

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



Звіт

з лабораторної роботи № 7
з дисципліни: «Кросплатформенні засоби програмування»

На тему: «Файли та виключення в Python»

Виконав:

студент групи КІ-307
Бажулін С.В.

Прийняв:

доцент кафедри ЕОМ
Іванов Ю. С.

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

Завдання(варіант №2):

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.

Індивідуальне завдання:

$$2. y = \text{ctg}(x)$$

Вихідний код програми:

```
import os
import struct
import sys
import math

def writeResTxt(fName, result):
    with open(fName, "w") as f:
        f.write(str(result))

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

def writeResBin(fName, result):
    with open(fName, "wb") as f:
        f.write(struct.pack("f", result))
```

```

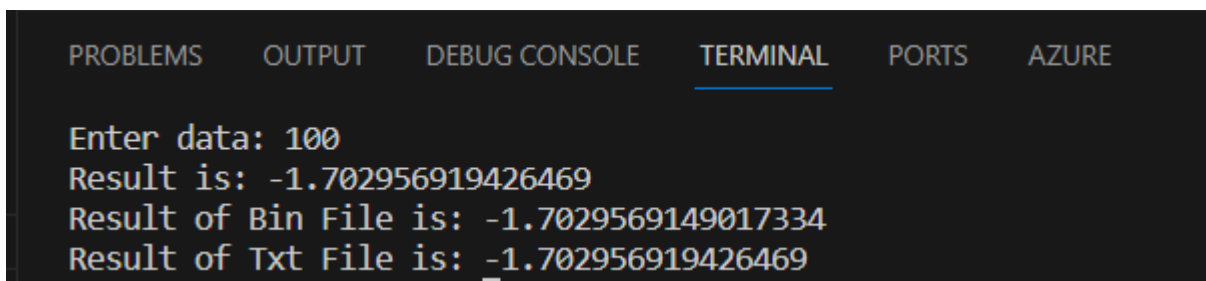
def readResBin(fName):
    result = 0.0
    try:
        if os.path.exists(fName):
            with open(fName, "rb") as f:
                result = struct.unpack("f", f.read())[0]
        else:
            raise FileNotFoundError(f"File {fName} not found.")
    except FileNotFoundError as e:
      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

def calculate(x):
    return 1 / math.tan(x)

if __name__ == "__main__":
    resTxt = "D:/repos/CPPT_LABS/LAB_08/textRes.txt"
    binTxt = "D:/repos/CPPT_LABS/LAB_08/binRes.txt"
    data = float(input("Enter data: "))
    result = calculate(data)
    print(f"Result is: {result}")
    try:
        writeResTxt(resTxt, result)
        writeResBin(binTxt, result)
        print("Result is: {0}".format(readResBin(binTxt)))
        print("Result is: {0}".format(readResTxt(resTxt)))
    except FileNotFoundError as e:
        print(e)
    sys.exit(1)

```

Результат роботи програми:



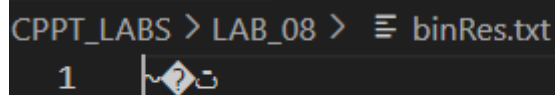
```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  AZURE

Enter data: 100
Result is: -1.702956919426469
Result of Bin File is: -1.7029569149017334
Result of Txt File is: -1.702956919426469

```

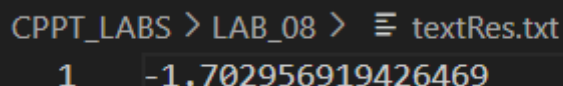
Файли txtRes.txt і binRes.txt



```

CPPT_LABS > LAB_08 > binRes.txt
1 -1.7029569149017334

```



```

CPPT_LABS > LAB_08 > textRes.txt
1 -1.702956919426469

```

Відповідь на контрольні питання

1. Обробка виключень:

- У мові Python обробка виключень використовує конструкцію `try...except`, яка дозволяє обробляти виняткові ситуації під час виконання програми.

2. Особливості роботи блоку `except`:

- Блок `except` використовується для обробки виключень. Він виконується, якщо сталася виняткова ситуація і співпадає з типом винятку.

3. Функція для відкривання файлів у Python:

- Для відкриття файлів у Python використовується функція `open()`.

4. Особливості використання функції `open`:

- Функція `open()` використовується для відкриття файлів у різних режимах, таких як читання, запис, додавання і бінарний режим.

5. Режими відкриття файлу:

- Режими включають 'r' (читання), 'w' (запис), 'a' (додавання), 'b' (бінарний режим) та інші.

6. Читання і запис файлу:

- Для читання файлу використовуйте методи `read()` або ітерацію по файловому об'єкту. Для запису - використовуйте метод `write()`.

7. Особливості функцій у Python:

- Функції в Python - це фрагменти коду, які виконують певну дію та можуть бути викликані з інших частин програми.

8. Призначення оператора `with`:

- Оператор `with` використовується для створення контексту, який автоматично відкриває та закриває ресурси, такі як файли.

9. Вимоги до об'єктів, що передаються під контроль оператора `with`:

- Об'єкти, які передаються під контроль оператора `with`, повинні мати методи `__enter__` та `__exit__`.

10. Поєднання обробки виключень і оператора `with`:

- Обробка виключень може бути впроваджена у методах `__enter__` та `__exit__` об'єкта, що передається під контроль `with`, для відловлювання і обробки помилок.

Висновок: на цій лабораторній роботі, я оволодів навичками використання засобів мови Python для роботи з файлами.