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

**Зміст**

[Завдання №1.1 5](#_Toc149832028)

[Хід виконання завдання: 5](#_Toc149832029)

[Текст програми: 5](#_Toc149832030)

[Посилання на github: 5](#_Toc149832031)

[Знімок екрану з посилання на github: 5](#_Toc149832032)

[Завдання №1.2 6](#_Toc149832033)

[Хід виконання завдання: 6](#_Toc149832034)

[Текст програми: 6](#_Toc149832035)

[Посилання на github: 6](#_Toc149832036)

[Знімок екрану з посилання на github: 6](#_Toc149832037)

[Завдання №1.3 8](#_Toc149832038)

[Хід виконання завдання: 8](#_Toc149832039)

[Текст програми: 8](#_Toc149832040)

[Посилання на github: 8](#_Toc149832041)

[Знімок екрану з посилання на github: 8](#_Toc149832042)

[Завдання №2.1 9](#_Toc149832043)

[Хід виконання завдання: 9](#_Toc149832044)

[Текст програми: 9](#_Toc149832045)

[Посилання на github: 9](#_Toc149832046)

[Знімок екрану з посилання на github: 9](#_Toc149832047)

[Завдання №2.2 11](#_Toc149832048)

[Хід виконання завдання: 11](#_Toc149832049)

[Текст програми: 11](#_Toc149832050)

[Посилання на github: 11](#_Toc149832051)

[Знімок екрану з посилання на github: 11](#_Toc149832052)

[Завдання №2.3 13](#_Toc149832053)

[Хід виконання завдання: 13](#_Toc149832054)

[Текст програми: 13](#_Toc149832055)

[Посилання на github: 13](#_Toc149832056)

[Знімок екрану з посилання на github: 13](#_Toc149832057)

[Завдання №3.1 15](#_Toc149832058)

[Хід виконання завдання: 15](#_Toc149832059)

[Текст програми: 15](#_Toc149832060)

[Посилання на github: 16](#_Toc149832061)

[Знімок екрану з посилання на github: 16](#_Toc149832062)

[Завдання №3.2 17](#_Toc149832063)

[Хід виконання завдання: 17](#_Toc149832064)

[Текст програми: 17](#_Toc149832065)

[Посилання на github: 18](#_Toc149832066)

[Знімок екрану з посилання на github: 18](#_Toc149832067)

[Завдання №3.3 19](#_Toc149832068)

[Хід виконання завдання: 19](#_Toc149832069)

[Текст програми: 19](#_Toc149832070)

[Посилання на github: 19](#_Toc149832071)

[Знімок екрану з посилання на github: 19](#_Toc149832072)

[Завдання №3.4 20](#_Toc149832073)

[Хід виконання завдання: 20](#_Toc149832074)

[Текст програми: 20](#_Toc149832075)

[Посилання на github: 20](#_Toc149832076)

[Знімок екрану з посилання на github: 20](#_Toc149832077)

[Завдання №4.1-2 21](#_Toc149832078)

[Хід виконання завдання: 21](#_Toc149832079)

[Текст програми: 21](#_Toc149832080)

[Посилання на github: 22](#_Toc149832081)

[Знімок екрану з посилання на github: 22](#_Toc149832082)

[Завдання №4.3 24](#_Toc149832083)

[Хід виконання завдання: 24](#_Toc149832084)

[Текст програми: 24](#_Toc149832085)

[Посилання на github: 25](#_Toc149832086)

[Знімок екрану з посилання на github: 25](#_Toc149832087)

[Завдання №5.1 26](#_Toc149832088)

[Хід виконання завдання: 26](#_Toc149832089)

[Текст програми: 26](#_Toc149832090)

[Посилання на github: 27](#_Toc149832091)

[Знімок екрану з посилання на github: 27](#_Toc149832092)

[Завдання №5.2 29](#_Toc149832093)

[Хід виконання завдання: 29](#_Toc149832094)

[Текст програми: 29](#_Toc149832095)

[Посилання на github: 30](#_Toc149832096)

[Знімок екрану з посилання на github: 31](#_Toc149832097)

[Завдання №5.3 32](#_Toc149832098)

[Хід виконання завдання: 32](#_Toc149832099)

[Текст програм: 32](#_Toc149832100)

[Посилання на github: 33](#_Toc149832101)

[Знімок екрану з посилання на github: 33](#_Toc149832102)

[Завдання №6.1 35](#_Toc149832103)

[Хід виконання завдання: 35](#_Toc149832104)

[Текст програм: 35](#_Toc149832105)

[Посилання на github: 37](#_Toc149832106)

[Знімок екрану з посилання на github: 37](#_Toc149832107)

[Завдання №6.2 38](#_Toc149832108)

[Хід виконання завдання: 38](#_Toc149832109)

[Текст програми: 39](#_Toc149832110)

[Посилання на github: 40](#_Toc149832111)

[Знімок екрану з посилання на github: 40](#_Toc149832112)

[Завдання №7.1 41](#_Toc149832113)

[Хід виконання завдання: 41](#_Toc149832114)

[Завдання №7.2 41](#_Toc149832115)

[Хід виконання завдання: 41](#_Toc149832116)

[Текст програми: 41](#_Toc149832117)

[Посилання на github: 43](#_Toc149832118)

[Знімок екрану з посилання на github: 43](#_Toc149832119)

[Завдання №7.3 44](#_Toc149832120)

[Хід виконання завдання: 44](#_Toc149832121)

[Текст програми: 44](#_Toc149832122)

[Посилання на github: 45](#_Toc149832123)

[Знімок екрану з посилання на github: 45](#_Toc149832124)

[Завдання №7.4 46](#_Toc149832125)

[Хід виконання завдання: 46](#_Toc149832126)

[Текст програм: 46](#_Toc149832127)

[Посилання на github: 48](#_Toc149832128)

[Знімок екрану з посилання на github: 48](#_Toc149832129)

Звіт до Теми №1

Функції та змінні

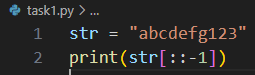
Завдання №1.1

Необхідно рядок, що має вигляд "abcdefg123" перетворити наступним чином "321gfedcba", вважаючи сталою довжину рядку в 10 символів.

Хід виконання завдання:

1. Вписав необхідний текст.
2. Виконав дії для перетворення рядку задом наперед.
3. Виконав програму та перевірив її на успішність.

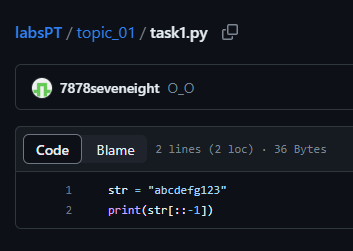
Текст програми:



Посилання на github:

<https://github.com/7878seveneight/labsPT/blob/main/topic_01/task1.py>

Знімок екрану з посилання на github:



Завдання №1.2

Перевірити роботу функцій strip() , capitalize(), title(), upper(), lower() та т.п.

Хід виконання завдання:

1. Зробив змінні для тексту на якому будуть проходити перевірки та написав сам текст.
2. Ввів змінні для зберігання в них форматованого тексту.
3. Надрукував форматований текст.

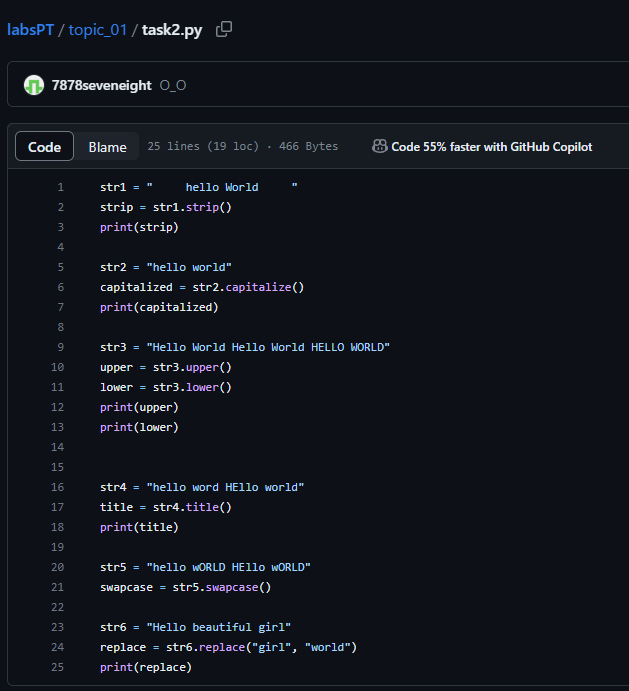
Текст програми:



Посилання на github:

<https://github.com/7878seveneight/labsPT/blob/main/topic_01/task2.py>

Знімок екрану з посилання на github:



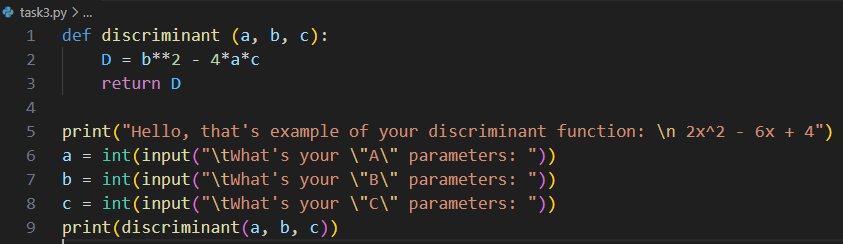
Завдання №1.3

Написати функцію для знаходження дискріминанта.

Хід виконання завдання:

1. Написав функцію, для знаходження дискріминта, з трьома аргументами.
2. Ввів змінні для підставлення аргументів у функцію.
3. Вивів відповідь на приклад.

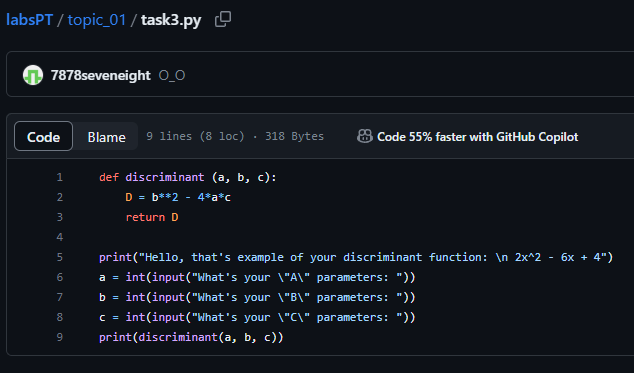
Текст програми:



Посилання на github:

<https://github.com/7878seveneight/labsPT/blob/main/topic_01/task3.py>

Знімок екрану з посилання на github:



Звіт до Теми №2

Функції та змінні

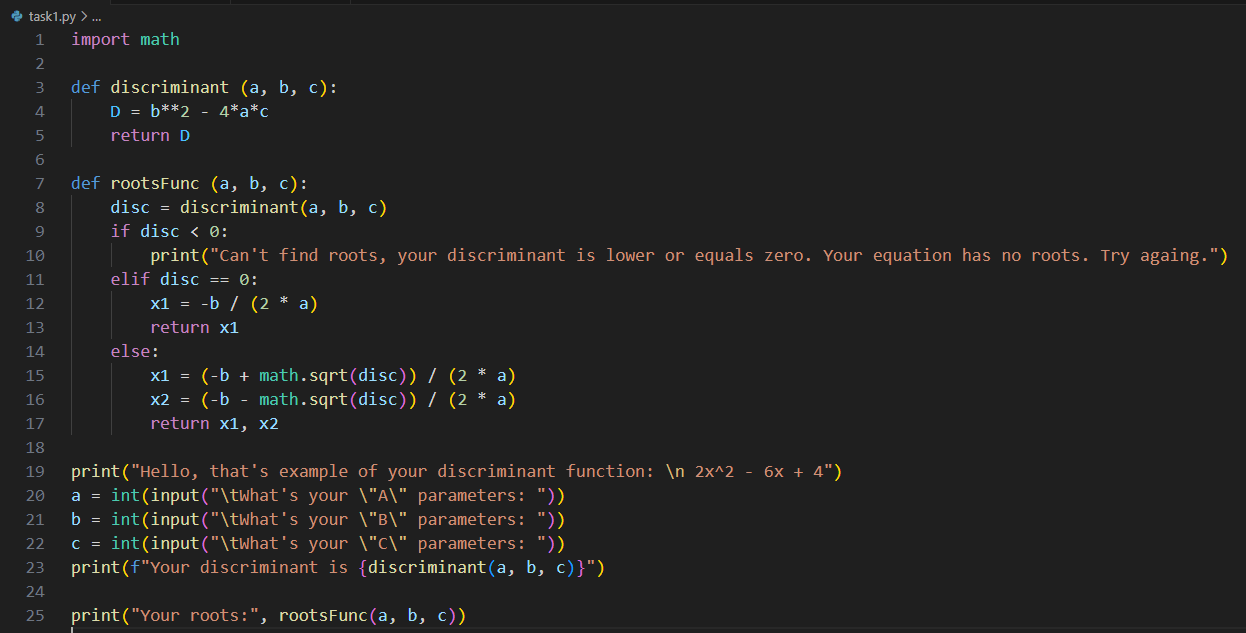
Завдання №2.1

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

Хід виконання завдання:

1. Імпортував бібліотеку math для знахоження квадратного кореня.
2. Вставив код функції дискимінанта з минулого завдання.
3. Написав функцію для знахоження коренів.
4. Дізнався значення змінних від користувача через input.
5. Підставив ці значення та вивів результати.

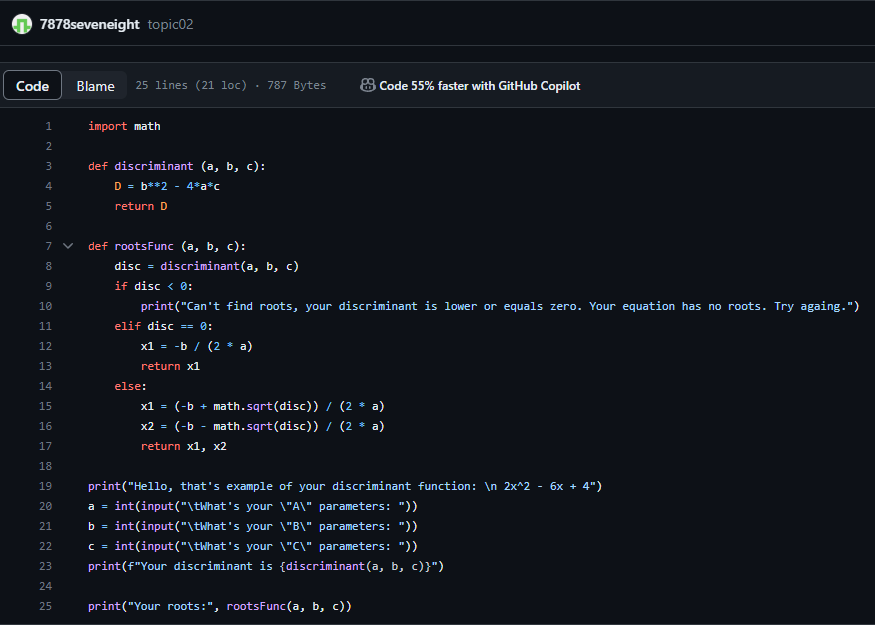
Текст програми:



Посилання на github:

[\*link\*](https://github.com/7878seveneight/TP-KB-222-Petro-Ratozii/blob/main/topic_02/task1.py)

Знімок екрану з посилання на github:



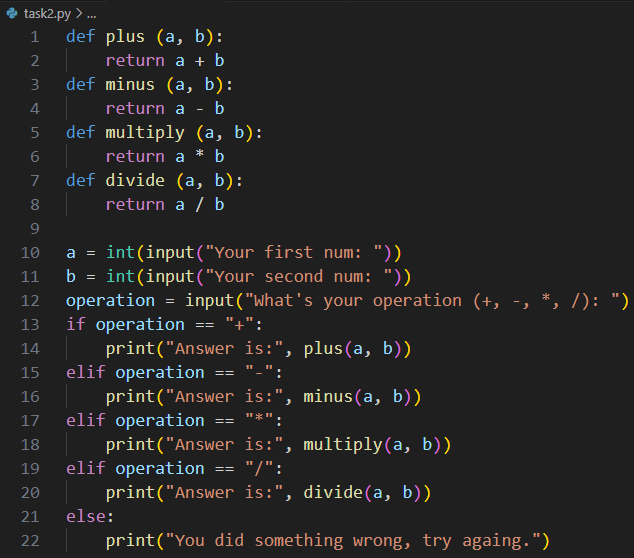
Завдання №2.2

Написати програму калькулятор використовуючи if else конструкцію.

Хід виконання завдання:

1. Написав функції з простими математичними діями.
2. Дізнався значення змінних двох чисел та операцію від користувача через input.
3. Написав логіку програми для підставлення чисел та операції використовуючи if else конструкцію, вивів результат.

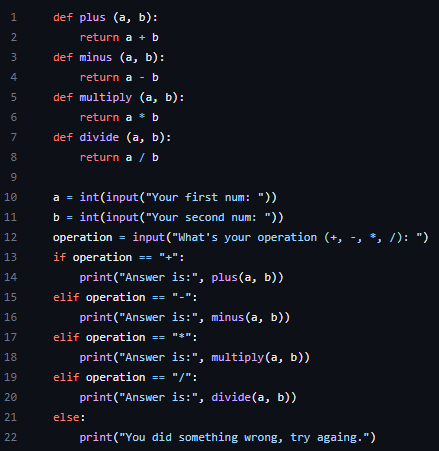
Текст програми:



Посилання на github:

[\*link\*](https://github.com/7878seveneight/TP-KB-222-Petro-Ratozii/blob/main/topic_02/task2.py)

Знімок екрану з посилання на github:



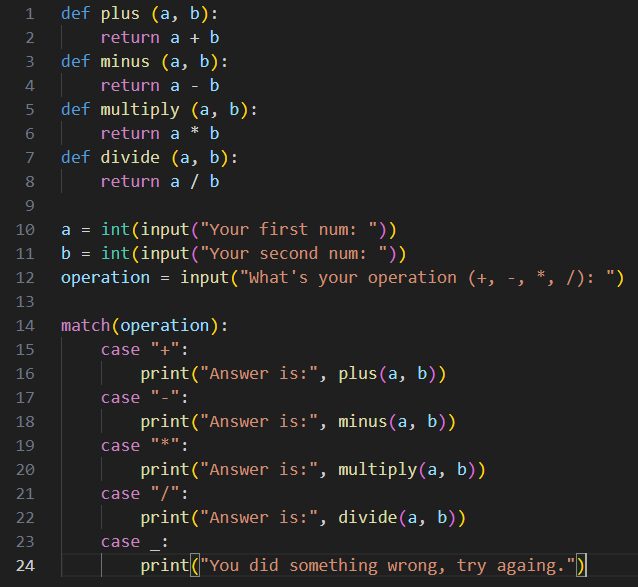
Завдання №2.3

Написати програму калькулятор використовуючи match конструкцію

Хід виконання завдання:

1. Скопіював код, математичних функцій та отримання зміних від користувача, дій зі завдання вище.
2. Написав логіку програми для підставлення чисел та операції використовуючи match конструкцію, вивів результат.

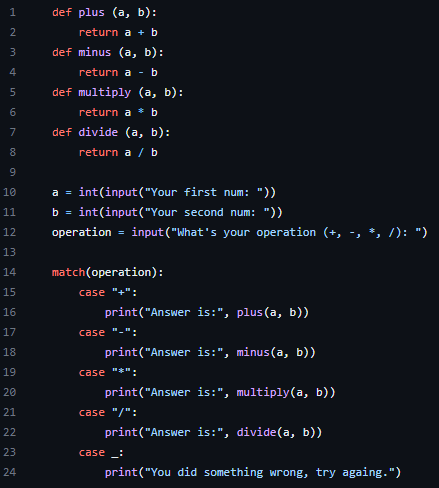
Текст програми:



Посилання на github:

[\*link\*](https://github.com/7878seveneight/TP-KB-222-Petro-Ratozii/blob/main/topic_02/task3.py)

Знімок екрану з посилання на github:



Звіт до Теми №2

Функції та змінні

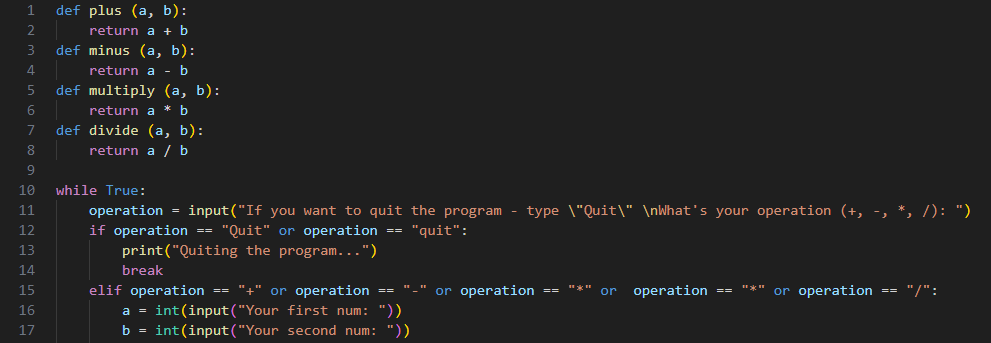
Завдання №3.1

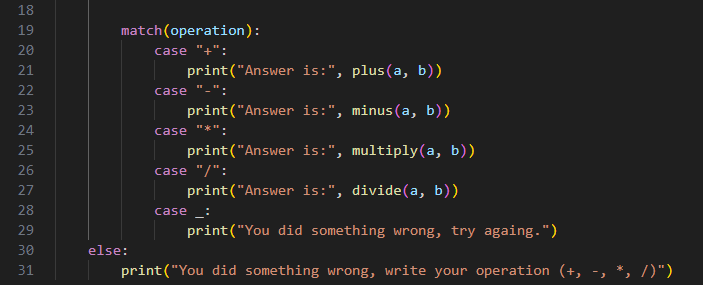
Переписати калькулятор з попередніх завдань використовуючи нескінченний цикл. Запитувати кожен раз дві функції та оператор. У випадку коли нажметься щось не те, додати можливість вийти.

Хід виконання завдання:

1. Скопіював функції та деякі деталі коду калькулятора з попередніх завдань.
2. Зробив нескінченний цикл, який запитує кожен раз дві функції та оператор.
3. У випадку коли нажметься щось не те, додав можливість виходу.

Текст програми:





Посилання на github:

[\*link\*](https://github.com/7878seveneight/TP-KB-222-Petro-Ratozii/blob/main/topic_03/task1.py)

Знімок екрану з посилання на github:



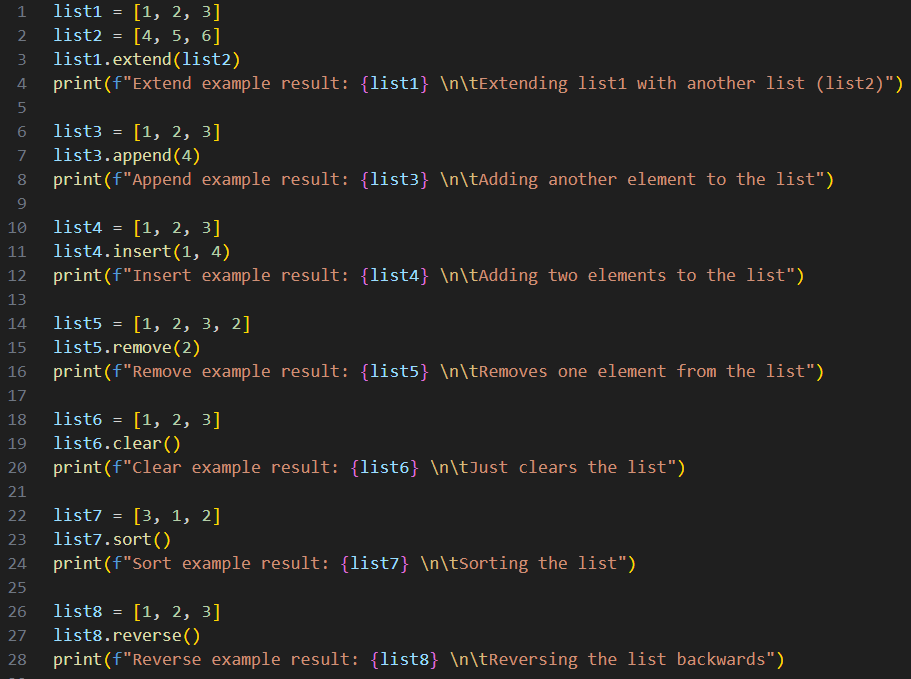
Завдання №3.2

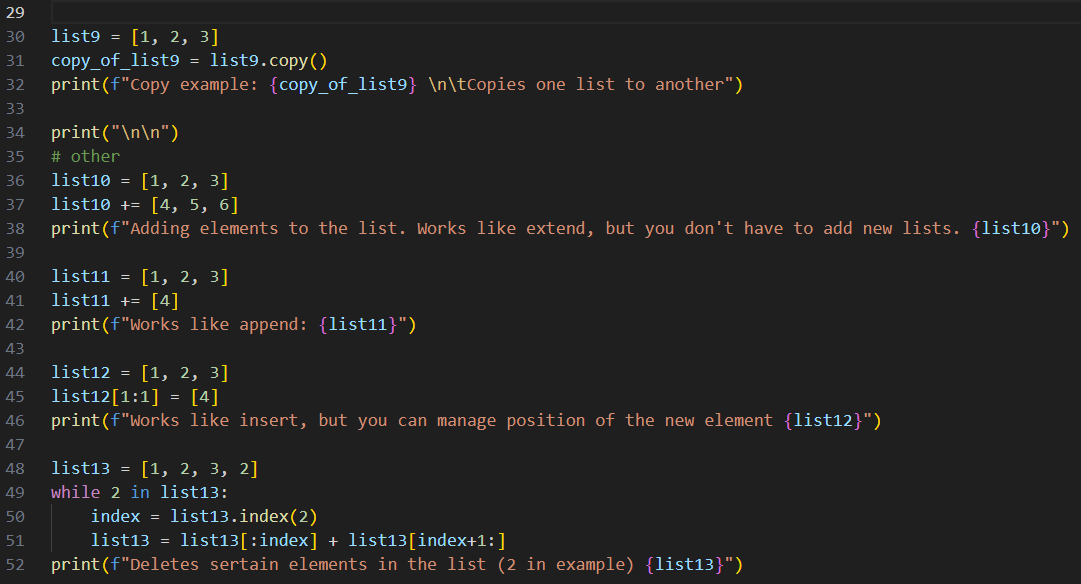
Перевірити поведінку функцій для списків: extend(), append(), insert(id, val), remove(val), clear(), sort(), reverse(), copy(), та інші.

Хід виконання завдання:

1. Знайшов інформацію по цим функціям.
2. Реалізував їх у коді.
3. Описав їх своїми словами, на англійській мові.

Текст програми:

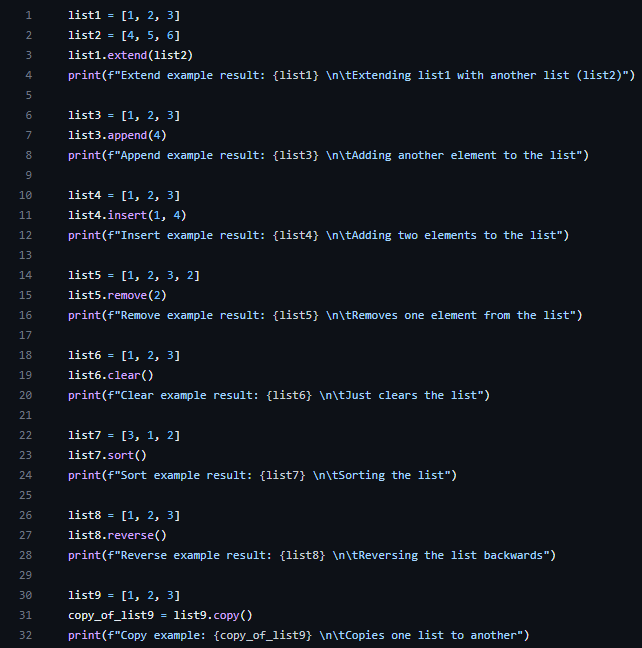


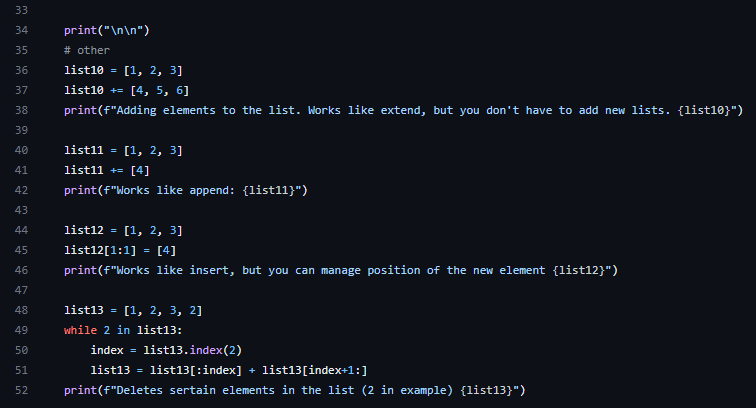


Посилання на github:

[\*link\*](https://github.com/7878seveneight/TP-KB-222-Petro-Ratozii/blob/main/topic_03/task2.py)

Знімок екрану з посилання на github:





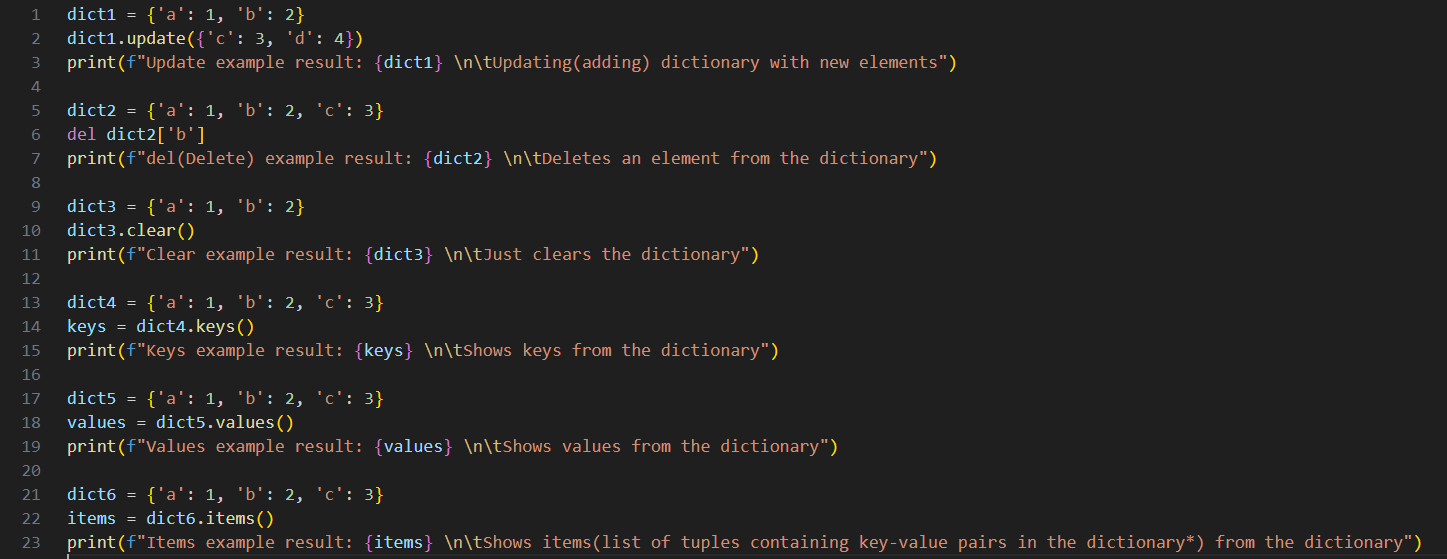
Завдання №3.3

Перевірити поведінку функцій для словників: update(), del, clear, keys(), values(), items().

Хід виконання завдання:

1. Знайшов інформацію по цим функціям.
2. Реалізував їх у коді.
3. Описав їх своїми словами, на англійській мові.

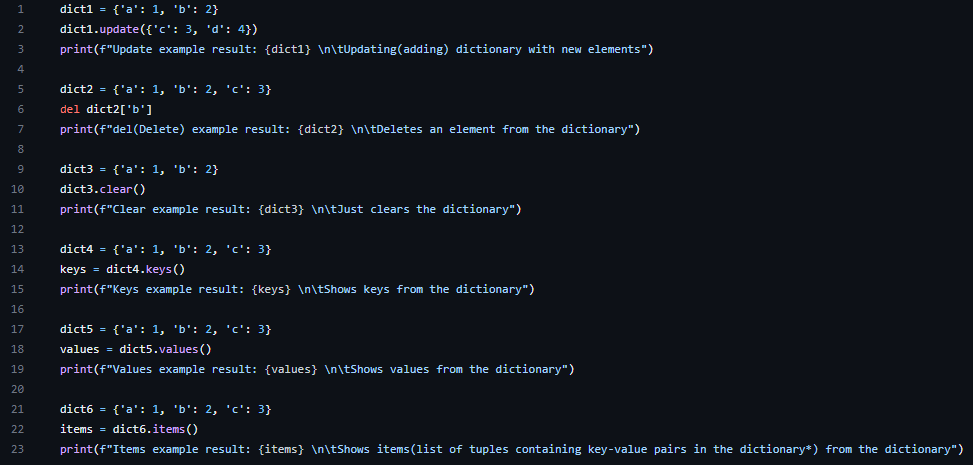
Текст програми:



Посилання на github:

[\*link\*](https://github.com/7878seveneight/TP-KB-222-Petro-Ratozii/blob/main/topic_03/task3.py)

Знімок екрану з посилання на github:



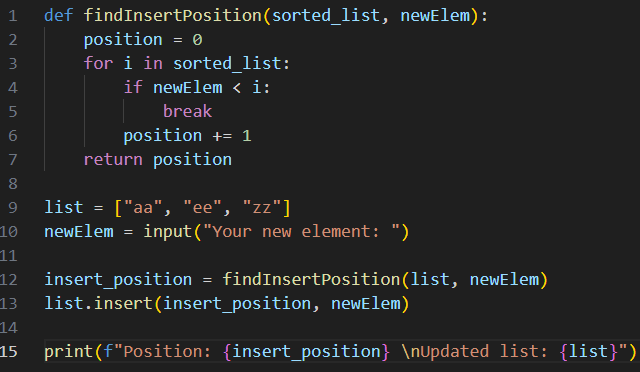
Завдання №3.4

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

Хід виконання завдання:

1. Уважно перечитав завдання, щоб добре осмислити задачу.
2. Написав функцію для пошуку позиції вставлення нового елементу в список, використовуючи цикл та умовний оператор.
3. Задав початковий та відсортований список, створив запит для нового елементу, який необхідно вставити.
4. Підставив аргументи взяті вище.
5. Через функцію insert() вставив новий елемент на позицію визначеною написаною функцією вище.
6. Вивів результати.

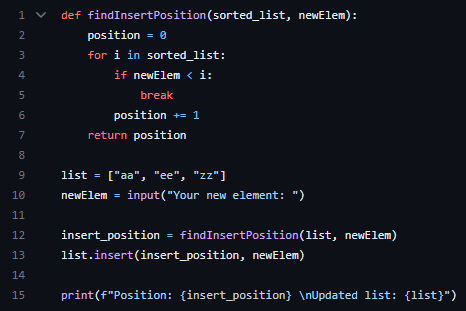
Текст програми:



Посилання на github:

[\*link\*](https://github.com/7878seveneight/TP-KB-222-Petro-Ratozii/blob/main/topic_03/task4.py)

Знімок екрану з посилання на github:



Завдання №4.1-2

Розширити програму калькулятор функцією запитів від користувача, що обробляє виняткові ситуації.

Розширити функцію ділення обробкою виняткової ситуації ділення на нуль.

Хід виконання завдання:

1. Повторив матеріал з лекції.
2. Написав функцію getIntValue, з конструкцією try-except, метою якої є обробка виняткової ситуації, яка виникає коли користувач вводить неправильний тип даних.
3. Доробив функцію ділення, додавши до неї try-except, для того щоб уникнути помилки при випадку коли користувач намагається поділити число на нуль.

Текст програми:

def plus (a, b):

    return a + b

def minus (a, b):

    return a - b

def multiply (a, b):

    return a \* b

def divide (a, b):

    try:

        return a / b

    except ZeroDivisionError:

        print("You can't divide a number by zero. Try againg...")

def getIntValue(promt: str):

    while True:

        try:

            return int(input(promt))

        except ValueError:

            print("Value is not integer. Try againg...")

while True:

    operation = input("If you want to quit the program - type \"Quit\" \nWhat's your operation (+, -, \*, /): ")

    if operation == "Quit" or operation == "quit":

        print("Quiting the program...")

        break

    elif operation == "+" or operation == "-" or operation == "\*" or  operation == "\*" or operation == "/":

        a = getIntValue("Your first num: ")

        b = getIntValue("Your second num: ")

        match(operation):

            case "+":

                print("Answer is:", plus(a, b))

            case "-":

                print("Answer is:", minus(a, b))

            case "\*":

                print("Answer is:", multiply(a, b))

            case "/":

                print("Answer is:", divide(a, b))

            case \_:

                print("You did something wrong, try againg.")

    else:

        print("You did something wrong, write your operation (+, -, \*, /)")

Посилання на github:

[GitHub](https://github.com/7878seveneight/TP-KB-222-Petro-Ratozii/blob/main/topic_04/task_01_02.py)

Знімок екрану з посилання на github:



Завдання №4.3

Ознайомитись зі списком виняткових ситуацій за посиланням <https://docs.python.org/3/library/exceptions.html>

Хід виконання завдання:

1. Ознайомився з матеріалом по посиланню.
2. Знайшов та подивився декілька відео з прикладами на цю тему.
3. Написав декілька конструкцій try-except.

Текст програми:

# ValueError

while True:

    try:

        x = int(input("Enter Int: "))

        print(x)

        break

    except ValueError:

        print("Your value isn't Integer. Print Int...")

# ZeroDivisionError, else, ValueError and Finally(works in any scenario)

while True:

    try:

        x = 5

        y = int(input("Divide 5 with any Int except 0: "))

        print(x/y)

    except ZeroDivisionError:

        print("You must type any Int except 0. Try again..")

    except ValueError:

        print("Your value isn't Integer. Print Int...")

    else:

        break

    finally:

        print("0\_0")

# KeyError

dict = {'x': 1, 'y': 2}

try:

    value = dict['z']

except KeyError:

    print("Key error. Key for 'value' doesn't exist")

# TypeError

try:

    result = "10" + 5

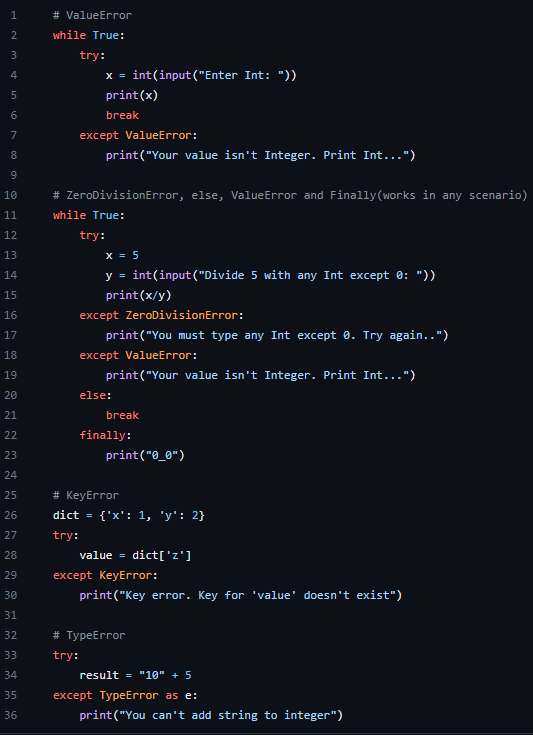
except TypeError as e:

    print("You can't add string to integer")

Посилання на github:

[GitHub](https://github.com/7878seveneight/TP-KB-222-Petro-Ratozii/blob/main/topic_04/task_03.py)

Знімок екрану з посилання на github:



Завдання №5.1

Гра з комп’ютером: камінь, ножиці, папір. Програма виконує запит від користувача на введення одного із значень ["rock", "scissor", "paper"]. Наступним кроком, використовуючи модуль random, програма у випадковому порядку вибирає одне із значень ["rock", "scissor", "paper"]. В залежності від умови, що камінь перемагає ножиці, ножиці перемагають папір, а папір перемагає камінь визначити переможця.

Хід виконання завдання:

1. Додав необхідну для виконання завдання бібліотеку random та бібліотеку cowsay для більшого інтерактиву, краще грати з коровою ніж з якимось   
   “комп’ютером”.
2. Реалізував запити до користувача та випадковий вибір “корови”.
3. Написав логіку програми через конструкцію match-case, тому-що гадаю що вона найпродуктивніша та найлегша ніж інші розглянуті мною конструкції для ціїє програми.
4. Написав репліки “корови” в результаті кожного можливого випадку.

Текст програми:

import random

import cowsay

cowsay.cow("Hi, let's play Rock-paper-scissors")

option = ["rock", "paper", "scissors"]

while (userChoice := input("(Choose between rock, paper or scissors)\n\t").strip().lower()) not in option:

    print("Invalid choice. Choose between rock, paper or scissors!")

cowChoice = random.choice(option)

cowsay.cow(f"{cowChoice}!!!")

match userChoice, cowChoice:

    case ('rock', 'paper'):

        print("Cow wins!")

        cowsay.cow("I WIN! MOOO!!!")

    case ('rock', 'scissors'):

        print("You win!")

        cowsay.trex("NOOO! How could I lose... I'm filled with anger and now I've become a dinosaur! I WILL EAT YOU")

    case ('rock', 'rock'):

        print("Game is tied.")

        cowsay.cow("Oh... It's tie.")

    case ('paper', 'rock'):

        print("You win!")

        cowsay.trex("NOOO! How could I lose... I'm filled with anger and now I've become a dinosaur! I WILL EAT YOU")

    case ('paper', 'paper'):

        print("Game is tied.")

        cowsay.cow("Oh... It's tie.")

    case ('paper', 'scissors'):

        print("Cow wins!")

        cowsay.cow("I WIN! MOOO!!!")

    case ('scissors', 'rock'):

        print("Cow wins!")

        cowsay.cow("I WIN! MOOO!!!")

    case ('scissors', 'paper'):

        print("You win!")

        cowsay.trex("NOOO! How could I lose... I'm filled with anger and now I've become a dinosaur! I WILL EAT YOU")

    case ('scissors', 'scissors'):

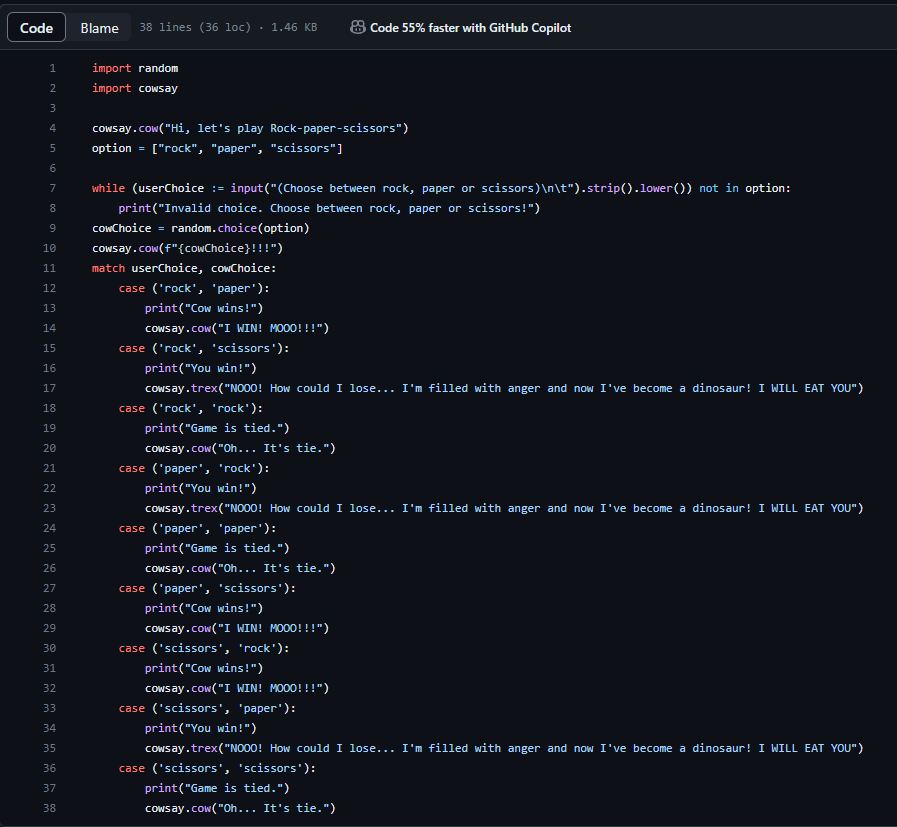
        print("Game is tied.")

        cowsay.cow("Oh... It's tie.")

Посилання на github:

[https://github.com/7878seveneight/](https://github.com/7878seveneight/TP-KB-222-Petro-Ratozii/blob/main/topic_05/task1.py)

Знімок екрану з посилання на github:



Завдання №5.2

Програма 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:

1. Реалізував програму конвертування іноземної валюти в української гривні, додаткого реалізував конвертування української гривні в іноземну валюту.
2. Додав модуль requests що надає можливість виконувати запити до сторонніх сервісів, в нашому випадку API НБУ.
3. Написав функцію пошуку необхідної валюти, щоб не копіювати і не вставляти код двічі.
4. С такою ж самою логікою написав функцію getFloatAmount().
5. Реалізував вибір між конвертуванням іноземної валюти в української гривні, та конвертуванням української гривні в іноземну валюту. Додав можливість виходу з програми.
6. Написав математичні дії для конвертування і зворотного конвертування.

Текст програми:

import requests

def listAndState():

    list = {elem["cc"]: elem["rate"] for elem in data}

    print(f"Current state of UAH to {currency} is {list[currency]}")

    return list

def getFloatAmount():

    while True:

        try:

            amount = float(input("Amount to calculate: "))

            return amount

        except ValueError:

            print("Use numbers for amount.")

response = requests.get("https://bank.gov.ua/NBUStatService/v1/statdirectory/exchangenew?json")

data = response.json()

option = ["USD", "EUR", "PLN"]

while True:

    operation = input("If you want to convert UAH to USD, EUR, PLN - type '1'\nIf you want to convert USD, EUR, PLN to UAH - type '2'\n  If you want to quit - type 'Quit'\n\tOption: ")

    if operation == '1':

        while (currency := input("Enter the currency you want to convert (USD, EUR, PLN): ").upper()) not in option:

            print("Invalid currency. Try USD, EUR or PLN")

        list = listAndState()

        amount = getFloatAmount()

        convert = amount \* list[currency]

        print(f"\t{amount} {currency} \n\t{convert} UAH")

    elif operation == '2':

        while (currency := input("Enter the currency you want to be converted (USD, EUR, PLN): ").upper()) not in option:

            print("Invalid currency. Try USD, EUR or PLN")

        list = listAndState()

        amount = getFloatAmount()

        convert = amount / list[currency]

        print(f"\t{amount} UAH \n\t{convert} {currency}")

    elif operation == "Quit" or operation == "quit":

        print("Quiting the program...")

        break

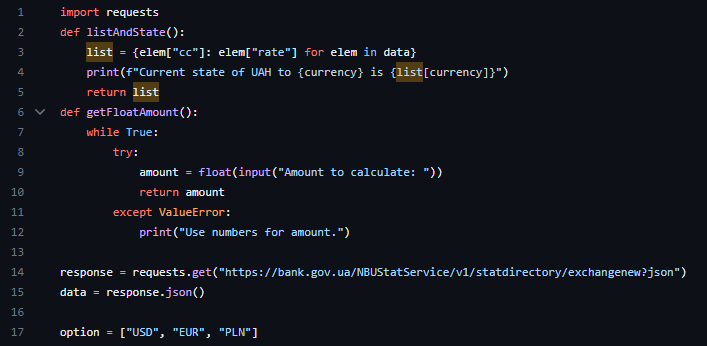
    else:

        print("If you want to convert UAH to USD, EUR, PLN - type '1'\nIf you want to convert USD, EUR, PLN to UAH - type '2'\n\tOption: ")

Посилання на github:

[https://github.com/7878seveneight/](https://github.com/7878seveneight/TP-KB-222-Petro-Ratozii/blob/main/topic_05/task2.py)

Знімок екрану з посилання на github:







Завдання №5.3

Використання модулів для програми калькулятор. Функції додавання, віднімання, множення та ділення перенести в файл functions.py. Функції запиту на введення даних для операцій та самих операцій перемістити в файл operations.py. Програму калькулятор реалізувати в файлі calc.py, до якого підключають файл functions.py та operations.py.

Хід виконання завдання:

1. Переніс функції додавання, віднімання, множення та ділення в файл functions.py
2. Функції запиту на введення даних для операцій та самих операцій перемістив у файл operations.py
3. Програму калькулятор реалізував у файлі calc.py, до якого підключив файл functions.py та operations.py.

Текст програм:

calc.py:

from functions import \*

from operations import \*

while True:

    operation = getOperation(operations)

    if operation == "Quit" or operation == "quit":

        print("Quiting the program...")

        break

    else:

        a = getFloatValue("Your first num: ")

        b = getFloatValue("Your second num: ")

        match(operation):

            case "+":

                print("Answer is:", plus(a, b))

            case "-":

                print("Answer is:", minus(a, b))

            case "\*":

                print("Answer is:", multiply(a, b))

            case "/":

                print("Answer is:", divide(a, b))

functions.py:

def plus (a, b):

    return a + b

def minus (a, b):

    return a - b

def multiply (a, b):

    return a \* b

def divide (a, b):

    try:

        return a / b

    except ZeroDivisionError:

        print("You can't divide a number by zero. Try againg...")

operations.py:

def getFloatValue(promt: str):

    while True:

        try:

            return float(input(promt))

        except ValueError:

            print("Value is not integer or float. Try againg...")

def getOperation(operations):

    while True:

        operation = input("If you want to quit the program - type \"Quit\" \nWhat's your operation (+, -, \*, /): ")

        if operation in operations:

            return operation

        else:

            print("You did something wrong, write your operation (+, -, \*, /)")

operations = ['Quit', 'quit', '+', '-', '\*', '/']

Посилання на github:

[https://github.com/7878seveneight/](https://github.com/7878seveneight/TP-KB-222-Petro-Ratozii/blob/main/topic_05/calc.py)

Знімок екрану з посилання на github:



Завдання №6.1

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:

1. В файл operations.py імпортував два модулі – datetime, os.

datetime – модуль який дозволяє працювати з датою і часом в Python.

os - модуль який дозволяє взаємодіяти з операційною системою, наприклад: створення, переміщення або видалення файлів та директорій(тек).

1. Написав функцію логування яка приймає 4 параметра, а саме - operation, a, b, result. Виводить в текстовий докумет log.txt дату та час, весь вираз, кожну змінну окремо. Це дозволяє відстежувати результати підрахунків.
2. Написав функцію логування у випадку коли користувач закінчив процес програми. Виводить в текстовий докумет log.txt дату та час виходу з програми.

Текст програм:

operations.py:

import datetime

import os

logFile = "log.txt"

dir = os.path.dirname(os.path.abspath(\_\_file\_\_))

path = os.path.join(dir, "log.txt")

def getFloatValue(promt: str):

    while True:

        try:

            return float(input(promt))

        except ValueError:

            print("Value is not integer or float. Try againg...")

def getOperation(operations):

    while True:

        operation = input("If you want to quit the program - type \"Quit\" \nWhat's your operation (+, -, \*, /): ")

        if operation in operations:

            return operation

        else:

            print("You did something wrong, write your operation (+, -, \*, /)")

def logs(operation, a, b, result):

    currentDate = datetime.datetime.now()

    date = currentDate.strftime("%Y-%m-%d %H:%M:%S")

    log = f"\n\n\t{date} \nInstance: {a} {operation} {b} = {result} \nOperation: {operation}, First num: {a}, Second num: {b}, Result or operation: {result}"

    with open(path, "a") as file:

        file.write(log)

def logsForExit():

    currentDate = datetime.datetime.now()

    date = currentDate.strftime("%Y-%m-%d %H:%M:%S")

    with open(path, "a") as file:

        file.write(f"\n\n\t{date}\nQuit from program.")

operations = ['Quit', 'quit', '+', '-', '\*', '/']

calc.py:

from functions import \*

from operations import \*

while True:

    operation = getOperation(operations)

    if operation == "Quit" or operation == "quit":

        print("Quiting the program...")

        logsForExit()

        break

    else:

        a = getFloatValue("Your first num: ")

        b = getFloatValue("Your second num: ")

        match(operation):

            case "+":

                print("Answer is:", result := plus(a, b))

            case "-":

                print("Answer is:", result := minus(a, b))

            case "\*":

                print("Answer is:", result := multiply(a, b))

            case "/":

                print("Answer is:", result := divide(a, b))

        logs(operation, a, b, result)

log.txt:

2023-10-26 15:11:00

Instance: 2.0 + 5.0 = 7.0

Operation: +, First num: 2.0, Second num: 5.0, Result or operation: 7.0

2023-10-26 15:11:06

Instance: 5.0 - 10.0 = -5.0

Operation: -, First num: 5.0, Second num: 10.0, Result or operation: -5.0

2023-10-26 15:11:15

Instance: 2.0 \* 4.0 = 8.0

Operation: \*, First num: 2.0, Second num: 4.0, Result or operation: 8.0

2023-10-26 15:11:26

Instance: 1.0 / 0.5 = 2.0

Operation: /, First num: 1.0, Second num: 0.5, Result or operation: 2.0

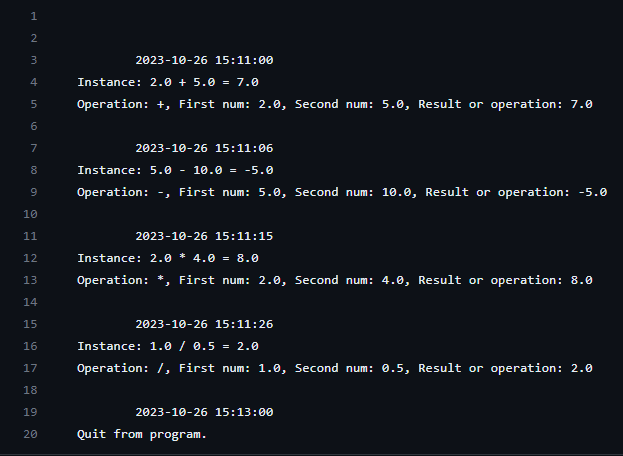
2023-10-26 15:13:00

Quit from program.

Посилання на github:

[https://github.com/7878seveneight/](https://github.com/7878seveneight/TP-KB-222-Petro-Ratozii/tree/main/topic_06/task1)

Знімок екрану з посилання на github:

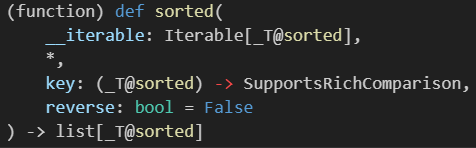
 



Завдання №6.2

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:

1. Створив не відсортований список, елементами якого є словники з двома параметрами (ім’я та оцінка).
2. Зробив lambda функцію, що повертає ім’я або оцінку із елемента словника, щоб далі застосувати як другий елемент для функції sorted().
3. Додатково зробив вибір між типом сортування – зростання чи спадання. Функція sorted() дозволяє третім параметром реалізувати таку ідею. 
4. Реалізував сортування за ім’ям та сортування за оцінкою та вивід цих даних у термінал.

Текст програми:

students = [{'name': 'Petr', 'grade': 110},

            {'name': 'Petro', 'grade': 65},

            {'name': 'Ihor', 'grade': 100},

            {'name': 'NotPetr', 'grade': 70}]

getName = lambda student: student['name']

getGrade = lambda student: int(student['grade'])

while True:

    whichSort = input("Sort by descending or ascending: ")

    if whichSort == "ascending":

        sortType = True

        break

    elif whichSort == "descending":

        sortType = False

        break

    else:

        print("Invalid option. Try againg...")

sortedName = sorted(students, key=getName, reverse=sortType)

print("\n Сортування за ім'ям:")

for student in sortedName:

    print(f"Name: {student['name']}, Grade: {student['grade']}")

sortedGrade = sorted(students, key=getGrade, reverse=sortType)

print("\n Сортування за оцінкою:")

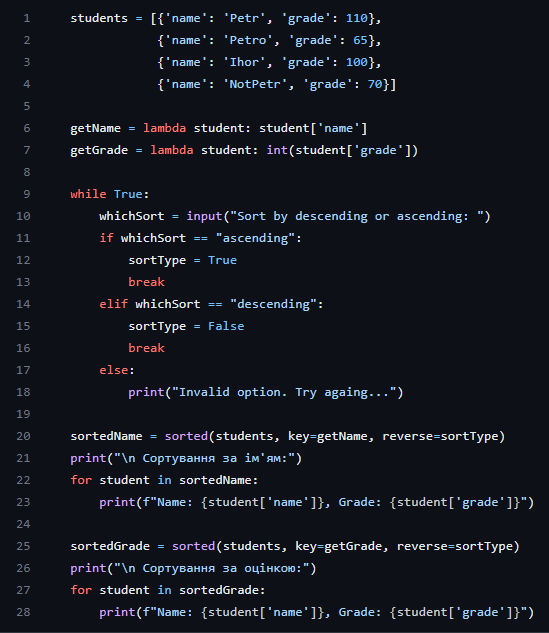
for student in sortedGrade:

    print(f"Name: {student['name']}, Grade: {student['grade']}")

Посилання на github:

[https://github.com/7878seveneight/](https://github.com/7878seveneight/TP-KB-222-Petro-Ratozii/blob/main/topic_06/task2.py)

Знімок екрану з посилання на github:



Завдання №7.1

Ознайомитись з документацією що описує можливості використання класів у мові Python <https://docs.python.org/3/tutorial/classes.html>

Хід виконання завдання:

1. Ознайомився з документацією що описує можливості використання класів у мові Python.
2. Подивився кілька відео для більшого розуміння данної документації
3. Прийнявся виконувати наступні завдання

Завдання №7.2

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

Хід виконання завдання:

1. Після ознайомлення з цією темою обдумав завдання.
2. Реалізував написання та виведення списку студентів, який потім сортирується по оцінці студента.
3. Зробив завдання в стилі ООП.

Текст програми:

class Student():

    def \_\_init\_\_(self, name, age, grade):

        self.name = name

        self.age = age

        self.grade = grade

    def \_\_str\_\_(self):

        return f"Student's name: {self.name} \nStudent's age: {self.age} \nStudent's grade: {self.grade}"

    @property

    def name(self):

        return self.\_name

    @name.setter

    def name(self, name):

        self.\_name = name

    @property

    def age(self):

        return self.\_age

    @age.setter

    def age(self, age):

        if age < 0:

            raise ValueError("Student's age must be greater than 0.")

        self.\_age = age

    @property

    def grade(self):

        return self.\_grade

    @grade.setter

    def grade(self, grade):

        if grade < 0:

            raise ValueError("Student's grade must be greater than 0.")

        self.\_grade = grade

def getStudent():

    try:

        name = input("Student's name: ")

        age = int(input("Student's age: "))

        grade = int(input("Student's grade: "))

        student = Student(name, age, grade)

    except Exception as e:

        print(e)

        return getStudent()

    else:

        return student

def main():

    amount = int(input('Amount of your students: '))

    students = []

    for \_ in range(amount):

        students.append(getStudent())

        print("\n")

    for student in sorted(students, key=lambda s: s.grade):

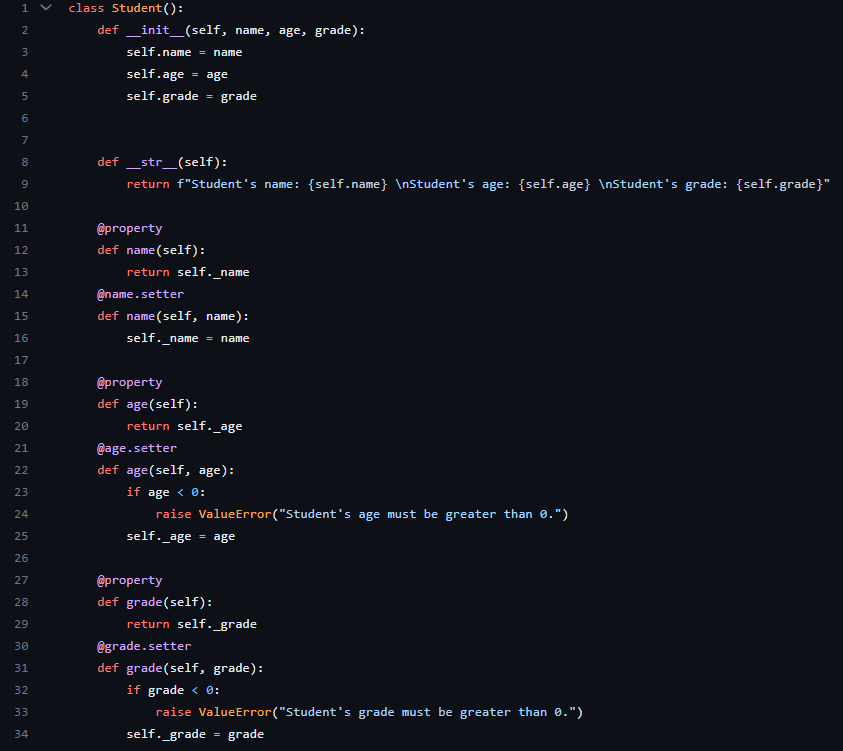
        print(f"\n{student}")

main()

Посилання на github:

[https://github.com/7878seveneight/](https://github.com/7878seveneight/TP-KB-222-Petro-Ratozii/blob/main/topic_07/task2.py)

Знімок екрану з посилання на github:





Завдання №7.3

Розробити клас Student атрибутами якого э два параметра name та age. Створити список елементами якого є об'єкти класу Student. Написати цикл який виводить на екран елементи списку у відсортованому порядку. Для сортування використати стандартну функцію sorted. Функція sorted має використовувати lambda функцію для визначення ключа сортування.

Хід виконання завдання:

1. Створив клас Студент і перейдав у нього параметри ім’я та віку.
2. Створив список студентів використовуючи клас Студент, щоб передавати в нього данні.
3. Додав вибір сортування по спаданню чи зростанню віку.
4. Створив змінну для сортованого списку та виконав функцію сортування.
5. Вивів список через цикл for.

Текст програми:

class Student:

    def \_\_init\_\_(self, name, age):

        self.name = name

        self.age = age

students = [

    Student("Petr", 19),

    Student("Petro", 20),

    Student("Ihor", 27),

    Student("NotPetr", 18),

    Student("Ivan", 17),

]

while True:

    whichSort = input("Sort by descending or ascending: ")

    if whichSort == "ascending":

        sortType = True

        break

    elif whichSort == "descending":

        sortType = False

        break

    else:

        print("Invalid option. Try againg...")

sortStudents = sorted(students, key=lambda student: student.age, reverse=sortType)

for student in sortStudents:

    print(f"Name: {student.name}, Age: {student.age}")

Посилання на github:

[https://github.com/7878seveneight/](https://github.com/7878seveneight/TP-KB-222-Petro-Ratozii/blob/main/topic_07/task3.py)

Знімок екрану з посилання на github:



Завдання №7.4

Використовуючи принципи ООП переписати програму Калькулятор. Завдання має бути виконано використовуючи модульний підхід.

Хід виконання завдання:

1. Обдумав де саме в калькуляторі використовувати принципи ООП.
2. Переписав файл functions.py за принципами ООП.
3. Переписав файл operations.py за принципами ООП.
4. Не додавав ніякого нового функціоналу, тільки удосконалив той що був.

Текст програм:

calc.py

from functions import \*

from operations import \*

while True:

    oper = CalcOperatoins()

    operation = oper.getOperation(oper.operations)

    if operation == "Quit" or operation == "quit":

        print("Quiting the program...")

        oper.logsForExit()

        break

    else:

        func = CalcFunctions()

        oper = CalcOperatoins()

        a = func.getFloatValue("Your first num: ")

        b = func.getFloatValue("Your second num: ")

        match(operation):

            case "+":

                print("Answer is:", result := func.plus(a, b))

            case "-":

                print("Answer is:", result := func.minus(a, b))

            case "\*":

                print("Answer is:", result := func.multiply(a, b))

            case "/":

                print("Answer is:", result := func.divide(a, b))

        oper.logs(operation, a, b, result)

functions.py

class CalcFunctions:

    def getFloatValue(self, prompt):

        while True:

            try:

                return float(input(prompt))

            except ValueError:

                print("Value is not integer or float. Try againg...")

    def plus (self, a, b):

        return a + b

    def minus (self, a, b):

        return a - b

    def multiply (self, a, b):

        return a \* b

    def divide (self, a, b):

        try:

            return a / b

        except ZeroDivisionError:

            print("You can't divide a number by zero. Try againg...")

operations.py

import datetime

import os

class CalcOperatoins:

    def \_\_init\_\_(self):

        self.operations = ['+', '-', '\*', '/', 'Quit', 'quit']

    def getOperation(self, operations):

        while True:

            operation = input("If you want to quit the program - type \"Quit\" \nWhat's your operation (+, -, \*, /): ")

            if operation in operations:

                return operation

            else:

                print("You did something wrong, write your operation (+, -, \*, /)")

    def logs(self, operation, a, b, result):

        dir = os.path.dirname(os.path.abspath(\_\_file\_\_))

        path = os.path.join(dir, "log.txt")

        currentDate = datetime.datetime.now()

        date = currentDate.strftime("%Y-%m-%d %H:%M:%S")

        log = f"\n\n\t{date} \nInstance: {a} {operation} {b} = {result} \nOperation: {operation}, First num: {a}, Second num: {b}, Result or operation: {result}"

        with open(path, "a") as file:

            file.write(log)

    def logsForExit(self):

        dir = os.path.dirname(os.path.abspath(\_\_file\_\_))

        path = os.path.join(dir, "log.txt")

        currentDate = datetime.datetime.now()

        date = currentDate.strftime("%Y-%m-%d %H:%M:%S")

        with open(path, "a") as file:

            file.write(f"\n\n\t{date}\nQuit from program.")

Посилання на github:

[https://github.com/7878seveneight/](https://github.com/7878seveneight/TP-KB-222-Petro-Ratozii/tree/main/topic_07/task4)

Знімок екрану з посилання на github:



