**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:

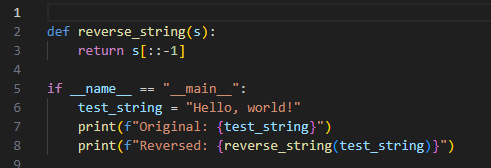
**Перетворення рядка**

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.

Текст програми:



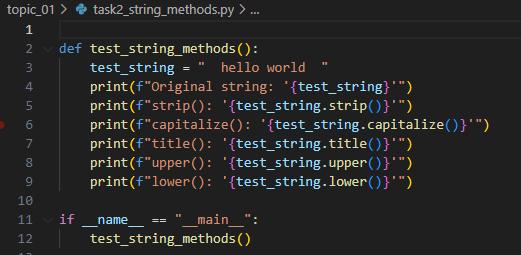
**Виконати тестування функцій, що працюють з рядками**

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:



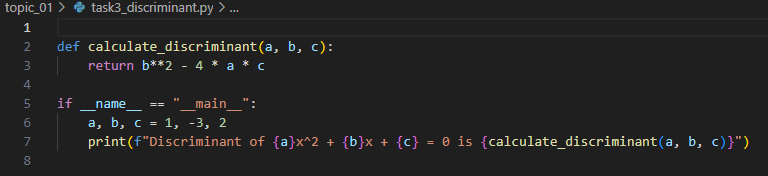
**Дискримінант**

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:

Звіт до Теми №2

Умовні переходи

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

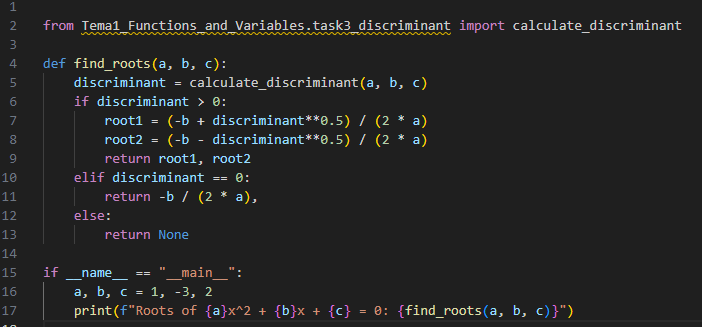
**Коренів квадратного рівняння**

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

Хід виконання завдання:

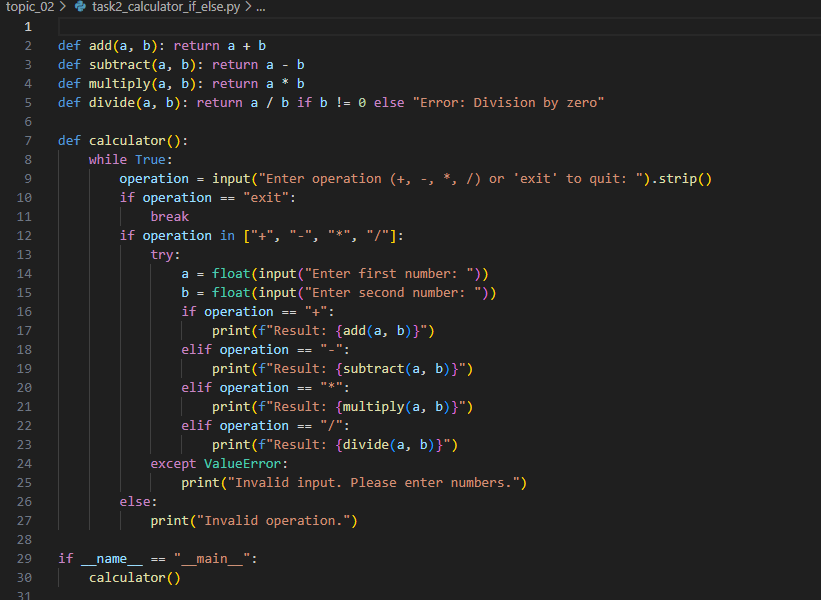
Використовуючи функцію розрахування дискримінанту з минулої теми, додав функцію обрахування коренів та перевірку дійсності вводу.

Текст програми:



**Калькулятор на if else конструкції**

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



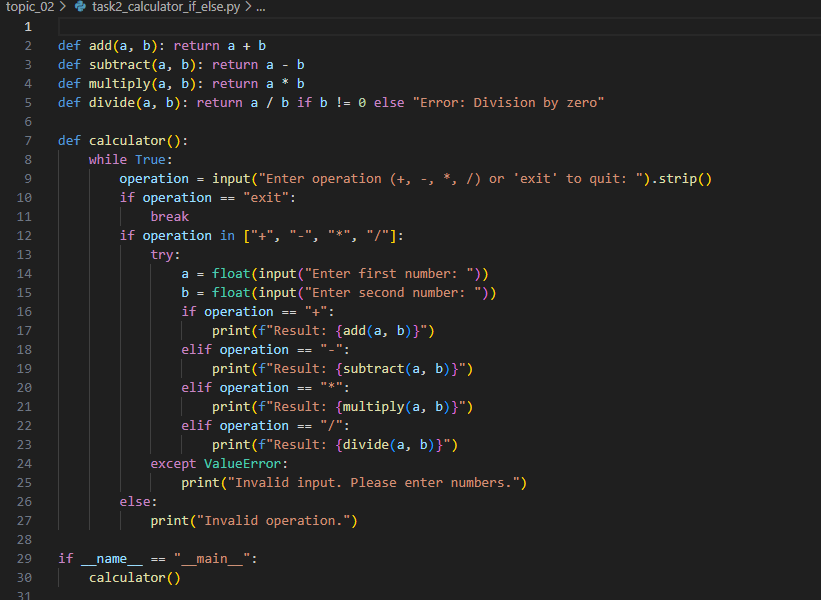
**Калькулятор на match конструкції**

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:



Звіт до Теми №3

Цикли

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

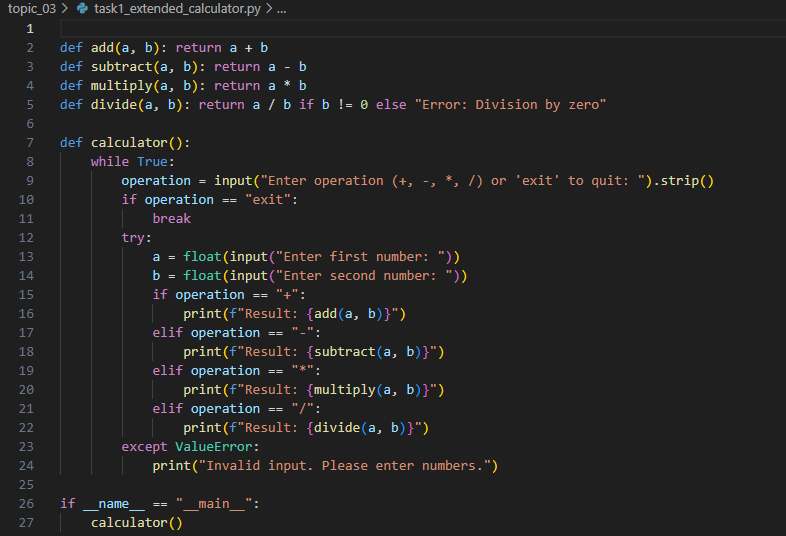
**Калькулятор з постійними запитами**

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:



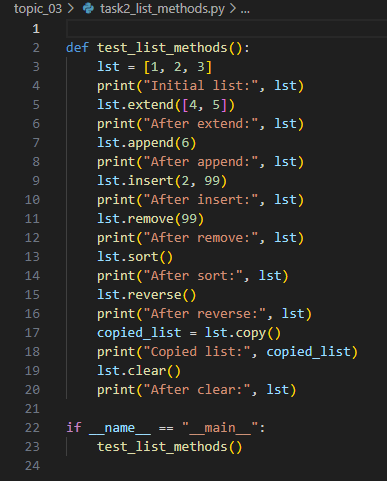
**Тестування функцій списків**

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:



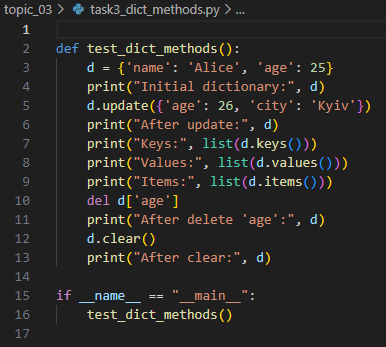
**Тестування функцій словників**

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:



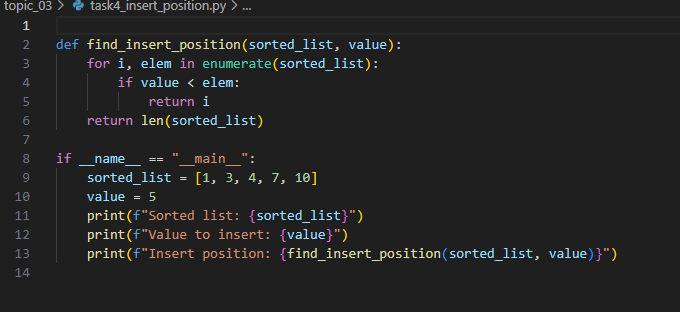
**Функція пошуку позиції для вставки**

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:



Звіт до Теми №4

Виняткові ситуації

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

**Модифікація програми калькулятор**

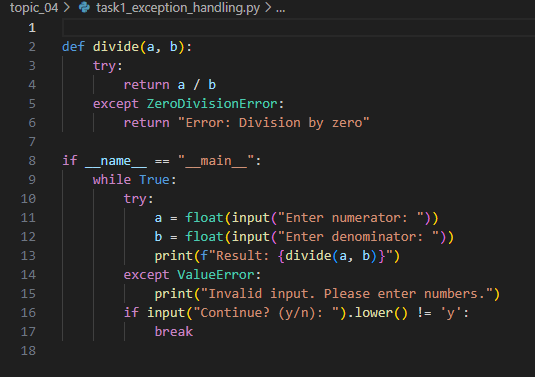
Розширити програму калькулятор функцією запитів даних для виконання операцій від користувача, що обробляє виняткові ситуації.

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:



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:



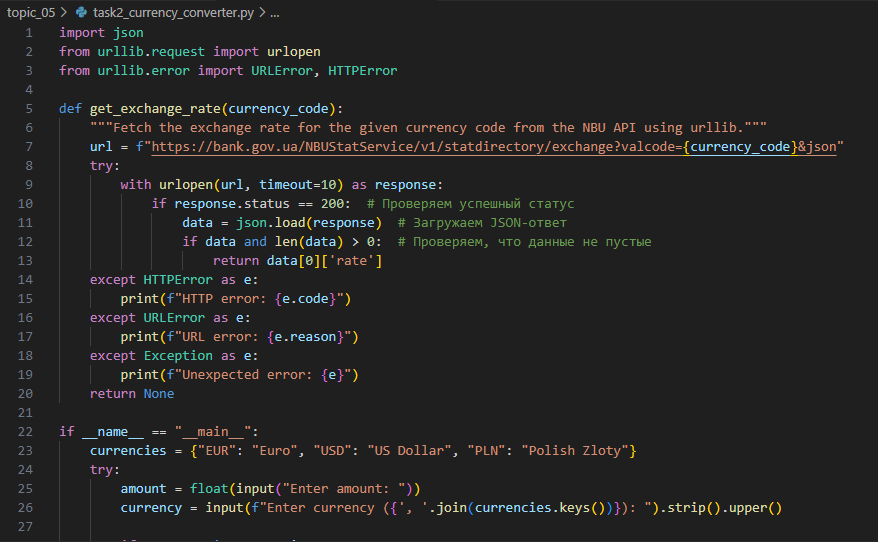
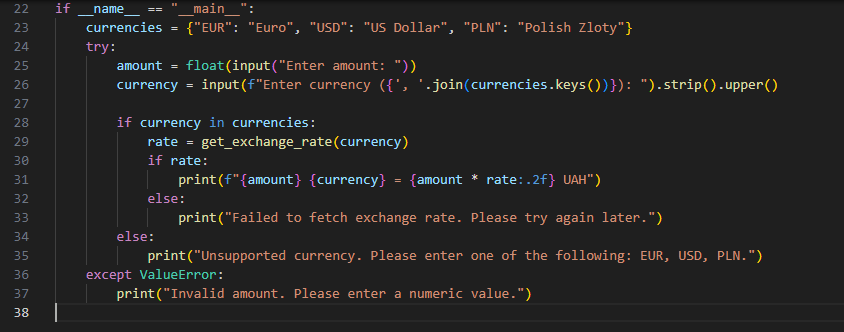
**Конвертування іноземної валюти**

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:

Звіт до Теми №6

Робота з файлами

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

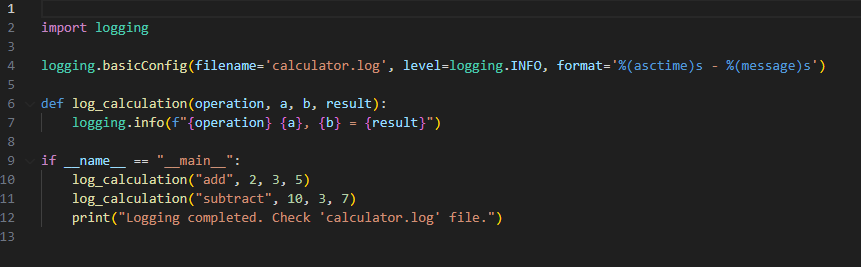
**Механізм логування**

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:

Текст програми:



Звіт до Теми №7

Об’єктно-орієнтоване програмування

Під час виконання практичного завдання до Теми №7 було надано варіанти рішення до наступних задач:

**Методики класу**

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

Хід виконання завдання:

Створив найпростіші класи Person та Unit, та задав їм атрибути. Для першого класу не задавав метод str, та порівняв їх відображення.

**Калькулятор**

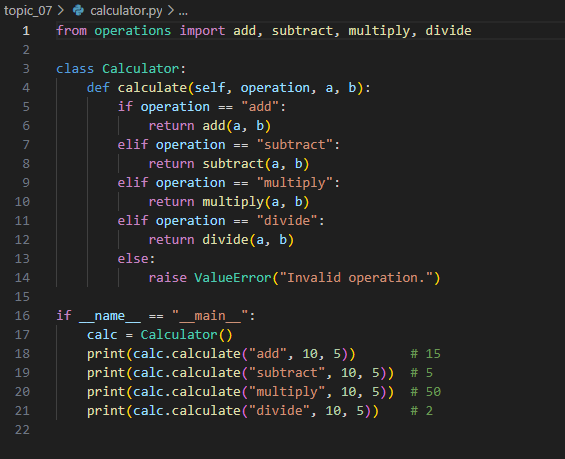
Використовуючи принципи ООП переписати програму Калькулятор. Завдання має бути виконано використовуючи модульний підхід.

Хід виконання завдання:

Список основних змін:

* Основна програма очищена від функцій, залишено лише перевірку виконання основної програми та імпорт функцій.
* Створено файл calculator.py, куди перенесені функції виводу інформаційних повідомлень про список операцій, та запит на вибір операції.
* Файл operations.py збережено структуру, але функції об’єднано в клас.

Текст програми:



Посилання на github:

<https://github.com/09premium/09premium-TP-KB-231-Oleksandr-Tymoshenko>

Знімок екрану з посилання на github:

