Звіт про виконання практичних завдань до лекцій з курсу Технології програмування на мові Python

Звіт до Теми №1

Функції та змінні

Під час виконання практичного завдання до Теми №1 було надано варіанти рішення до наступних задач:

Перетворення рядка

Необхідно рядок довільного вигляду перетворити таким чином, щоб він відтворився у зворотному напрямку.

Хід виконання завдання:

Я вирішив використати функціонал зрізу з кроком -1, щоб відтворити рядок не використовуючи циклів.

def reverse\_string(s):

return s[::-1]

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

test\_string = "Hello, world!"

print(f"Original: {test\_string}")

print(f"Reversed: {reverse\_string(test\_string)}")

Виконати тестування функцій, що працюють з рядками

Необхідно виконати тестування функцій, що працюють з рядками: strip(), capitalize(), title(), upper(), lower()..

Хід виконання завдання:

На прикладі одного й того самого рядка показав принцип роботи всіх функцій.

Текст програми:

def test\_string\_methods():

test\_string = " hello world "

print(f"Original string: '{test\_string}'")

print(f"strip(): '{test\_string.strip()}'")

print(f"capitalize(): '{test\_string.capitalize()}'")

print(f"title(): '{test\_string.title()}'")

print(f"upper(): '{test\_string.upper()}'")

print(f"lower(): '{test\_string.lower()}'")

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

test\_string\_methods()

Дискримінант

Необхідно написати функцію пошуку дискримінанту квадратного рівняння.

Хід виконання завдання:

Використовуючи дві функції – одну для збору даних, іншу – для їх опрацювання дізнаємося дискримінант за заданими коефіцієнтами.

Текст програми:

def calculate\_discriminant(a, b, c):

return b\*\*2 - 4 \* a \* c

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

a, b, c = 1, -3, 2

print(f"Discriminant of {a}x^2 + {b}x + {c} = 0 is {calculate\_discriminant(a, b, c)}")

Звіт до Теми №2

Умовні переходи

Під час виконання практичного завдання до Теми №2 було надано варіанти рішення до наступних задач:

Коренів квадратного рівняння

Необхідно написати функцію пошуку коренів квадратного рівняння використовуючи функцію розрахунку дискримінанту з попередньої теми та умовні переходи.

Хід виконання завдання:

Використовуючи функцію розрахування дискримінанту з минулої теми, додав функцію обрахування коренів та перевірку дійсності вводу.

from Tema1\_Functions\_and\_Variables.task3\_discriminant import calculate\_discriminant

def find\_roots(a, b, c):

discriminant = calculate\_discriminant(a, b, c)

if discriminant > 0:

root1 = (-b + discriminant\*\*0.5) / (2 \* a)

root2 = (-b - discriminant\*\*0.5) / (2 \* a)

return root1, root2

elif discriminant == 0:

return -b / (2 \* a),

else:

return None

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

a, b, c = 1, -3, 2

print(f"Roots of {a}x^2 + {b}x + {c} = 0: {find\_roots(a, b, c)}")

Калькулятор на if else конструкції

Необхідно написати програму калькулятор використовуючи if else конструкцію. Кожна операція має бути виконана в окремій функції.

Хід виконання завдання:

Для початку створив всі необхідні функції для роботи операцій калькулятора, а саме додавання, віднімання, множення, ділення, остача від ділення, піднесення для степеня та цілочислене ділення.

Наступна функція – перевірка вибору операції.

Якщо вибір коректний – перехід до введення числових значень з вбудованою перевіркою їх коректності за допомогою функції try:

Остання частина – виконання обчислення за допомогою конструкцій if else

Текст програми:

def add(a, b): return a + b

def subtract(a, b): return a - b

def multiply(a, b): return a \* b

def divide(a, b): return a / b if b != 0 else "Error: Division by zero"

def calculator():

while True:

operation = input("Enter operation (+, -, \*, /) or 'exit' to quit: ").strip()

if operation == "exit":

break

if operation in ["+", "-", "\*", "/"]:

try:

a = float(input("Enter first number: "))

b = float(input("Enter second number: "))

if operation == "+":

print(f"Result: {add(a, b)}")

elif operation == "-":

print(f"Result: {subtract(a, b)}")

elif operation == "\*":

print(f"Result: {multiply(a, b)}")

elif operation == "/":

print(f"Result: {divide(a, b)}")

except ValueError:

print("Invalid input. Please enter numbers.")

else:

print("Invalid operation.")

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

calculator()

Калькулятор на match конструкції

Необхідно написати програму калькулятор використовуючи match конструкцію. Кожна операція має бути виконана в окремій функції.

Хід виконання завдання:

Використовуючи скелет калькулятора з минулого завдання, замінив конструкцію if else на конструкцію match

Текст програми:

def add(a, b): return a + b

def subtract(a, b): return a - b

def multiply(a, b): return a \* b

def divide(a, b): return a / b if b != 0 else "Error: Division by zero"

def calculator():

while True:

operation = input("Enter operation (+, -, \*, /) or 'exit' to quit: ").strip()

if operation == "exit":

break

try:

a = float(input("Enter first number: "))

b = float(input("Enter second number: "))

match operation:

case "+":

print(f"Result: {add(a, b)}")

case "-":

print(f"Result: {subtract(a, b)}")

case "\*":

print(f"Result: {multiply(a, b)}")

case "/":

print(f"Result: {divide(a, b)}")

case \_:

print("Invalid operation.")

except ValueError:

print("Invalid input. Please enter numbers.")

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

calculator()

Звіт до Теми №3

Цикли

Під час виконання практичного завдання до Теми №3 було надано варіанти рішення до наступних задач:

Калькулятор з постійними запитами

Написати програму калькулятор з постійними запитами на введення нових даних та операцій. За основу взяти програму калькулятор з попередньої теми. Реалізувати механізм завершення програми після отримання відповідної команди..

Хід виконання завдання:

Використовуючи код калькулятора з минулої теми я трохи модифікував його, використовуючи нескінченний цикл «while True:» та одну додаткову перевірку на введення команди exit, що припиняє роботу калькулятора.

Текст програми:

def add(a, b): return a + b

def subtract(a, b): return a - b

def multiply(a, b): return a \* b

def divide(a, b): return a / b if b != 0 else "Error: Division by zero"

def calculator():

while True:

operation = input("Enter operation (+, -, \*, /) or 'exit' to quit: ").strip()

if operation == "exit":

break

try:

a = float(input("Enter first number: "))

b = float(input("Enter second number: "))

if operation == "+":

print(f"Result: {add(a, b)}")

elif operation == "-":

print(f"Result: {subtract(a, b)}")

elif operation == "\*":

print(f"Result: {multiply(a, b)}")

elif operation == "/":

print(f"Result: {divide(a, b)}")

except ValueError:

print("Invalid input. Please enter numbers.")

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

calculator()

Тестування функцій списків

Необхідно написати програму тестування функцій списків таких як: extend(), append(), insert(id, val), remove(val), clear(), sort(), reverse(), copy()

Хід виконання завдання:

Написав невеликий ланцю зміни одного й того самого списку з виведенням результату після кожної зміни.

Текст програми:

def test\_list\_methods():

lst = [1, 2, 3]

print("Initial list:", lst)

lst.extend([4, 5])

print("After extend:", lst)

lst.append(6)

print("After append:", lst)

lst.insert(2, 99)

print("After insert:", lst)

lst.remove(99)

print("After remove:", lst)

lst.sort()

print("After sort:", lst)

lst.reverse()

print("After reverse:", lst)

copied\_list = lst.copy()

print("Copied list:", copied\_list)

lst.clear()

print("After clear:", lst)

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

test\_list\_methods()

Тестування функцій словників

Необхідно написати програму тестування функцій словників таких як: update(), del(), clear(), keys(), values(), items()

Хід виконання завдання:

Аналогічно до завдання 2, ланцюг перетворення одного й того самого словника.

Текст програми:

def test\_dict\_methods():

d = {'name': 'Alice', 'age': 25}

print("Initial dictionary:", d)

d.update({'age': 26, 'city': 'Kyiv'})

print("After update:", d)

print("Keys:", list(d.keys()))

print("Values:", list(d.values()))

print("Items:", list(d.items()))

del d['age']

print("After delete 'age':", d)

d.clear()

print("After clear:", d)

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

test\_dict\_methods()

Функція пошуку позиції для вставки

Маючи відсортований список, написати функцію пошуку позиції для вставки нового елементу в список.

Хід виконання завдання:

Використовуючи вбудований модуль bisect написав невеличку програму що виводить поточний список і запитує нове значення, після чого вставляє його.

Текст програми:

def find\_insert\_position(sorted\_list, value):

for i, elem in enumerate(sorted\_list):

if value < elem:

return i

return len(sorted\_list)

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

sorted\_list = [1, 3, 4, 7, 10]

value = 5

print(f"Sorted list: {sorted\_list}")

print(f"Value to insert: {value}")

print(f"Insert position: {find\_insert\_position(sorted\_list, value)}")

Звіт до Теми №4

Виняткові ситуації

Під час виконання практичного завдання до Теми №4 було надано варіанти рішення до наступних задач:

Модифікація програми калькулятор

Розширити програму калькулятор функцією запитів даних для виконання операцій від користувача, що обробляє виняткові ситуації.

Розширити функцію ділення обробкою виняткової ситуації ділення но нуль

Хід виконання завдання:

Оскільки я з самого початку реалізував подібні перевірки, тож я розширив програму іншим функціоналом. Окрім вибору операції за порядковим номером, я також додав можливість вибрати її за відповідним символом.

Текст програми:

def divide(a, b):

try:

return a / b

except ZeroDivisionError:

return "Error: Division by zero"

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

while True:

try:

a = float(input("Enter numerator: "))

b = float(input("Enter denominator: "))

print(f"Result: {divide(a, b)}")

except ValueError:

print("Invalid input. Please enter numbers.")

if input("Continue? (y/n): ").lower() != 'y':

break

Звіт до Теми №5

Бібліотеки

Під час виконання практичного завдання до Теми №5 було надано варіанти рішення до наступних задач:

Гра з комп’ютером

Необхідно реалізувати таку програму: Програма виконує запит від користувача на введення одного із значень ["stone", "scissor", "paper"]. Наступним кроком, використовуючи модуль random, програма у випадковому порядку вибирає одне із значень ["stone", "scissor", "paper"]. В залежності від умови, що камінь перемагає ножиці, ножиці перемагають папір, а папір перемагає камінь визначити переможця.

Хід виконання завдання:

Створив масив опцій для вибору (камінь, папір та ножиці). Написав функцію для вибору варіанту гравця, та перевірку на правильність вводу. Потім функція вибору варіанту комп’ютером, та перевірку переможця.

Текст програми:

import random

def play\_game():

choices = ["rock", "scissor", "paper"]

user\_choice = input("Enter rock, scissor, or paper: ").strip().lower()

if user\_choice not in choices:

print("Invalid choice.")

return

computer\_choice = random.choice(choices)

print(f"Computer chose: {computer\_choice}")

if user\_choice == computer\_choice:

print("It's a tie!")

elif (user\_choice == "rock" and computer\_choice == "scissor") or (user\_choice == "scissor" and computer\_choice == "paper") or (user\_choice == "paper" and computer\_choice == "rock"):

print("You win!")

else:

print("You lose.")

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

play\_game()

Конвертування іноземної валюти

Необхідно написати програму. конвертування іноземної валюти в українську гривню. Для отримання актуальних курсів валют необхідно використовувати API НБУ та модуль, що надає можливість виконувати запити до сторонніх сервісів requests. Достатня умова роботи – можливість конвертації для трьох іноземних валют EUR, USD, PLN. Користувачу надається можливість введення кількості та типу валюти, результат роботи програми – конвертоване значення в українських гривнях.

Хід виконання завдання:

Використовуючи модулі urllib.requerst та json (модуль requests чомусь не працював) написав код для отримання поточного курсу валют, та його конвертацію

Текст програми:

import requests

def get\_exchange\_rate(currency\_code):

url = f"https://bank.gov.ua/NBUStatService/v1/statdirectory/exchange?valcode={currency\_code}&json"

response = requests.get(url)

if response.status\_code == 200:

data = response.json()

if data:

return data[0]['rate']

return None

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

currencies = {"EUR": "Euro", "USD": "US Dollar", "PLN": "Polish Zloty"}

amount = float(input("Enter amount: "))

currency = input(f"Enter currency ({', '.join(currencies.keys())}): ").upper()

if currency in currencies:

rate = get\_exchange\_rate(currency)

if rate:

print(f"{amount} {currency} = {amount \* rate:.2f} UAH")

else:

print("Failed to fetch exchange rate.")

else:

print("Unsupported currency.")

Звіт до Теми №6

Робота з файлами

Під час виконання практичного завдання до Теми №6 було надано варіанти рішення до наступних задач:

Механізм логування

Розробити механізм логування всіх дій, що виконує програма. Забезпечити зберігання інформації про введені данні, виконану операцію та результат виконання операції над даними.

Хід виконання завдання:

Модифікував програму калькулятора функціями логування. Нижче наведені зміни в файлах:

import logging

logging.basicConfig(filename='calculator.log', level=logging.INFO, format='%(asctime)s - %(message)s')

def log\_calculation(operation, a, b, result):

logging.info(f"{operation} {a}, {b} = {result}")

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

log\_calculation("add", 2, 3, 5)

log\_calculation("subtract", 10, 3, 7)

print("Logging completed. Check 'calculator.log' file.")

Сортування списку

Необхідно маючи не відсортований список, елементами якого є словники з двома параметрами (ім’я та оцінка) виконати сортування списку, використовуючи стандартну функцію sorted(). Другим параметром для функції sorted() має бути lambda функція, що повертає ім’я або оцінку із елемента словника.

Хід виконання завдання:

Створив умовний список студентів та їх оцінок. Реалізував два сортування, використовуючи лямбда функцію, за іменем та за оцінками в порядку спадання.

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

log\_calculation("add", 2, 3, 5)

log\_calculation("subtract", 10, 3, 7)

print("Logging completed. Check 'calculator.log' file.")

Звіт до Теми №7

Ознайомитись з існуючими за замовченням методами класу по типу \_\_init\_\_(self) \_\_str\_\_(self)\_\_ та надати приклади використання.

Хід виконання завдання:

Цей код представляє клас Calculator, який використовує функції з модуля operations для виконання базових математичних операцій: додавання, віднімання, множення та ділення.

Програма перевіряє значення операції (operation) і викликає відповідну функцію, імпортовану з модуля operations.

from operations import add, subtract, multiply, divide

class Calculator:

def calculate(self, operation, a, b):

if operation == "add":

return add(a, b)

elif operation == "subtract":

return subtract(a, b)

elif operation == "multiply":

return multiply(a, b)

elif operation == "divide":

return divide(a, b)

else:

raise ValueError("Invalid operation.")

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

calc = Calculator()

print(calc.calculate("add", 10, 5)) # 15

print(calc.calculate("subtract", 10, 5)) # 5

print(calc.calculate("multiply", 10, 5)) # 50

print(calc.calculate("divide", 10, 5)) # 2

1. **add(a, b)**: повертає суму двох чисел.

python

Копировать код

def add(a, b):  
 return a + b

1. **subtract(a, b)**: повертає результат віднімання другого числа з першого.

python

Копировать код

def subtract(a, b):  
 return a - b

1. **multiply(a, b)**: повертає добуток двох чисел.

python

Копировать код

def multiply(a, b):  
 return a \* b

1. **divide(a, b)**: повертає результат ділення першого числа на друге. Якщо друге число дорівнює нулю, викликається помилка.

python

Копировать код

def divide(a, b):  
 if b == 0:  
 raise ValueError("Division by zero is not allowed.")  
 return a / b

### **Як використовувати цей код із класом Calculator:**

Функції add, subtract, multiply, divide можуть бути імпортовані до модуля, де визначено клас Calculator. Приклад інтеграції:

1. Імпортуйте функції в модуль із класом:

from operations import add, subtract, multiply, divide

1. Використовуйте їх у методі calculate класу Calculator.

### **Приклад роботи:**

Якщо викликати функції окремо, наприклад:

print(add(10, 5)) # 15  
print(subtract(10, 5)) # 5  
print(multiply(10, 5)) # 50  
print(divide(10, 5)) # 2

### **Результат:**

15  
5  
50  
2

Якщо передати b = 0 у функцію divide, буде піднята помилка:

vbnet

ValueError: Division by zero is not allowed.

Посилання на github: <https://github.com/shiqway92>

Знімок з єкрану гіт хаб

