

Міністерство освіти і науки України
Національний університет «Львівська політехніка»
Кафедра «Електронних обчислювальних машин»



Звіт
з лабораторної роботи № 8
з дисципліни: «Кросплатформенні засоби програмування»
на тему: «Файли та виключення у Python»

Виконав:

студент групи КІ-306

Щирба Д.В.

Перевірив:

доцент кафедри ЕОМ

Іванов Ю. С.

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

Завдання (варіант № 28): $y = 1 / \text{ctg}(2x)$

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.

Вихідний код програми:

Файл Files_utils.py

```
import os
import struct

def write_res_txt(file_name, result):
    with open(file_name, 'w') as f:
        f.write(str(result))

def read_res_txt(file_name):
    result = 0.0
    try:
        if os.path.exists(file_name):
            with open(file_name, 'r') as f:
                result = float(f.read())
        else:
            raise FileNotFoundError(f"File {file_name} not found.")
    except FileNotFoundError as e:
        print(e)
    return result

def write_res_bin(file_name, result):
    with open(file_name, 'wb') as f:
        f.write(struct.pack('d', result))

def read_res_bin(file_name):
    result = 0.0
    try:
        if os.path.exists(file_name):
            with open(file_name, 'rb') as f:
                result = struct.unpack('d', f.read())[0]
        else:
            raise FileNotFoundError(f"File {file_name} not found.")
    except FileNotFoundError as e:
        print(e)
    return result
```

Файл calc.py

```
import math

def calculate(x):
    if x == 45 or x == -45 or x == 90 or x == -90 or x == 360 or x==0:
        return None
    else:
        return math.tan(2 * x * math.pi / 180)
```

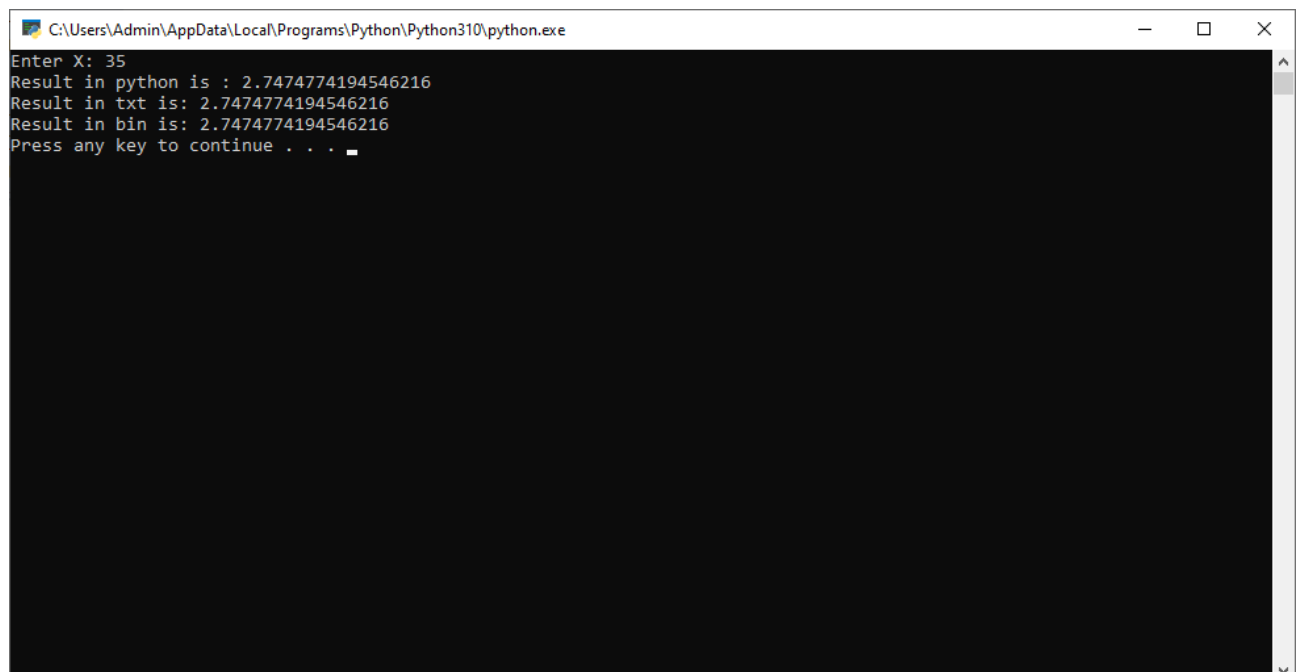
Файл Lab8_Python.py

```
import sys
import calc
import Files_utils

if __name__ == "__main__":
    data = float(input("Enter X: "))
    result = calc.calculate(data)
    print(f"Result in python is : {result}")

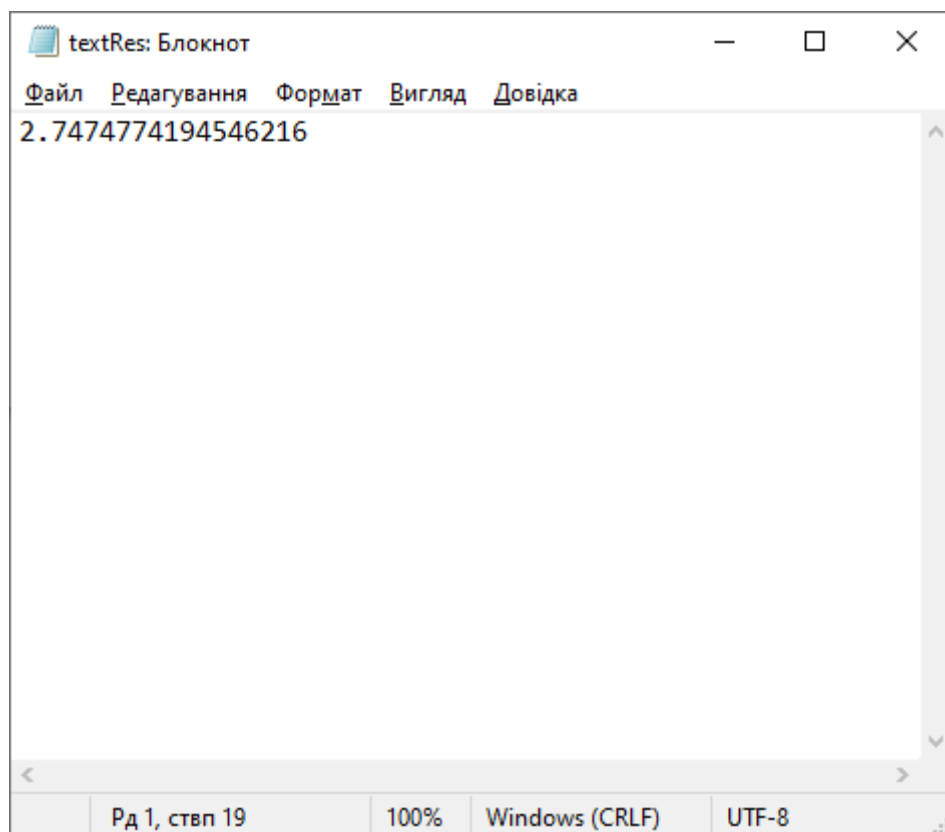
    try:
        Files_utils.write_res_txt("textRes.txt", result)
        Files_utils.write_res_bin("binRes.bin", result)
        print("Result in txt is: {0}".format(Files_utils.read_res_bin("binRes.bin")))
        print("Result in bin is: {0}".format(Files_utils.read_res_txt("textRes.txt")))
    except FileNotFoundError as e:
        print(e)
        sys.exit(1)
```

Результат виконання програми:



```
C:\Users\Admin\AppData\Local\Programs\Python\Python310\python.exe
Enter X: 35
Result in python is : 2.7474774194546216
Result in txt is: 2.7474774194546216
Result in bin is: 2.7474774194546216
Press any key to continue . . .
```

Результат виконання програми в textRes.txt:



Відповіді на контрольні запитання:

1. В мові Python виключні ситуації обробляються за допомогою конструкції ``try`-`except``.
2. Блок ``except`` використовується для обробки виняткових ситуацій, які виникають під час виконання коду в блоку ``try``.
3. Функція, яка використовується для відкриття файлів у Python, називається ``open()``.
4. Функція ``open()`` приймає два аргументи: шлях до файлу і режим відкриття.
5. Файл можна відкрити в різних режимах, таких як "читання" (``r``), "запис" (``w``), "додавання" (``a``), "бінарний режим" (``b``) та інші.
6. Для читання файлу використовується метод ``read()`` або ітерація по файловому об'єкту. Для запису - метод ``write()``.
7. Функції в мові Python можуть приймати аргументи, повертати значення, бути вкладеними, а також бути передані в якості аргументу іншій функції.
8. Оператор ``with`` використовується для автоматичного управління контекстом. Він забезпечує відкриття та закриття ресурсів в правильному порядку.
9. Об'єкти, що передаються під контроль оператору ``with``, повинні мати методи ``__enter__`` та ``__exit__``.
10. Обробка виключних ситуацій може бути вбудована в оператор ``with``, щоб гарантувати коректне закриття ресурсів, навіть якщо виникає виняткова ситуація.

Висновок:

Під час лабораторної роботи, я оволодів навиками використання засобів мови Python для роботи з файлами.