИ, КОРТЕЖИ, МНОЖИНИ

|  |  |
| --- | --- |
| Виконала: | Керівник: |
| студентка групи КМ-93 | ст.вик. Дрозденко О. М. |
| Довгаль Є. О. |  |

Київ — 2019

Зміст

[1 ПОСТАНОВКА ЗАДАЧІ 3](#_Toc22052512)

[1.2 Завдання до лабораторної роботи 3](#_Toc22052514)

[2 ОПИС ПРОГРАМИ 4](#_Toc22052515)

[2.1 Змінні та функції 4](#_Toc22052516)

[Додаток А 5](#_Toc22052517)

[Текст програми 5](#_Toc22052518)

[Додаток Б 6](#_Toc22052519)

[Скріншоти тестувань програми 6](#_Toc22052520)

[Додаток В 7](#_Toc22052521)

[Відповіді на питання для самоперевірки 7](#_Toc22052522)

# ПОСТАНОВКА ЗАДАЧІ

## Мета роботи

Вивчення структур даних (list, tuple, set), створюваних користувачем.

## 1.2 Завдання до лабораторної роботи

1) Вивчити принципи створення списків і множин у мові Python.

2) Розробити програми відповідно до варіанта завдання.

3) Вхідні дані і результат роботи супроводжувати відповідною інформацією на екрані.

Завдання для лабораторної роботи складається з 2-х частин:  
- робота зі списками;  
- робота з множинами.

# 2 ОПИС ПРОГРАМИ

## Змінні та функції

### Завдання 1:

*Створити два списки, однакових за кількістю слів. Необхідно:*

*- порівняти слова на однакових позиціях і видалити більше;*

*- якщо слова однакові по довжині, видалити обидва;*

*- вивести на екран обидва модифікованих списку.*

Пояснення:

Маємо ф-цію task1(list1, list2), яка у якості параметрів очікує списки.

Якщо довжина списків не однакова – помилка (за умовою).

Далі створюються два допоміжні списки – rem1 та rem2. Вони будуть заповнюватися індексами тих елементів, які необхідно видалити.

Починається головне: цикл з заповнюванням rem1, rem2. Якщо довжина i-того елемента першого списка більша за другий – записуємо індекс і в rem1 через insert(0, i). Таким чином індекси будуть добавлятися не в кінець, а в початок. Це для того, щоб потім не було проблем з видаленням і не потрібно було для вирішення цього попередньо використовувати reverse(). Якщо другий елемент більший за перший – записуємо в rem2. Якщо елементи однакової довжини – записуємо обидва.

Після циклу з пошуком індексів на видалення, переходимо безпосередньо до самого видалення. Це в одну строку через list1.pop(rem1[i]). Через pop(rem1[i]) видаляємо елементи з індексами, які знаходяться у списку rem1. Аналогічно з другим списком.

Завдання виконано. Для перевірки інших елементів користувач повинен самостійно створити список у програмі.

### Завдання 2:

*Знайти:*

*U = {1,2,3,4,a,b,c,d,ee,tt,ww}, A = {1,2,ee,a,d} B = {d,ee,4,3}*

Пояснення:

Створюється дві множини *A = {1,2,ee,a,d}* та *B = {d,ee,4,3}*.

Python має скорочення для операцій з множинами. Відповідь виглядяє так: ((A ^ B) & (A | B)).

Для перевірки:

1. A+B = {1,2,3,4,a}
2. AB = {1,2,ee,a,d,4,3}
3. (A+B) (AB) = {1,2,3,4,a}

# Додаток А

## Текст програми

Зовнішній вигляд збережено для більш зручного читання

def task1(list1, list2):  
 print('\n<<< Виконання першого завдання >>>')  
  
 if len(list1) != len(list2):  
 print('Кількість елементів двох списків не співпадає :(')  
 return -1  
  
 rem1, rem2 = [], []  
 for i in range(len(list1)):  
 if len(list1[i]) > len(list2[i]):  
 rem1.insert(0, i)  
 elif len(list1[i]) < len(list2[i]):  
 rem2.insert(0, i)  
 else:  
 rem1.insert(0, i)  
 rem2.insert(0, i)  
  
 for i in range(len(rem1)): list1.pop(rem1[i])  
 for i in range(len(rem2)): list2.pop(rem2[i])  
  
 print(f'1: {list1}\n2: {list2}')  
  
  
def task2():  
 print('\n<<< Виконання другого завдання >>>')  
 A = {1, '2', 'ee', 'a', 'd'}  
 B = {'d', 'ee', 4, 3}  
 print(f'((A+B)∩(A∪B)) = {((A ^ B) & (A | B))}')  
  
  
# Списки для задвання 1:  
first\_list = ['hello', 'hello', 'WorlD@', 'd', 'EVAEVA', 'nul1l', 'TEST!!!!']  
second\_list = ['d', 'ee', 'adssadsa', 'ssssss', 'EVAEVA', '2', 'TEST!!!!']  
# Запуск завдань:  
task1(first\_list, second\_list)  
task2()

# Додаток Б

## Скріншоти тестувань програми

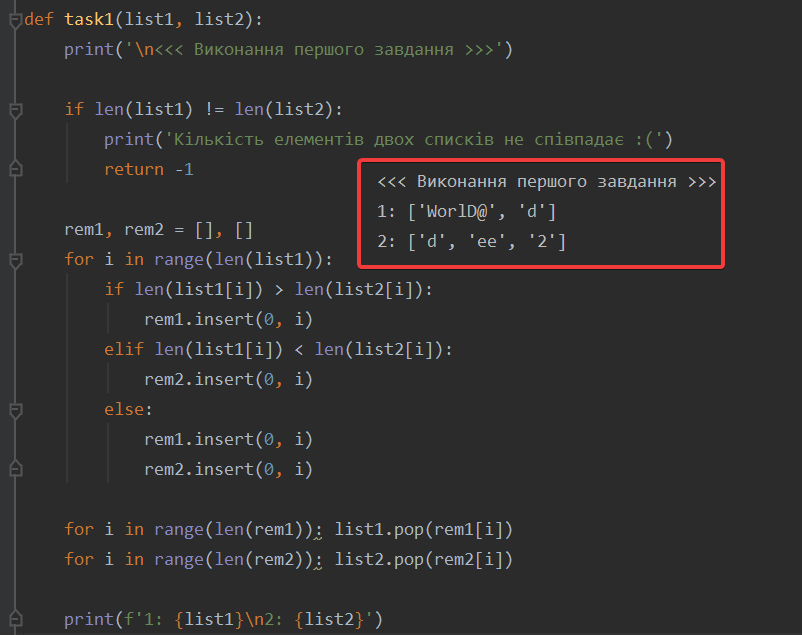


Рисунок 3.1 – Виконання завдання 1

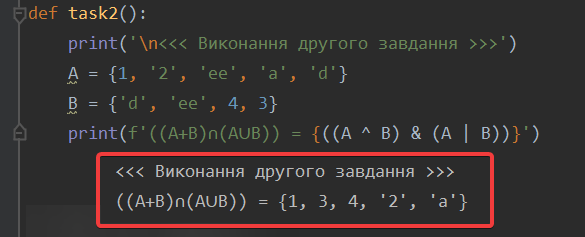


Рисунок 3.2 – Виконання завдання 2

# Додаток В

## Відповіді на питання для самоперевірки

1. Що таке список, кортеж, множина.

Список – це список впорядкованих елементів, розділених комами, і укладених в квадратні дужки. Дуже схожий на масив в інших мовах програмування. Є змінним типом даних. Елементи можуть бути різні за типами даних.

Кортеж – майже те ж саме, що й список, але його неможливо редагувати. Записується у круглих дужках.

Множина – також схоже на список, але елементи є унікальними і невпорядковані. Записується у фігурних дужках. Можна виконувати ті ж самі операції, як з множинами у математиці (перетин, доповнення тощо).

1. Перерахуйте відмінності списку від кортежу.

Не можна редагувати. Записується у кв. дужках.

1. Наведіть приклади створення списку, кортежу, множини.

Names = [‘Eva’, ‘Jhon’, ‘Maria’, ‘Eva’]

Numbers = (1, 7, 13, 13, 13)

A = {‘a’, ‘b’, 2, 4, ‘6’}

1. Перерахуйте методи зміни списку.

append(), insert(), extend(), pop(), remove(), reverse(), sort(), clear()

5.Перерахуйте функції вищого порядку для роботи з послідовностями.

mар(), zip(), filter(), reduce(), lambda()

6. Що повертає функція *zip*.

На кожній ітерації повертає кортеж, що містить елементи послідовностей, які розташовані на однаковому зміщенні:

a=[1,2] b=[3,4] list(zip(a,b)) => [(1,3),(2,4)]

7. Як кортеж перетворити у список.

tuple1={1,2,3} 🡪 tuple1=list(tuple1) 🡪 type(tuple1) => list

8. Скільки разів елемент може входити до множини.

Один.

9. Чи може список бути елементом множини.

Може.

10. Як можна із множини зробити список.

set1=(1,2,3) 🡪 set1=list(set1) 🡪 type(set1) => list

11. Чи можна у Python виконувати звичайні для математики операції над множинами.

Можна (якщо мова про додавання-віднімання).