МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ "ЛЬВІВСЬКА ПОЛІТЕХНІКА"

Кафедра інформаційних систем та мереж

   
 

Лабораторна робота №2

з курсу

Спеціалізовані мови програмування

   
 

Виконав студент

групи ІТ-21сп

**Гула В. М.**

Львів – 2023

**Лабораторна робота № 2.**

Основи побудови об’єктно-орієнтованих додатків на Python

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

**План 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.

**Реалізація:**

class BaseCalculator:

def \_\_init\_\_(self, valid\_operators: list[str]):

self.\_operator = None

self.\_second\_number = None

self.\_first\_number = None

self.\_\_valid\_operators = valid\_operators

def do\_operation(self):

while True:

result = self.perform\_operation()

if result is not None:

print(f"Result: {result}")

answer = input("Do you want to perform another operation?(y/n)\n")

if answer.lower() != 'y':

break

def perform\_operation(self):

self.\_\_user\_input()

result = None

if self.\_operator == '+':

result = self.\_first\_number + self.\_second\_number

elif self.\_operator == '-':

result = self.\_first\_number - self.\_second\_number

elif self.\_operator == '\*':

result = self.\_first\_number \* self.\_second\_number

elif self.\_operator == '/':

if self.\_second\_number == 0:

print("Cannot divide by zero\n")

else:

result = self.\_first\_number / self.\_second\_number

return result

def \_\_user\_input(self):

self.\_first\_number = self.\_\_enter\_number("Enter first number\n")

self.\_second\_number = self.\_\_enter\_number("Enter second number\n")

self.\_operator = input("Enter operator\n")

def \_\_enter\_operator(self):

while True:

operator = input("Enter operator:\n")

if operator not in self.\_\_valid\_operators:

print(f"Operator is not valid:{operator}")

else:

self.\_operator = operator

break

@staticmethod

def \_\_enter\_number(input\_msg: str = "Enter number:\n") -> float:

while True:

user\_input = input(input\_msg)

try:

number = float(user\_input)

return number

except ValueError:

print("Invalid number entered.")

class ModernCalculator(BaseCalculator):

def \_\_init\_\_(self, valid\_operators: list[str]):

super().\_\_init\_\_(valid\_operators)

def do\_operation(self):

while True:

result = self.perform\_operation()

print(f"Result: {result}")

answer = input("Do you want to perform another operation?(y/n)\n")

if answer.lower() != 'y':

break

def perform\_operation(self):

result = super().perform\_operation()

if result is None:

if self.\_operator == '%':

result = self.\_first\_number % self.\_second\_number

elif self.\_operator == '^':

result = self.\_first\_number \*\* self.\_second\_number

elif self.\_operator == 'sqrt':

result = self.\_\_get\_sqrt()

else:

print("Invalid operator")

return result

def \_\_get\_sqrt(self) -> float:

print(f"Your numbers:\n1 - {self.\_first\_number}\n2 - {self.\_second\_number}")

option = int(input("Choose the number that should be converted?\n"))

if option == 1:

if self.\_first\_number < 0:

print(f"Cannot get square of {self.\_first\_number} because it is < 0")

else:

return math.sqrt(self.\_first\_number)

if option == 2:

if self.\_second\_number < 0:

print(f"Cannot get square of {self.\_second\_number} because it is < 0")

else:

return math.sqrt(self.\_second\_number)

def main():

valid\_operators = ['+', '-', '\*', '/', '^', '%', 'sqrt']

calc = ModernCalculator(valid\_operators)

calc.do\_operation()

if \_\_name\_\_ == '\_\_main\_\_':

main()

**Висновок:** розробив консольний калькулятора в об’єктно орієнтованому стилі з використанням класів