

**Міністерство освіти і науки України**  
**Національний університет «Львівська політехніка»**

**Кафедра ЕОМ**

**Звіт**



**Лабораторна робота № 8**

**“ФАЙЛИ ТА ВИКЛЮЧЕННЯ У PYTHON”**  
з курсу “Кросплатформні засоби програмування”  
Варіант: 4

Виконав:

ст.гр.КІ-205

Воробець Тетяна

Прийняв:

доцент кафедри ЕОМ

Олексів М.В.

**Львів 2024**

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

**Теоретичні відомості:** Функції у мові python не відрізняються за своєю суттю від функцій C/C++. У Python функції можуть мати довільну кількість параметрів. У цьому випадку їм можна передавати неіменовані або іменовані параметри, або їх комбінацію. Мова Python має вбудований механізм обробки виключних ситуацій. Обробка виключних ситуацій забезпечується блоками try-except-finally.

Ключовою функцією для роботи з файлами є функція `open(file, mode='r', buffering=-1, encoding=None, errors=None, newline=None, closefd=True, opener=None)`. Вона повертає дескриптор відкритого файлу або `None`.

Параметри функції: - `file` – шлях до файлу - `mode` – режим відкривання файлу. Може приймати наступні значення та їх комбінації.

### Режими відкривання файлу

Параметр	Значення
'r'	Відкрити для читання (за замовчуванням)
'w'	Відкрити для запису, очистивши попередньо файл, якщо файл існує
'x'	Відкрити для ексклюзивного створення, якщо файл уже існує, то функція завершується невдачею
'a'	Відкрити для запису, дописуючи в кінець файлу, якщо він існує
'b'	Бінарний режим
't'	Текстовий режим (за замовчуванням)
'+'	Відкрити для оновлення (читання та запис)

Читання з файлів здійснюється за допомогою методу `read` об'єкту-файлу. Для читання одnobайтних текстових рядків достатньо викликати метод `read` (для читання всього файлу чи певної кількості байт, кількість яких передається аргументом методу), або методу `readline` (для по-рядкового читання з файлу). Для читання даних інших типів вони мають бути записані як байтові послідовності, які вичитуються методом `read` після чого приводяться до відповідного типу. Для цього можна використати модуль `struct`, який призначений для полегшення інтерпретації байт як запакованих бінарних даних. Він перетворює значення Python на структури C, представлені як байтові об'єкти

Запис у файл здійснюється за допомогою методу `write` об'єкту-файлу. Для запису одnobайтних текстових рядків достатньо їх передати у метод `write`. При запису двійкових даних їх необхідно спочатку перетворити у послідовність байт. Для цього можна використати приведення до типу даних `bytearray`, метод `to_bytes` типу даних (наприклад, `int.to_bytes(var)`), або використати модуль `struct`. Тож, для запису бінарних даних їх спочатку треба запакувати у об'єкт, який являє собою послідовність байт та записати цю послідовність у файл.

**Завдання:** 1. Написати та налагодити програму на мові Python згідно варіанту(4.  $y = \cos(x)/\sin(x)$ ). 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.

### Код програми:

#### writer.py

```
import os
import struct

#Write in text file
def wr_txt(f_name, result):
    with open(f_name, 'w') as f:
        f.write(f"{result:.6f}\n")

#Read in text file
def rd_txt(f_name):
    try:
        if os.path.exists(f_name):
            with open(f_name, 'r') as f:
                result = float(f.readline())
            return result
        except FileNotFoundError as ex:
            return "File {f_name} not found"

#Write in bin file
def wr_bin(f_name, result):
    with open(f_name, 'wb') as f:
        f.write(struct.pack('d', result))

#Read in bin file
def rd_bin(f_name):
    try:
        with open(f_name, 'rb') as f:
            result = struct.unpack('d', f.read(8))[0]
        return result
    except FileNotFoundError as ex:
        print(ex)
        return "File {f_name} not found"
```

#### example.py

```
import math

#calculate example  $\cos(x)/\sin(x)$ 
def calculate(x):
    try:
        return math.cos(x) / math.sin(x)
    except ArithmeticError:
        raise ArithmeticError
```

#### main.py

```
import writer, example

#Start project

if __name__ == "__main__":
    try:
```

```

x = int(input("Enter x (in radians): "))
try:
    result = example.calculate(x)

    writer.wr_bin("Res.bin", result)
    print("Res bin: ", writer.rd_bin("Res.bin"))

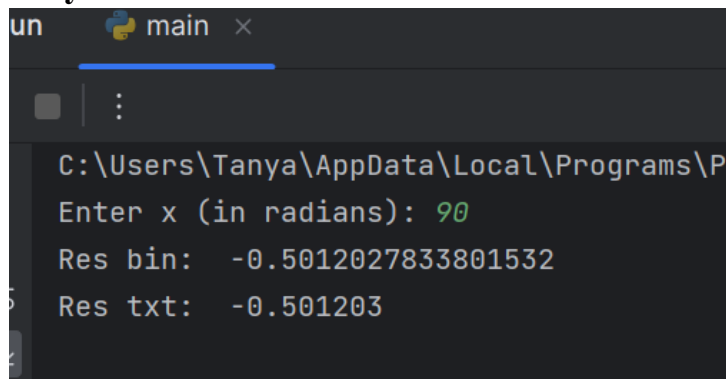
    writer.wr_txt("Res.txt", result)
    print("Res txt: ", writer.rd_txt("Res.txt"))

except ArithmeticError as ex:
    print("Error: Divine to zero")

except ValueError:
    print("Error: Wrong type number")

```

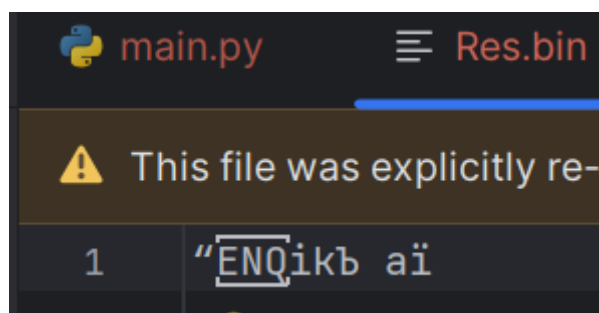
## Результат:



```

C:\Users\Tanya\AppData\Local\Programs\Python\Python39\python.exe main.py
Enter x (in radians): 90
Res bin: -0.5012027833801532
Res txt: -0.501203

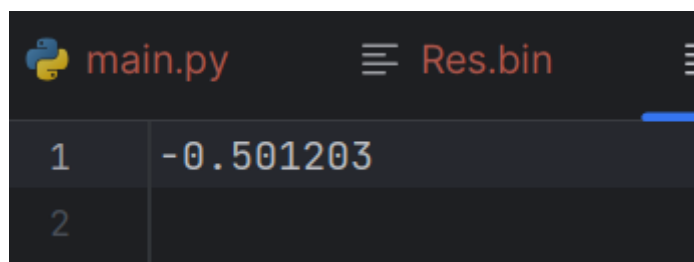
```



```

1 ENQikb aï

```



```

1 -0.501203
2

```

## Посилання на репозиторій:

[https://github.com/NikaDe7/CPPT\\_Vorobets\\_TI\\_KI-35\\_1.git](https://github.com/NikaDe7/CPPT_Vorobets_TI_KI-35_1.git)

**Висновок:** оволоділа навиками використання засобів мови Python для роботи з файлами.