МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

ІНСТИТУТ КОМП'ЮТЕРНИХ НАУК ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра ІСМ

******

Звіт

до лабораторної роботи №2

На тему “Основи побудови об’єктно-орієнтованих додатків на Python”

З дисципліни “Спеціалізовані мови програмування”

*Виконав:*

*ст. гр. ІТ-31*

*Лисецький Володимир*

*Прийняв:*

*Щербак С. С.*

*Львів - 2023*

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

**Хід роботи**

**Програмний код:**

/operations/calculator.py

import math

class Calculator:

def \_\_init\_\_(self, operation=None, first\_number=None, second\_number=None):

if not (isinstance(first\_number, (int, float, type(None)))):

raise ValueError("first\_number must be int, float, or None")

if not (isinstance(second\_number, (int, float, type(None)))):

raise ValueError("second\_number must be int, float, or None")

if operation not in ("+", "-", "/", "\*", "^", "√"):

raise ValueError("Invalid operator. Please enter one of: +, -, /, \*, ^, √")

else:

self.operation = operation

self.first\_number = first\_number

self.second\_number = second\_number

self.result = None

def calculate():

return 0

class Sum(Calculator):

def \_\_init\_\_(self, first\_number=None, second\_number=None):

super().\_\_init\_\_('+', first\_number, second\_number)

def calculate(self):

self.result = self.first\_number + self.second\_number

return self.result

class Subtraction(Calculator):

def \_\_init\_\_(self, first\_number=None, second\_number=None):

super().\_\_init\_\_('-', first\_number, second\_number)

def calculate(self):

self.result = self.first\_number - self.second\_number

return self.result

class Multiplication(Calculator):

def \_\_init\_\_(self, first\_number=None, second\_number=None):

super().\_\_init\_\_('\*', first\_number, second\_number)

def calculate(self):

self.result = self.first\_number \* self.second\_number

return self.result

class Division(Calculator):

def \_\_init\_\_(self, first\_number=None, second\_number=None):

super().\_\_init\_\_('/', first\_number, second\_number)

def calculate(self):

try:

self.result = self.first\_number / self.second\_number

except ZeroDivisionError:

print("error: you can not divide by zero")

return self.result

class Power(Calculator):

def \_\_init\_\_(self, first\_number=None, second\_number=None):

super().\_\_init\_\_('^', first\_number, second\_number)

def calculate(self):

self.result = self.first\_number \*\* self.second\_number

return self.result

class SquareRoot(Calculator):

def \_\_init\_\_(self, first\_number=None, second\_number=None):

print(first\_number)

super().\_\_init\_\_('√', first\_number, second\_number)

def calculate(self):

self.result = math.sqrt(self.first\_number)

return self.result

/operations/menu.py

from .calculator import Calculator, Subtraction, Sum, Multiplication, Division, SquareRoot, Power

class Menu():

def \_\_init\_\_(self):

self.calculator = None

self.calculations = [] # To store calculation history

def select\_operation(self):

print("Select an operation:")

print("1. Make Calculation")

print("2. Show History")

print("3. Show Specific Operation from History")

print("0. Exit")

choice = input("Enter your choice: ")

return choice

def make\_calculation(self):

first\_number = float(input("Enter first number: "))

while True:

operation = input("Enter the operator (+, -, /, \*, ^, √): ")

if operation not in ("+", "-", "/", "\*", "^", "√"):

print("Invalid operator. Please enter one of: +, -, /, \*, ^, √")

else:

break

if operation != '√':

second\_number = float(input("Enter second number: "))

if operation == "+":

calculation = Sum(first\_number, second\_number)

elif operation == "-":

calculation = Subtraction(first\_number, second\_number)

elif operation == "\*":

calculation = Multiplication(first\_number, second\_number)

elif operation == "/":

calculation = Division(first\_number, second\_number)

elif operation == "^":

calculation = Power(first\_number, second\_number)

else:

raise ValueError("Invalid operator. Please enter one of: +, -, /, \*, ^, √")

result = calculation.calculate()

print(f"{calculation.first\_number} {calculation.operation} {calculation.second\_number} = {calculation.result}")

self.calculations.append(calculation)

return result

else:

calculation = SquareRoot(first\_number)

calculation.calculate()

print(f"{calculation.operation} ({calculation.first\_number}) = {calculation.result}")

self.calculations.append(calculation)

result = calculation.result

return result

def show\_history(self):

if not self.calculations:

print("No calculations in history.")

else:

for i in range(len(self.calculations)):

print(f"{i + 1}. {self.calculations[i].first\_number} {self.calculations[i].operation} {self.calculations[i].second\_number} = {self.calculations[i].result}")

def show\_specific\_operation(self):

if not self.calculations:

print("No calculations in history.")

return

choice = int(input("Enter the calculation number to show: "))

i = choice - 1

if i < len(self.calculations):

if self.calculations[i].operation != '√':

print(f"{choice}. {self.calculations[i].first\_number} {self.calculations[i].operation} {self.calculations[i].second\_number} = {self.calculations[i].result}")

else:

print(f"{choice}. {self.calculations[i].operation} {self.calculations[i].first\_number} = {self.calculations[i].result}")

else:

print("Invalid choice. Please enter a valid number.")

def run(self):

while True:

choice = self.select\_operation()

if choice == "0":

print("Goodbye!")

break

elif choice == "1":

pass

self.make\_calculation()

elif choice == "2":

self.show\_history()

elif choice == "3":

self.show\_specific\_operation()

elif choice == "0":

return 0

else:

print("Invalid choice. Please try again.")

main.py

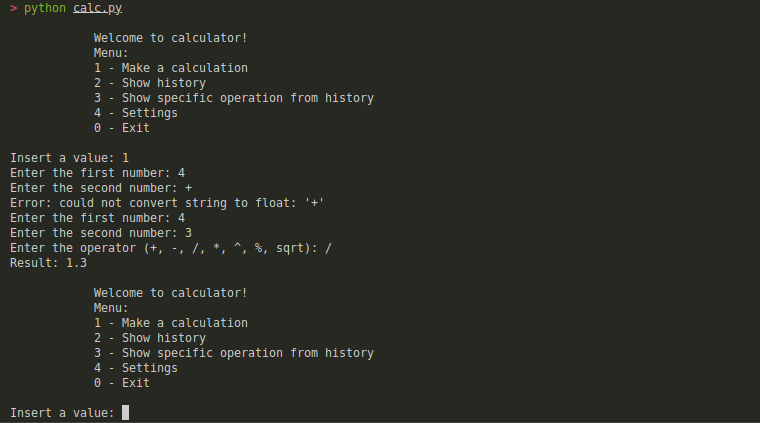
from operations.menu import Menu

if \_\_name\_\_ == "\_\_main\_\_":

menu = Menu()

menu.run()

Результат виконання програми:



**Висновок:** Під час виконання даної лабораторної роботи було створено консольний калькулятор в об’єктно орієнтованому стилі з використанням класів.