

Міністерство освіти і науки України  
Національний університет „Львівська політехніка”  
Кафедра ЕОМ



## **Звіт**

до лабораторної роботи № 8

з дисципліни: «Кросплатформні засоби програмування»

На тему: «Файли та виключення у Python»

Виконав:  
Студент групи КІ-306  
Баран В. М.  
Прийняв:  
Іванов Ю. С.

**Львів 2023**

**Мета роботи:** оволодіти навиками використання засобів мови Python для роботи з файлами.

## ЗАВДАННЯ

1. Написати та налагодити програму на мові Python згідно варіанту. Програма має задовольняти наступним вимогам:

- програма має розміщуватися в окремому модулі;
  - програма має реалізувати функції читання/запису файлів у текстовому і двійковому форматах результатами обчислення виразів згідно варіанту;
  - програма має містити коментарі.
2. Завантажити код на GitHub згідно методичних вказівок по роботі з GitHub.
3. Скласти звіт про виконану роботу з приведенням тексту програми, результату її виконання та фрагменту згенерованої документації та завантажити його у ВНС.
4. Дати відповідь на контрольні запитання.

### Варіант завдання:

1.  $y = \text{tg}(x)$

### Код програми:

```
import os
import math
import struct

# Функція для запису результату у текстовому форматі
def writeResTxt(fName, result):
    with open(fName, 'w') as f:
        f.write(str(result))

# Функція для читання результату з текстового файлу
def readResTxt(fName):
    result = 0.0
    try:
        if os.path.exists(fName):
            with open(fName, 'r') as f:
                result = float(f.read())
        else:
            raise FileNotFoundError(f"File {fName} not found.")
    except FileNotFoundError as e:
        print(e)
    return result

# Функція для запису результату у двійковому форматі
def writeResBin(fName, result):
    with open(fName, 'wb') as f:
        # Використовуємо struct.pack для збереження числа у двійковому форматі
        f.write(struct.pack('f', result))

# Функція для читання результату з двійкового файлу
```

```

def readResBin(fName):
    result = 0.0
    try:
        if os.path.exists(fName):
            with open(fName, 'rb') as f:
                # Використовуємо struct.unpack для розпакування числа з двійкового
                # формату
                result = struct.unpack('f', f.read())[0]
        else:
            raise FileNotFoundError(f"File {fName} not found.")
    except FileNotFoundError as e:
        print(e)
    return result

# Функція для обчислення y=tan(x)
def calculate_tan(x):
    return math.tan(x)

# Головна частина програми
if __name__ == "__main__":
    # Зчитуємо вхідні дані від користувача (в радіанах)
    x = float(input("Enter value of x (in radians): "))

    try:
        # Обчислюємо результат
        result = calculate_tan(x)

        # Виводимо результат у консоль
        print(f"Result of tan(x) is: {result}")

        # Записуємо результат у текстовий файл
        writeResTxt("textRes.txt", result)

        # Записуємо результат у двійковий файл
        writeResBin("binRes.bin", result)

        # Читаємо результат із двійкового та текстового файлів та виводимо їх
        print(f"Result from binary file: {readResBin('binRes.bin')}")
        print(f"Result from text file: {readResTxt('textRes.txt')}")

    except Exception as e:
        # Виводимо повідомлення про будь-яку помилку
        print(f"An error occurred: {e}")

```

## Результати роботи програми:

```

Enter value of x (in radians): 50
Result of tan(x) is: -0.27190061199763077
Result from binary file: -0.27190062403678894
Result from text file: -0.27190061199763077

Process finished with exit code 0

```

## Відповіді на контрольні запитання

1. За допомогою якої конструкції у мові Python обробляються виключні ситуації? - "try-except".
2. Особливості роботи блоку except?  
- Блок "except" використовується для обробки виключних ситуацій, які можуть виникнути у блоку "try". Він містить код, який виконується у випадку виникнення виключної ситуації.
3. Яка функція використовується для відкривання файлів у Python? - open()
4. Особливості використання функції open?  
- Функція "open()" дозволяє відкривати файли з різними параметрами, такими як режим відкриття, кодування і т.д.
5. В яких режимах можна відкрити файл?  
- у режимах читання ("r"), запису ("w"), додавання ("a") і бінарного режиму ("b"), а також їх комбінаціях.
6. Як здійснити читання і запис файлу?  
- Для читання файлу використовують функцію "read()", а для запису - "write()".
7. Особливості функцій у мові Python?  
- Функції у Python можуть мати параметри, повертати значення, і багато інших особливостей, що дозволяють створювати різноманітні функції для розв'язання завдань.
8. Для чого призначений оператор with?  
- Оператор "with" використовується для автоматичного відкриття і закриття файлів (або інших ресурсів) та забезпечення правильного їхнього використання.
9. Які вимоги ставляться до об'єктів, що передаються під контроль оператору with?  
- Об'єкти, що передаються під контроль оператору "with", повинні мати методи "enter" і "exit", які виконуються перед входом і виходом з контексту.
10. Як поєднуються обробка виключних ситуацій і оператор with?  
- Обробка виключних ситуацій і оператор "with" можуть комбінуватися, дозволяючи обробляти виключення у контексті "with" і гарантуючи закриття ресурсів навіть у випадку виникнення виключної ситуації.

## Висновок

У ході виконання даної лабораторної роботи, я успішно оволоділа необхідними навичками. Основні засоби мови Python для роботи з файлами були детально вивчені, включаючи відкриття, читання, запис та закриття файлів.