МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Кафедра інформаційних систем та мереж

Звіт

до лабораторної роботи №2

Основи побудови об’єктно-орієнтованих додатків на Python

Виконала:

ст. гр. PI-32 Довгошия А.А

Балів	**Дата**

Прийняв

: асис. каф.

ІСМ

Щербак С.С

Львів — 2024

Мета: Розробка консольного калькулятора в об’єктно орієнтованому стилі з використанням класів

Хід роботи

Завдання 1: Створення класу Calculator Створіть клас Calculator, який буде служити основою для додатка калькулятора.

Завдання 2: Ініціалізація калькулятора Реалізуйте метод init у класі Calculator для ініціалізації необхідних атрибутів або змінних.

Завдання 3: Введення користувача Перемістіть функціональність введення користувача в метод у межах класу Calculator. Метод повинен приймати введення для двох чисел і оператора.

Завдання 4: Перевірка оператора Реалізуйте метод у класі Calculator, щоб перевірити, чи введений оператор є дійсним (тобто одним із +, -, *, /). Відобразіть повідомлення про помилку, якщо він не є дійсним.

Завдання 5: Обчислення Створіть метод у класі Calculator, який виконує обчислення на основі введення користувача (наприклад, додавання, віднімання, множення, ділення).

Завдання 6: Обробка помилок Реалізуйте обробку помилок у межах класу Calculator для обробки ділення на нуль або інших потенційних помилок. Відобразіть відповідні повідомлення про помилку.

Завдання 7: Повторення обчислень Додайте метод до класу Calculator, щоб запитати користувача, чи він хоче виконати ще одне обчислення. Якщо так, дозвольте йому ввести нові числа і оператор. Якщо ні, вийдіть з програми.

Завдання 8: Десяткові числа Модифікуйте клас Calculator для обробки десяткових чисел (плаваюча кома) для більш точних обчислень.

Завдання 9: Додаткові операції Розширте клас Calculator, щоб підтримувати додаткові операції, такі як піднесення до степеня (^), квадратний корінь (√) та залишок від ділення (%).

Завдання 10: Інтерфейс, зрозумілий для користувача Покращте інтерфейс користувача у межах класу Calculator, надавши чіткі запити, повідомлення та форматування виводу для зручності читання. Main.py

```
from calculator import Calculator

if __name__ == "__main__":
calc = Calculator()
   calc.askAgain()
```

runner.py

```
from calculator import Calculator from
calculatorUi import CalculatorUI
from userInput import UserInput

def run_calculator():     ui
= CalculatorUI()
calculator = Calculator()
   user_input = UserInput(ui)

   while True:
ui.display_layout()
x = user_input.get_number(ui.prompts["enter_x"])
       operator = user_input.get_operator(ui.prompts["enter_operation"])

y = None         if operator != "√":
          y = user_input.get_number(ui.prompts["enter_y"])

       result = calculator.calculate(x, y, operator)
if isinstance(result, str):
ui.show_error(result)         else:
          ui.show_result(result)
```

```
        if not user_input.ask_for_more(ui.prompts["ask_again"]):
break
```

calculator.py

```
import math

class Calculator:     def
__init__(self):
self.operations = {             "+":
lambda x, y: x + y,
            "-": lambda x, y: x - y,
            "*": lambda x, y: x * y,
            "/": self._safe_division,
            "^": lambda x, y: x ** y,
            "%": lambda x, y: x % y,
            "√": self._safe_sqrt,
        }

    def _safe_division(self, x, y):         return
"division by zero" if y == 0 else x / y

    def _safe_sqrt(self, x):         return "square root from negative
num" if x < 0 else math.sqrt(x)

    def calculate(self, num1, num2, operator):
if operator == "√":             return
self.operations[operator](num1)
        return self.operations[operator](num1, num2)
```

calculatorUi.py

```
class CalculatorUI:
def __init__(self):
self.prompts = {
"enter_x": "x: ",
            "enter_y": "y: ",
            "enter_operation": "# operation: ",
            "result": "~ result: {}",
```

```
            "ask_again": "\n# would you like to continue? [y/n]: "
        }

    def display_layout(self):
        layout = (
"+   -   *   \n"
"7   8   9   ^\n"
"4   5   6   √\n"
            "1   2   3   %\n"
            "0   .\n"
        )
        print(layout)

    def show_result(self, result):
print(self.prompts["result"].format(result))

    def show_error(self, message):
print(f"! error: {message}\n")
```

userInput.py

```
class UserInput:
    def __init__(self, ui):
        self.ui = ui

    def get_number(self, prompt):
while True:             try:
                return float(input(prompt))
except ValueError:
self.ui.show_error("not a number")

    def get_operator(self, prompt):
while True:
            operator = input(prompt)            if operator in
["+", "-", "*", "/", "^", "%", "√"]:
                return operator
            self.ui.show_error("unknown operation")

    def ask_for_more(self, prompt):
```

return input(prompt).lower() == "y"

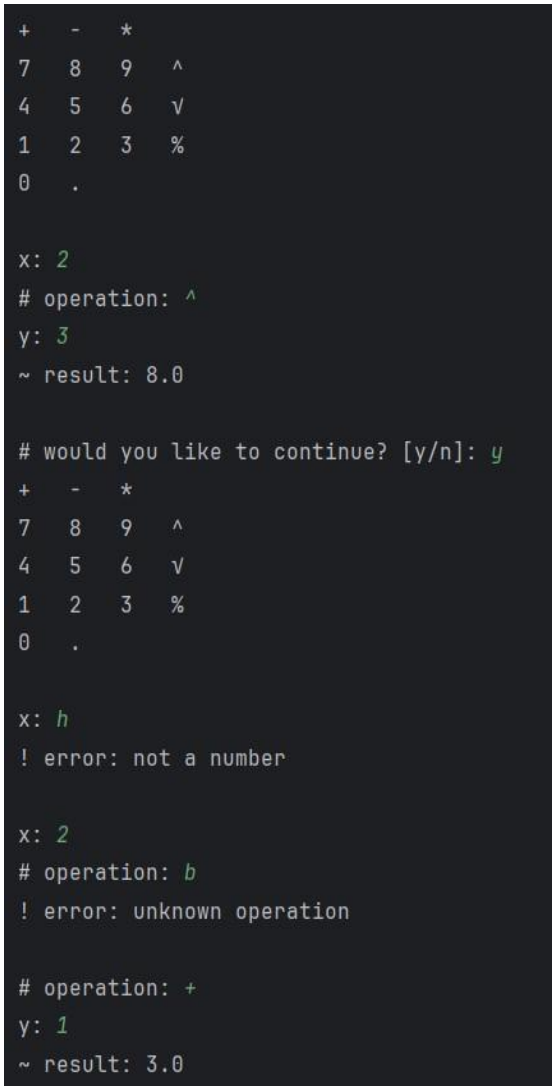

Рис.1. Результат програми.

Висновок: на цій лабораторній роботі я вивчила розробку консольного калькулятора в об’єктно орієнтованому стилі з використанням класів