**Звіт про виконання завдань для практики з курсу «Технології програмування»**

[1. Звіт до теми №1 2](#_Toc146747665)

[1.1. Обернення рядка. 2](#_Toc146747666)

[1.2. Тестування різних методів для роботи зі строками. 2](#_Toc146747667)

[1.3. Функція для знаходження дискримінанту. 4](#_Toc146747668)

[2. Звіт до теми №2 6](#_Toc146747669)

[2.1. Написати функцію для знаходження коренів рівняння, використовуючи умови. 6](#_Toc146747670)

[2.2. Написання калькулятора за допомогою умов if, elif. Написання калькулятора за допомогою оператора match. 8](#_Toc146747671)

# Звіт до теми №1

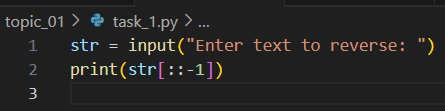
## Обернення рядка.

Завдання: Необхідно перевернути рядок **abcdefg123** так, щоб він був таким: **321gfedcba**.

Виконання завдання:

Створив змінну із запитом користувача, який запитує, яку строку обернути. Далі виводимо через print, у середині за допомогою оператора [::-1] я повернув обернений рядок. Перше та друга крапки в операторі [::-1] означають, що потрібно брати всі елементи від початку до кінця, а число «-1» означає, що відлік має починатися з кінця.

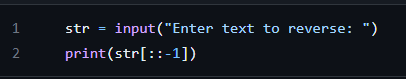
Код завдання:



GitHub:

Посилання на GitHub на цей файл: <https://github.com/danylosykylinda/TP-KB-221-Danylo-Sykylinda/blob/main/topic_01/task_1.py>

Скріншот коду з GitHub:



## Тестування різних методів для роботи зі строками.

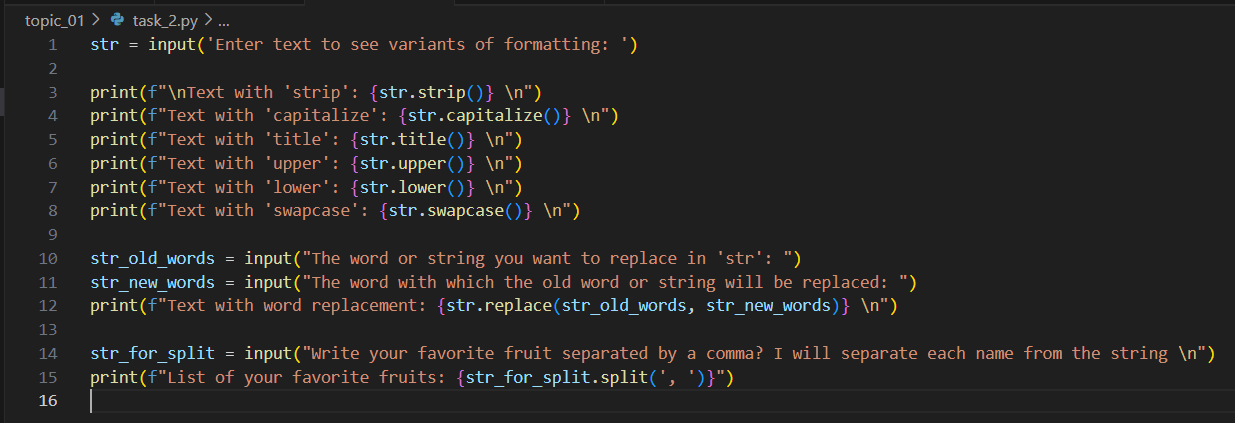
Завдання: Перевірити методи для роботи зі строками strip(), capitalize(), title(), upper(), lower(), swapcase(), replace, split().

Виконання завдання:

Створив змінну із запитом про текст у користувача, потім через функцію print виводжу відформатований текст різними методами(strip(), capitalize(), title(), upper(), lower(), swapcase()) і таким чином показую, як ці методи працюють. Далі для методу replace () я створив дві змінні з запитом до користувача. У першій я запитую, яке слово чи текст хоче замінити в строці, яка є в змінній **str**, а в другій – на яке слово чи текст буде замінений попереднє слово чи текст. Далі виводжу відформатований текст через print, у ньому всередині replace() заміняє старі слова або текст на нові слова або текст.

Далі я вирішив випробувати метод split(). Для цього створив змінну із запитом до користувача про його улюблені фрукти і попрохав перечислити його їх через кому. Потім через print виводжу список улюблених фруктів, список цей видає метод split(), за допомогою нього я виділив із строки назви фруктів, вказавши в split параметр ‘, ‘, тобто, шукати кому з пробілом. У підсумку цей метод повертає список елементів – улюблених фруктів користувача.

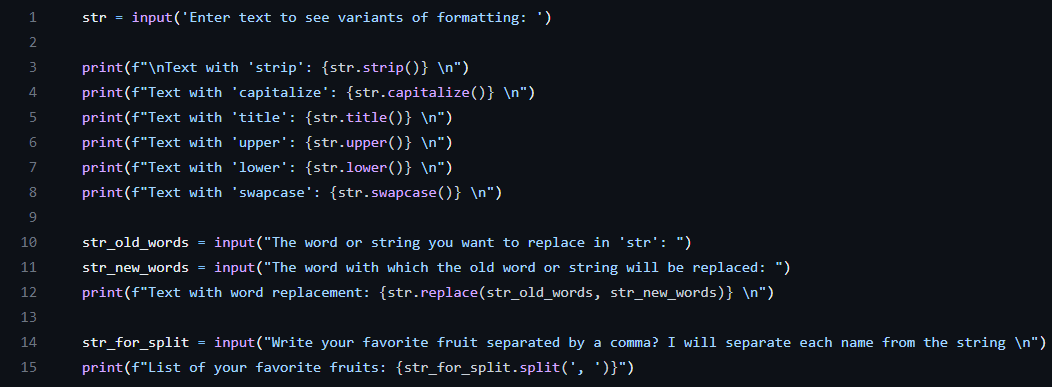
Код завдання:



GitHub:

Посилання на GitHub на цей файл: <https://github.com/danylosykylinda/TP-KB-221-Danylo-Sykylinda/blob/main/topic_01/task_2.py>

Скріншот коду з GitHub:



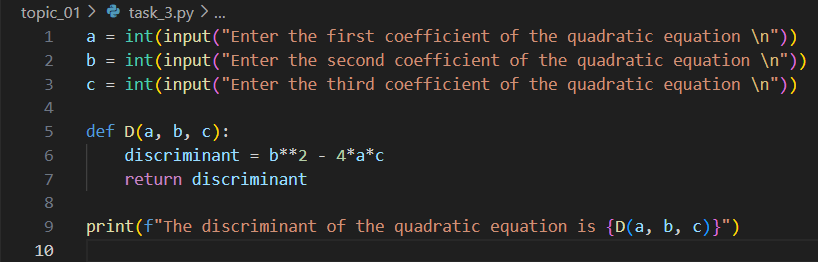
## Функція для знаходження дискримінанту.

Завдання: Потрібно написати функцію, яка приймає три параметри a, b, c – коєфіцієнти квадратного рівняння та повертає значення дискримінанту.

Виконання завдання:

Я створив три змінні із запитом до користувача, щоб отримати коєфіцієнти рівняння а, b, c. Далі створив функцію D, що приймає три аргументи a, b, c. У всередині функції створив змінну discriminant, якій присвоїв формулу b2-4ac. І через return повернув результат функції. Потім виводжу через print, при цьому всередині неї викликаю функцію D і передаю в якості параметрів змінні зі значеннями, що надав користувач. І тоді ми отримуємо результат.

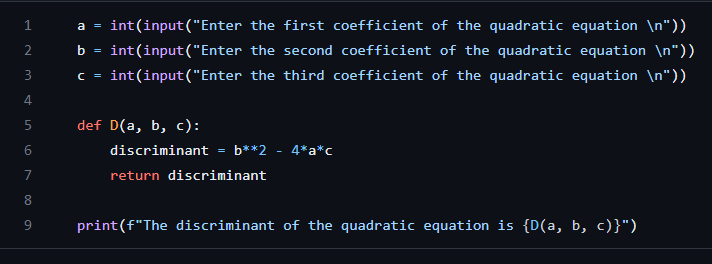
Код завдання:



GitHub:

Посилання на GitHub на цей файл: <https://github.com/danylosykylinda/TP-KB-221-Danylo-Sykylinda/blob/main/topic_01/task_3.py>

Скріншот коду з GitHub:



# Звіт до теми №2

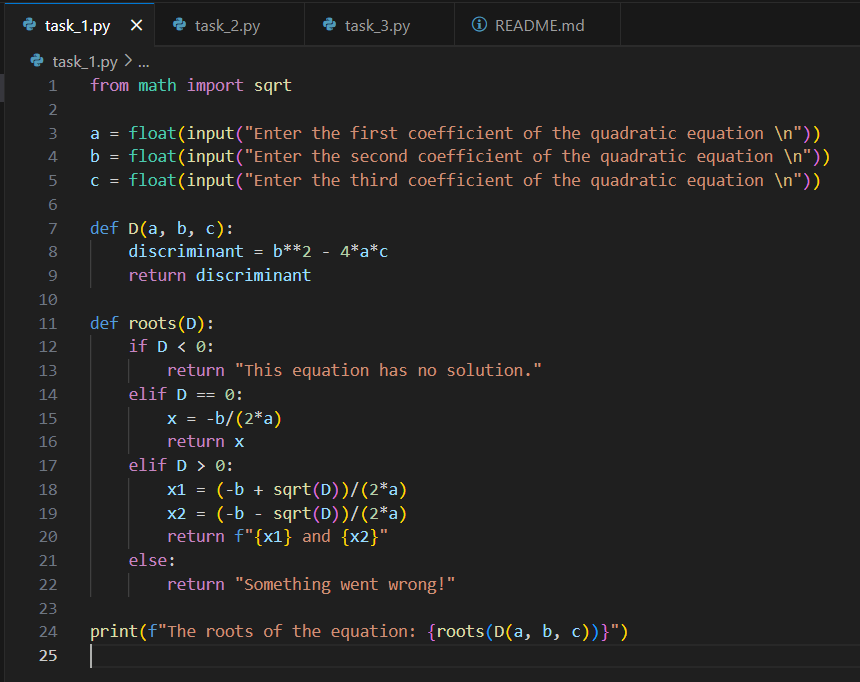
## Написати функцію для знаходження коренів рівняння, використовуючи умови.

Завдання: Необхідно написати функцію для знаходження коренів рівняння, використовуючи умови. Умови ми використовуємо для дискримінанту(далі буде показано як). Функцію знаходження дискримінанту ми візьмемо з розділу 1. завдання 1.3.

Виконання завдання:

Я імпортував з модулі math функцію sqrt. Далі переніс частину коду з розділу 1. завдання 1.3. Дещо переписав: отримання числа не в int, а вже у float для того, щоб можна було не тільки цілі числа використовувати, але й дробові. Потім написав функцію roots, яка отримує параметр – дискримінант. За допомогою умов if, elif прописав умови для різних випадків: якщо дискримінант менше нуля, повертаємо результат - стрічку, що рівняння немає коренів(стрічки 12-13), якщо дискримінант дорівнює нулю, то маємо один корінь(стрічки 14-16), обчислюємо його і виводим, якщо дискримінант більше нуля, то маємо два корені і обчислюємо і виводим. І за допомогою else, якщо піде щось не так, то виводили повідомлення про те, що щось не так. Далі через print виводимо результат, у print викликаємо функцію roots, у якості параметра передаємо функцію D з параметрами a, b, c.

Код завдання:

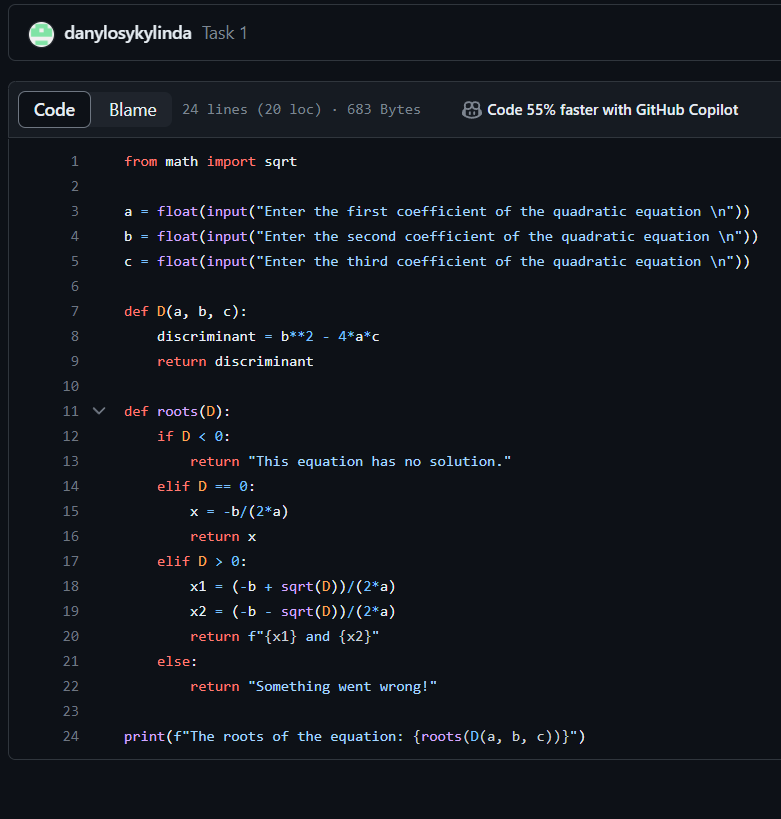


GitHub:

Посилання на GitHub на цей файл:

<https://github.com/danylosykylinda/TP-KB-221-Danylo-Sykylinda/blob/main/topic_02/task_1.py>

Скріншот коду з GitHub:

****

## Написання калькулятора за допомогою умов if, elif. Написання калькулятора за допомогою оператора match.

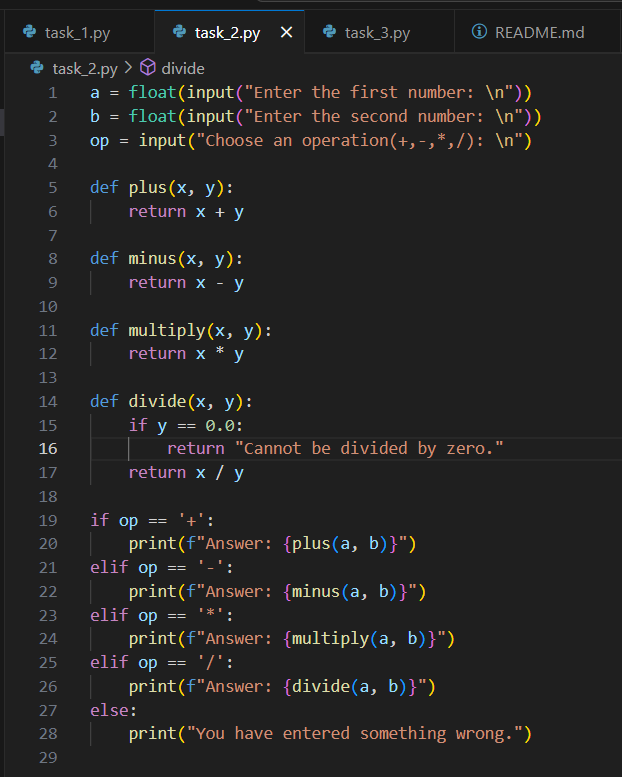
Завдання: Потрібно написати калькулятор, у якому ми запитуємо два числа і дію(+,-,/,\*) від користувача, при цьому коді використовуємо умови if, elif. Потім написати калькулятор, у коді якого ми використаємо оператор match.

Виконання завдання:

Частина 1.

Створив три змінні із запитом до користувача: a та b – числа, op – операція над цими числами. Потім написав функції, які виконують операції(plus – додавання, minus – віднімання, multiply – множення, divide - ділення). У функції divide ми ще умовою перевіряємо, чи другий аргумент є нулем, бо якщо другий дорівнює нулю, то не можна поділити, на нуль не ділиться. Потім написав умову, якщо у змінну користувач написав ‘+’, то ми виводимо результат в print, у ньому викликаємо функцію plus і передаємо два числа їй – a та b. Аналогічні дії робимо для інших операцій. У кінці дописуємо else з повідомленням, якщо користувач написав щось неправильно.

Код завдання:

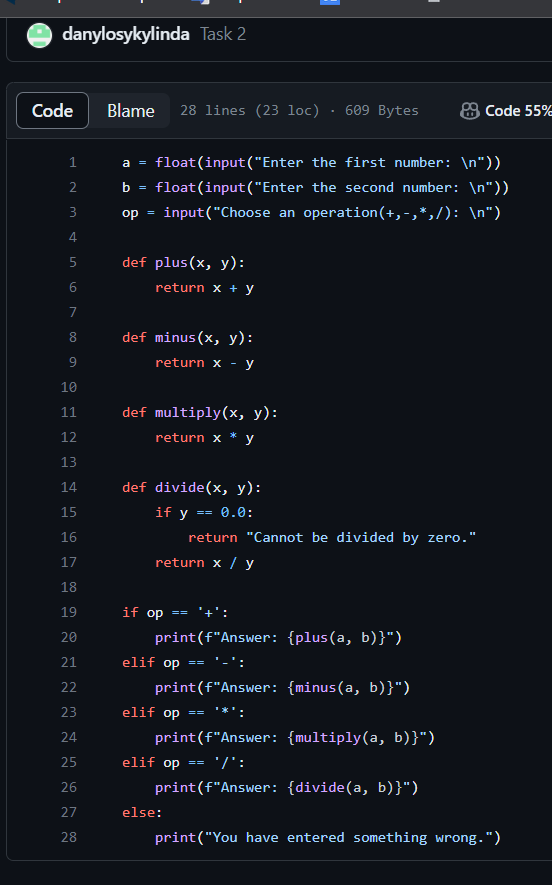


GitHub:

Посилання на GitHub на цей файл:

<https://github.com/danylosykylinda/TP-KB-221-Danylo-Sykylinda/blob/main/topic_02/task_2.py>

Скріншот коду з GitHub:

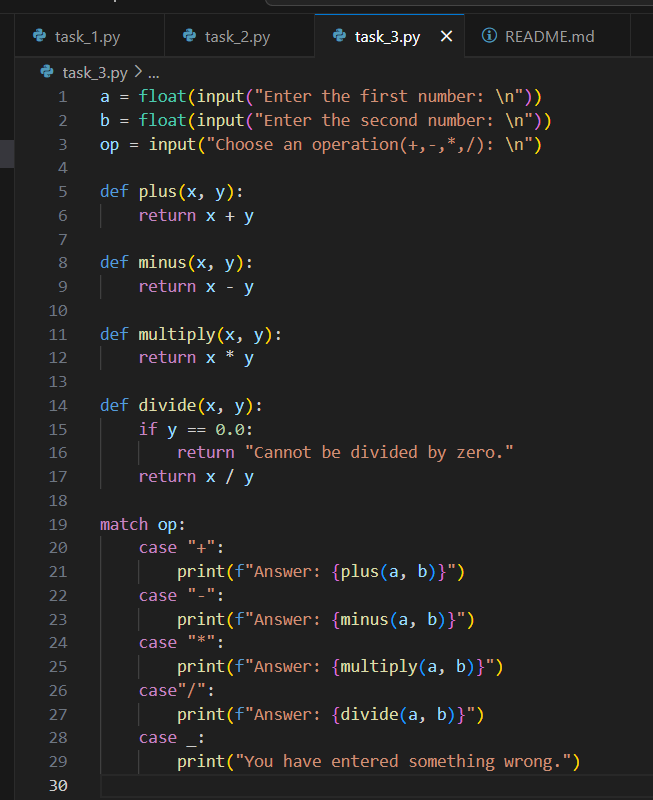
****

Виконання завдання:

Частина 2.

Стрічки 1-17 скопійовані з частини 1 цього завдання. Далі ми за допомогою оператора match прописуємо випадки(case) для (+, -, \*, /). І аналогічно, як частині 1 цього завдання, ми викликаємо функцію print, у ній викликаємо функцію plus і передаємо два аргументи a та b. Аналогічно з іншими математичними діями. У кінці прописуємо case \_: - це означає, що якщо будуть уведені інші символи відмінні, що зазначені в інших case, то реагувати певним чином, у даному case ми виводимо повідомлення, що щось пішло не так.

Код завдання:

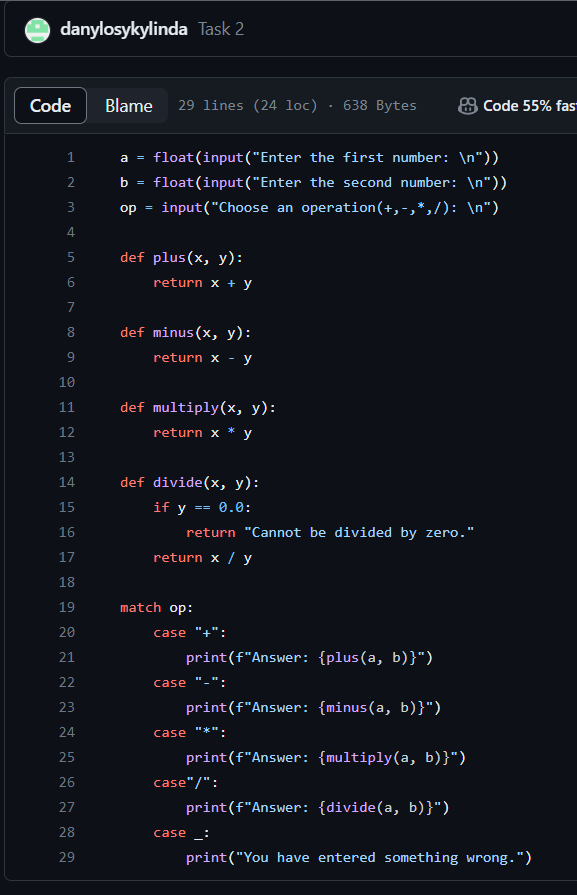


GitHub:

Посилання на GitHub на цей файл:

<https://github.com/danylosykylinda/TP-KB-221-Danylo-Sykylinda/blob/main/topic_02/task_3.py>

Скріншот коду з GitHub:

****