

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 в Python»

Виконав:
студент групи КІ-306
Бокало П.М.

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:

Файл Bokalo.KI_306_1.Lab8.py:

```
from Calco import Calco, ZeroInputError

class Lab8BokaloKI306:
    def main(self):
        x = 0
        try:
            x = float(input("Input your X : "))
        except ValueError:
            print("Invalid data inputted")
            return

        c = None

        try:
            c = Calco(x)
        except ZeroInputError as ex1:
            print(ex1)

        if c is not None:
            try:
                print("Ctg of x =", c.result())
            except ArithmeticError as ex:
                print(ex)

        self.process_and_print_results(c, "Lab8.txt", "Lab8.bin")

    def process_and_print_results(self, calco_instance, text_filename, binary_filename):
        calco_instance.in_text_file(text_filename)
```

```

with open(text_filename, "r") as fin:
    text_content = fin.read()
    print("Result from text file =", text_content)

calco_instance.in_binary_file(binary_filename)

result_from_binary_file = calco_instance.read_binary_file(binary_filename)
if result_from_binary_file is not None:
    print("Result from binary file =", result_from_binary_file)

if __name__ == "__main__":
    lab8 = Lab8BokaloKI306()
    lab8.main()

```

Файл Calco.py:

```

import math
import pickle

class ZeroInputError(Exception):
    pass

class Calco:
    def __init__(self, x):
        if x == 0:
            raise ZeroInputError("Exception: X is equal to 0!!!")
        self.x = x
        self.ctg = 0

    def result(self):
        if self.x == 0:
            raise ArithmeticError("Exception: X is equal to 0!!!")
        try:
            self.ctg = 1 / math.tan(self.x)
        except ZeroDivisionError:
            raise ArithmeticError("Exception: Division by zero error!")
        return self.ctg

    def in_text_file(self, filename):
        try:
            with open(filename, "w") as fout:
                fout.write(str(self.result()))
        except IOError:
            print("Can't use the file!!!")
        except ArithmeticError as a:
            print(a)

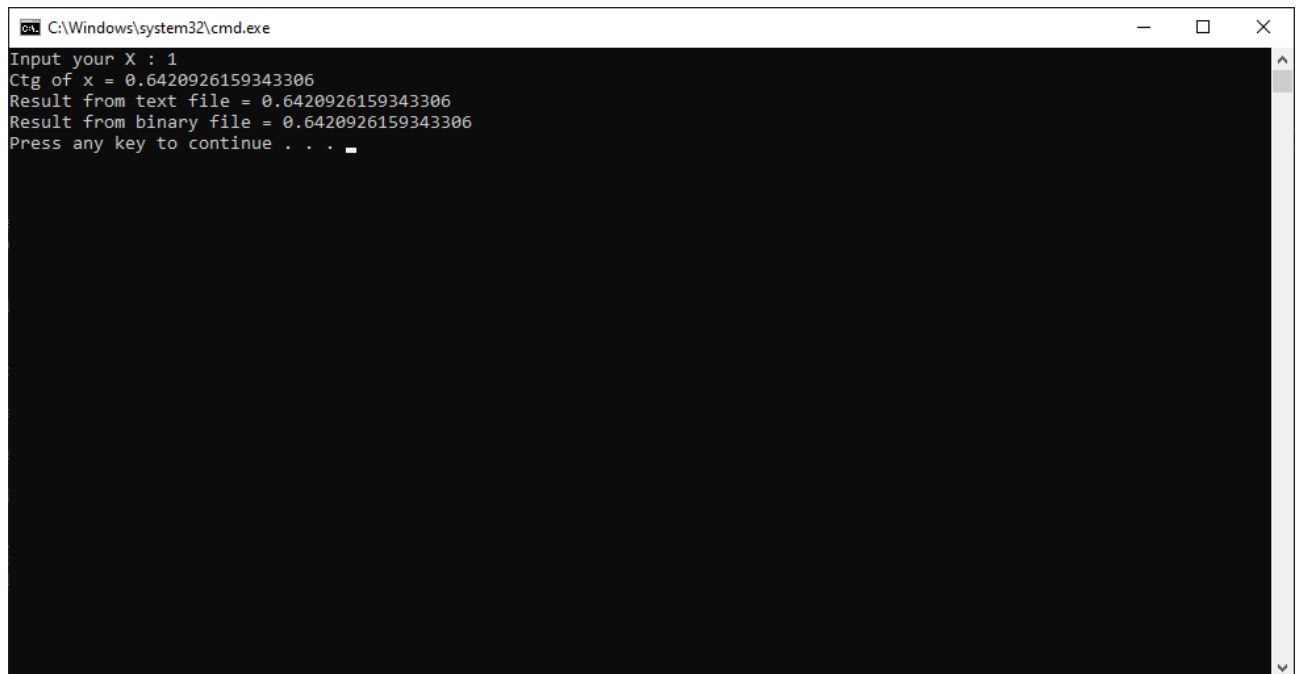
    def in_binary_file(self, filename):
        try:
            with open(filename, "wb") as fout:
                pickle.dump(self.result(), fout)
        except IOError:
            print("Can't use the file!!!")
        except ArithmeticError as a:
            print(a)

    def read_binary_file(self, filename):
        try:
            with open(filename, "rb") as fin:
                result = pickle.load(fin)
                if result is None:
                    raise ArithmeticError("Exception: Result is not available in the
file!!!")
            return result

```

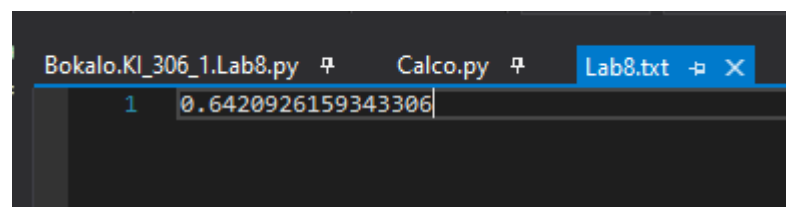
```
except IOError:
    print("Can't read the file!!!")
    return None
except ArithmeticError as a:
    print(a)
```

Результат роботи програми:



A screenshot of a Windows command prompt window titled "C:\Windows\system32\cmd.exe". The window has standard Windows window controls (minimize, maximize, close) in the top right corner. The command prompt shows the following text: "Input your X : 1", "Ctg of x = 0.6420926159343306", "Result from text file = 0.6420926159343306", "Result from binary file = 0.6420926159343306", and "Press any key to continue . . .". The cursor is positioned after the last line of text.

Файл Lab1.txt



A screenshot of a code editor window with three tabs: "Bokalo.KI_306_1.Lab8.py", "Calco.py", and "Lab8.txt". The "Lab8.txt" tab is active and highlighted in blue. The editor shows a single line of text: "1 0.6420926159343306". The line number "1" is in the left margin, and the text "0.6420926159343306" is in the main editing area.

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.