

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Кафедра інформаційних систем та мереж

Лабораторна робота №2
з дисципліни
СПЕЦІАЛІЗОВАНІ МОВИ ПРОГРАМУВАННЯ
на тему
Основи побудови об'єктно-орієнтованих додатків на Python

Виконав:
ст. гр. РІ-21сп
Рак В.П.
Прийняв:
Щербак С.С.

Мета лабораторної роботи: Розробка консольного калькулятора в об'єктно орієнтованому стилі з використанням класів

Завдання

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.

Виконання роботи

Текст програмної реалізації:

```
import math
```

```
class Calculator:
```

```
    """
```

Клас для виконання математичних обчислень. Підтримує операції додавання, віднімання, множення,

ділення, піднесення до степеня, обчислення квадратного кореня і залишку від ділення.

Атрибути:

result (float): Змінна для збереження результату обчислення.

num1 (float): Перше введене число.

num2 (float): Друге введене число.

operator (str): Оператор для виконання обчислення.

```
    """
```

```
def __init__(self):
```

```
    """
```

Ініціалізує калькулятор, встановлюючи результат обчислень на 0.

```
    """
```

```
    self.result = 0
```

```
def get_user_input(self):
```

```
    """
```

Запитує у користувача перше число, оператор і друге число (якщо необхідно).

Повертає:

bool: True, якщо введення правильне, False — якщо сталася помилка введення.

```
    """
```

```
    try:
```

```
        self.num1 = float(input("Введіть перше число: "))
```

```
        self.operator = input("Введіть оператор (+, -, *, /, ^, √, %): ")
```

```
        if self.operator != "√": # Для квадратного кореня потрібно тільки одне число
            self.num2 = float(input("Введіть друге число: "))
except ValueError:
    print("Помилка: введіть дійсне число.")
    return False
return True
```

```
def check_operator(self):
```

```
    """
```

Перевіряє правильність введеного оператора.

Повертає:

bool: True, якщо оператор допустимий, False — якщо оператор неправильний.

```
    """
```

```
    if self.operator in ['+', '-', '*', '/', '^', '√', '%']:
```

```
        return True
```

```
    else:
```

```
        print("Помилка: недійсний оператор.")
```

```
        return False
```

```
def calculate(self):
```

```
    """
```

Виконує математичні обчислення на основі введених чисел і оператора.

Повертає:

bool: True, якщо обчислення виконано успішно, False — якщо сталася помилка (наприклад, ділення на нуль).

```
    """
```

```
    try:
```

```
        if self.operator == '+':
```

```
            self.result = self.num1 + self.num2
```

```
        elif self.operator == '-':
```

```
            self.result = self.num1 - self.num2
```

```

elif self.operator == '*':
    self.result = self.num1 * self.num2
elif self.operator == '/':
    if self.num2 == 0:
        raise ZeroDivisionError
    self.result = self.num1 / self.num2
elif self.operator == '^':
    self.result = self.num1 ** self.num2
elif self.operator == '√':
    if self.num1 < 0:
        raise ValueError("Неможливо обчислити квадратний корінь з від'ємного числа.")
    self.result = math.sqrt(self.num1)
elif self.operator == '%':
    self.result = self.num1 % self.num2
except ZeroDivisionError:
    print("Помилка: ділення на нуль.")
    return False
except ValueError as e:
    print(e)
    return False
return True

```

```

def display_result(self):
    """
    Виводить результат обчислення на екран.
    """
    print(f'Результат: {self.result}')

```

```

def ask_for_repeat(self):
    """
    Запитує користувача, чи бажає він виконати ще одне обчислення.

    Повертає:

```

bool: True, якщо користувач хоче повторити обчислення, False — якщо ні.

```
"""
```

```
repeat = input("Бажаєте виконати ще одне обчислення? (так/ні): ").strip().lower()
```

```
return repeat == 'так'
```

```
def run(self):
```

```
"""
```

Основний цикл роботи калькулятора, який керує процесом введення даних, виконанням обчислень

і запитом на повторне обчислення.

```
"""
```

```
while True:
```

```
    if not self.get_user_input():
```

```
        continue
```

```
    if not self.check_operator():
```

```
        continue
```

```
    if not self.calculate():
```

```
        continue
```

```
    self.display_result()
```

```
    if not self.ask_for_repeat():
```

```
        print("Дякую за використання калькулятора!")
```

```
        break
```

Функція для запуску лабораторної роботи 2

```
def run_lab2():
```

```
"""
```

Запускає лабораторну роботу 2, створюючи об'єкт калькулятора і викликаючи його основний цикл.

```
"""
```

```
print("Лабораторна робота 2: виконання другої лабораторної...")
```

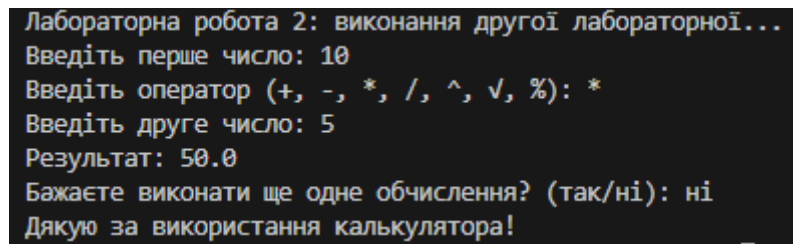
```
# Створюємо об'єкт калькулятора та запускаємо його
```

```
calculator = Calculator()
```

```
calculator.run()
```

```
if __name__ == "__main__":  
    run_lab2() # Для запуску лабораторної роботи 2
```

Результат роботи програми:



```
Лабораторна робота 2: виконання другої лабораторної...  
Введіть перше число: 10  
Введіть оператор (+, -, *, /, ^, √, %): *  
Введіть друге число: 5  
Результат: 50.0  
Бажаєте виконати ще одне обчислення? (так/ні): ні  
Дякую за використання калькулятора!
```

Рис. 1 - Використання калькулятора

Висновок: У ході виконання ЛР я перетворив консольний калькулятор у об'єктно-орієнтований калькулятор, використовуючи класи в Python. Калькулятор виконує обчислення, зберігає результат у пам'яті та має зручний інтерфейс.