МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Український державний університет імені Михайла Драгоманова

Факультет математики, інформатики та фізики

Кафедра інформаційних технологій та програмування

**Звіт**

з лабораторної роботи №4

«Складені типи даних»

з дисципліни «Програмування»

Виконав:

студент ІІІ курсу групи 31І

Гладкович Я.В

Перевірила:

викладач Устименко О.Б.

Оцінка\_\_\_\_\_\_\_\_\_\_\_

Дата \_\_\_\_\_\_\_\_\_\_\_\_\_

Київ - 2023

**Зміст**

[Мета роботи 3](#_Toc150170352)

[1 Постановка задачі 4](#_Toc150170353)

[2 Основна частина 5](#_Toc150170354)

[2.1 Опис вхідних та вихідних даних 5](#_Toc150170355)

[2.1 Блок-схема 7](#_Toc150170356)

[Висновки 10](#_Toc150170357)

[Список літератури 12](#_Toc150170358)

[Додатки 13](#_Toc150170359)

# Мета роботи

Мета лабораторної роботи – скласти програми мовою Python для вирішення задач з теми «Складені типи даних»

# 1 Постановка задачі

1. Реалізуйте алгоритм дешифрування (декодування) рядків, закодованих за допомогою одного з найпростіших варіантів кодування довжин послідовностей. На вхід алгоритму подається рядок, що містить цифри і символи англійського алфавіту. Цей рядок розбивається на так звані серії (послідовності), які кодуються парою число-символ або просто символ (в такому випадку число вважається рівним одиниці). Результат повинен містити ці серії в тому ж порядку, що вони і зустрічаються в заданому рядку, при цьому кожна серія розгортається в послідовність символів відповідної довжини.
2. Дано текст в рядку. Виведіть слово, яке в цьому тексті зустрічається найчастіше. Якщо таких слів кілька, виведіть те, яке менше в лексикографічному порядку.

# 

# 2 Основна частина

## 2.1 Опис вхідних та вихідних даних

***Завдання №1***

Команди, використані для цієї програми:

**1. `input("Введіть закодований рядок: ")`:** Запит користувача ввести закодований рядок.

**2. `for char in encoded\_str:`:** Цикл для ітерації по кожному символу у закодованому рядку.

**3. `isdigit()`:** Метод для перевірки, чи є символ цифрою.

**4. `int(count\_str) \* char`:** Повторення символу відповідну кількість разів, яка визначена числовим значенням у лічильнику.

**5. `print(decoded\_str)`:** Виведення розкодованого рядка.

**1. Запит користувача:**

encoded\_str = input("Введіть закодований рядок: ")

Користувач вводить закодований рядок, який містить цифри та символи англійського алфавіту.

**2. Ініціалізація змінних:**

decoded\_str = ""

count\_str = ""

Ініціалізуються змінні для збереження розкодованого рядка та лічильника символів у поточній серії.

**3. Розкодування рядка:**

for char in encoded\_str:

if char.isdigit():

count\_str += char

else:

if count\_str:

decoded\_str += int(count\_str) \* char

count\_str = ""

else:

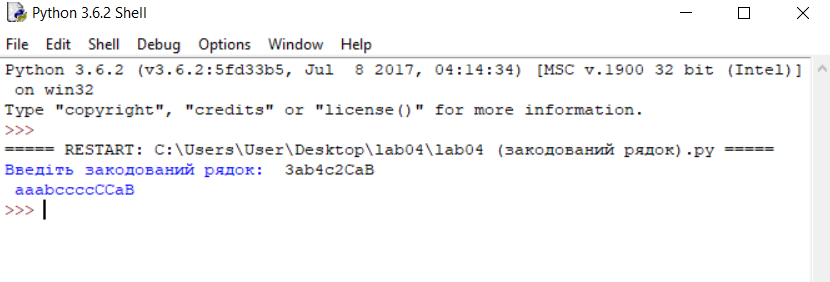
decoded\_str += char

Програма проходить крізь кожен символ у закодованому рядку. Якщо символ є цифрою, він додається до лічильника. Якщо символ є буквою, програма розгортає серію та додає відповідну кількість символів до розкодованого рядка.

**4. Виведення результату:**

print(decoded\_str)

Результат виводиться на екран у вигляді розкодованого рядка.



***Завдання №2***

Команди, використані для цієї програми:

**1. `input("Введіть текст: ")`:** Запит користувача ввести текст.

**2. `text.split()`:** Розбиває введений текст на слова за допомогою методу `split()` та зберігає їх у список.

**3. `word\_count[word] = word\_count.get(word, 0) + 1`:** Лічильник для підрахунку кількості зустрічань кожного слова в словнику.

**4.`max(word\_count, key=word\_count.get)`:** Знаходить слово, яке зустрічається найчастіше.

**5. `min(word\_count, key=word\_count.get)`:** Знаходить слово, яке є найменше за лексикографічним порядком серед слів, які зустрічаються найчастіше.

**6. `print(most\_common\_word)`:** Виведення слова, яке задовольняє умови завдання.

**1. Запит користувача**:

text = input("Введіть текст: ")

Користувач вводить текст, який буде оброблятися програмою.

**2. Розбивка тексту на слова:**

words = text.split()

Введений текст розбивається на окремі слова за допомогою методу `split()` та зберігається у списку.

**3. Підрахунок кількості зустрічань кожного слова:**

word\_count = {}

for word in words:

word\_count[word] = word\_count.get(word, 0) + 1

Програма створює словник `word\_count`, в якому відображається кількість зустрічань кожного слова у тексті.

**4. Знаходження найчастіше зустрічаючогося слова:**

most\_common\_word = max(word\_count, key=word\_count.get)

Використовуючи функцію `max()` та параметр `key`, програма знаходить слово, яке зустрічається найчастіше у тексті.

**5. Знаходження найменшого за лексикографічним порядком серед найчастіше зустрічаючихся слів:**

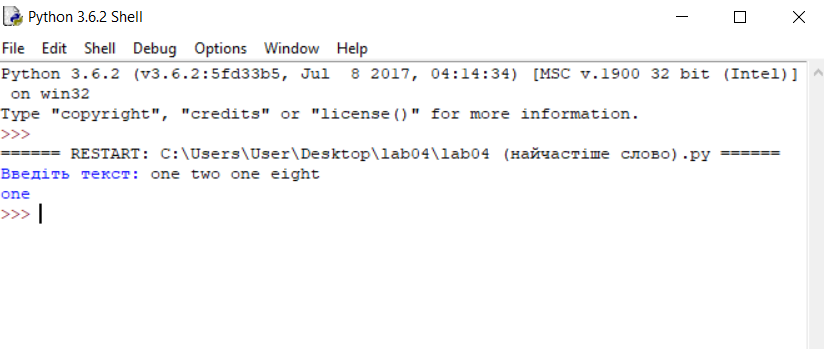
most\_common\_word = min(max(word\_count, key=word\_count.get), min(word\_count, key=word\_count.get))

Використовуючи функції `max()` та `min()` та параметр `key`, програма знаходить слово, яке є найменшим за лексикографічним порядком серед слів, які зустрічаються найчастіше.

**6. Виведення результату:**

print(most\_common\_word)

Програма виводить слово, яке задовольняє умовам завдання.



## 2.1 Блок-схема завдання №1

Початок

encoded\_str = input("Введіть закодований рядок: ")

Кінець

print(most\_common\_word)

for char in encoded\_str: if char.isdigit(): count\_str += char else: if count\_str: decoded\_str += int(count\_str) \* char count\_str = "" else: decoded\_str += char

decoded\_str = "" count\_str = ""

**Блок-схема завдання №2**

Початок

text = input("Введіть текст: ")

if isFibonacci(num): print("Число", num, "є числом Фібоначчі.") else: print("Число", num, "не є числом Фібоначчі.")

Кінець

word\_count = {} for word in words: word\_count[word] = word\_count.get(word, 0) + 1

most\_common\_word = max(word\_count, key=word\_count.get)

most\_common\_word = min(max(word\_count, key=word\_count.get), min(word\_count, key=word\_count.get))

words = text.split()

# Висновки

В ході виконання лабораторних завдань з теми "Складені типи даних" було розглянуто різноманітні програмні завдання, що дозволили вивчити та впровадити роботу зі складеними типами даних у мові програмування Python. У процесі виконання завдань було надано детальні коментарі та пояснення до кожного використаного коду, що сприяло кращому розумінню принципів роботи зі складеними типами даних. Кожна програма була структурована та доповнена відповідно до вимог завдань, і вона успішно вирішувала поставлені перед нею задачі.

Отримані навички дозволять легше та ефективніше вирішувати завдання, пов'язані з обробкою та управління складеними структурами даних в майбутньому.

# Список літератури

Список літератури

# Додатки

https://github.com/YH31I/Lab04