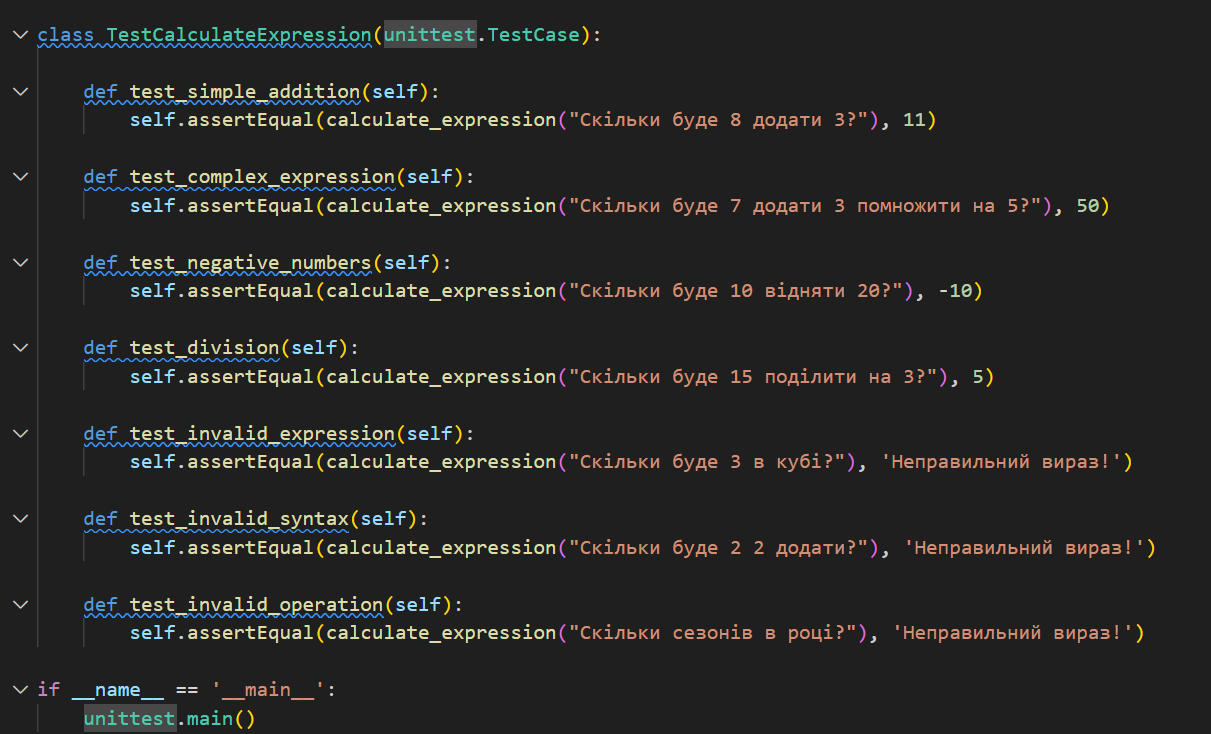
Оригінальний код:  
Chat GPT:

Промпт 1:  
 - \*скопійована задача\* This is a description of a task, I need to code in Python. Your task is to write unittest module to cover all the necessary functions and edgecases

-Запропоновані тести:

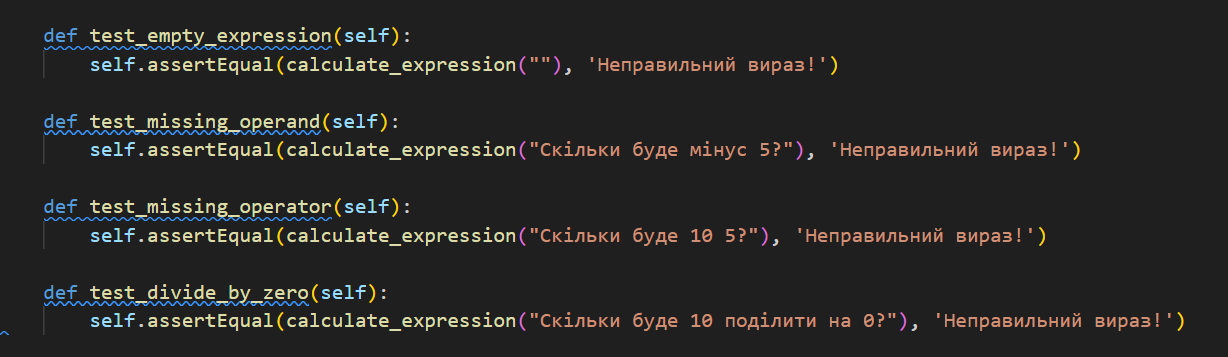


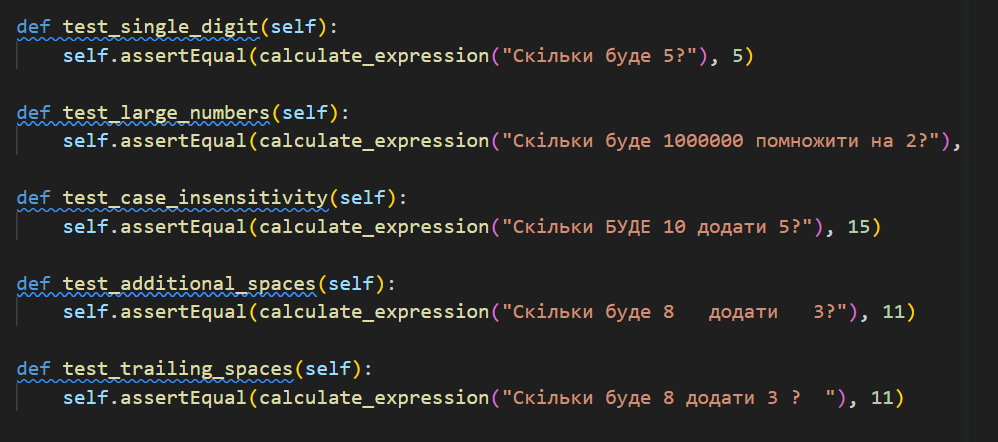
- Коментарі:

З першого промпту Chat GPT видав тільки ті приклади, які були запропоновані в умові. Тим не менш, синтаксис unittest був дотриманий, назви функцій добре пояснювати свої завдання і код не потребував допрацювань.

Промпт 2, 3:  
 - Now include more possible edge cases

-Додані тести:



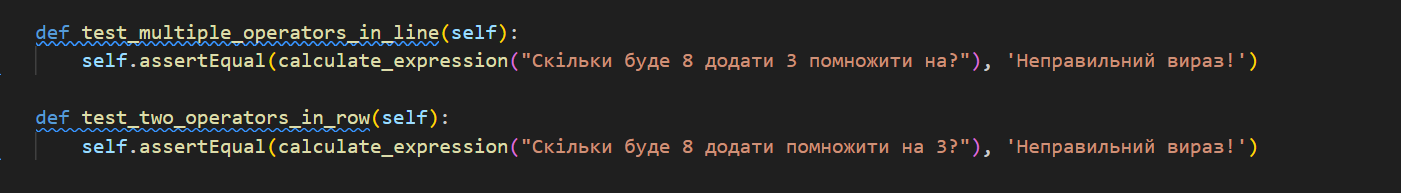


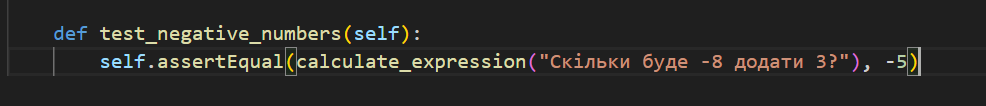
- Коментарі:

Тут ШІ додав більше своїх власних тестів: перевірки на відсутність деяких елементів, тестування на великих числах, ділення на 0. Це досить валідні еджкейси, які часто не враховуються при написанні коду. Але вже на 3 промпті, чат видав перевірку на впізнаваність регістру, чого не передбачав cms, тому довелося пізніше це прибрати.

Промпт 4, 5:

* You forgot to test an option where it`s digit, operator, operator, digit
* Now add a test with negative numbers
* Додані тести:



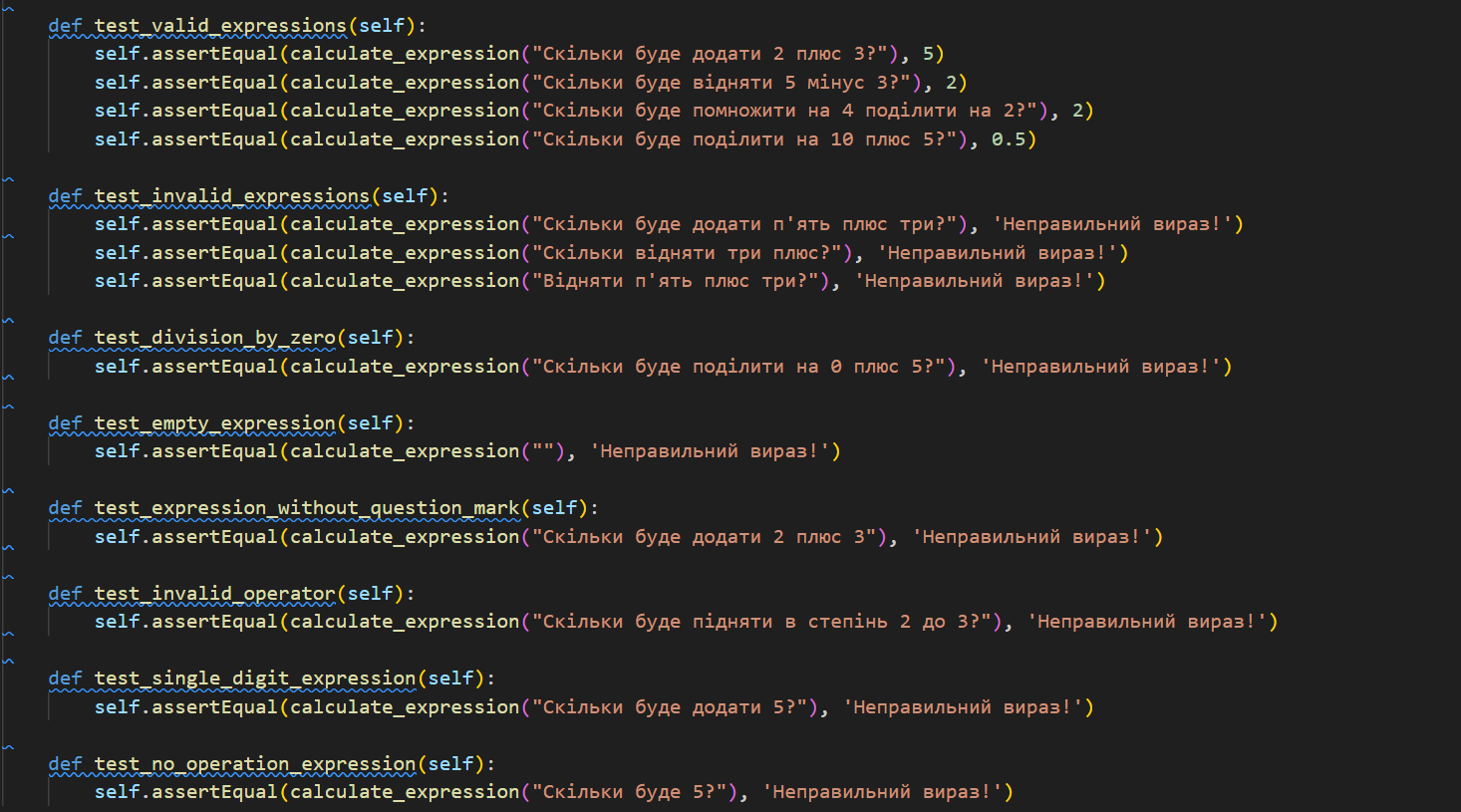


- Коментарі:

Після кількох повторних спроб змусити ШІ додати тест, де є 2 оператори підряд, довелося сказати йому про це напряму. Проте він додав ще один тест для перевірки виразу, який закінчується на оператор. З від’ємними числами така ж ситуація.

Промпт 6, 7:

* This is my code, write unittests to cover it. \*скопійований код\*
* Add more tests to make coverage 100%
* Додані тести:

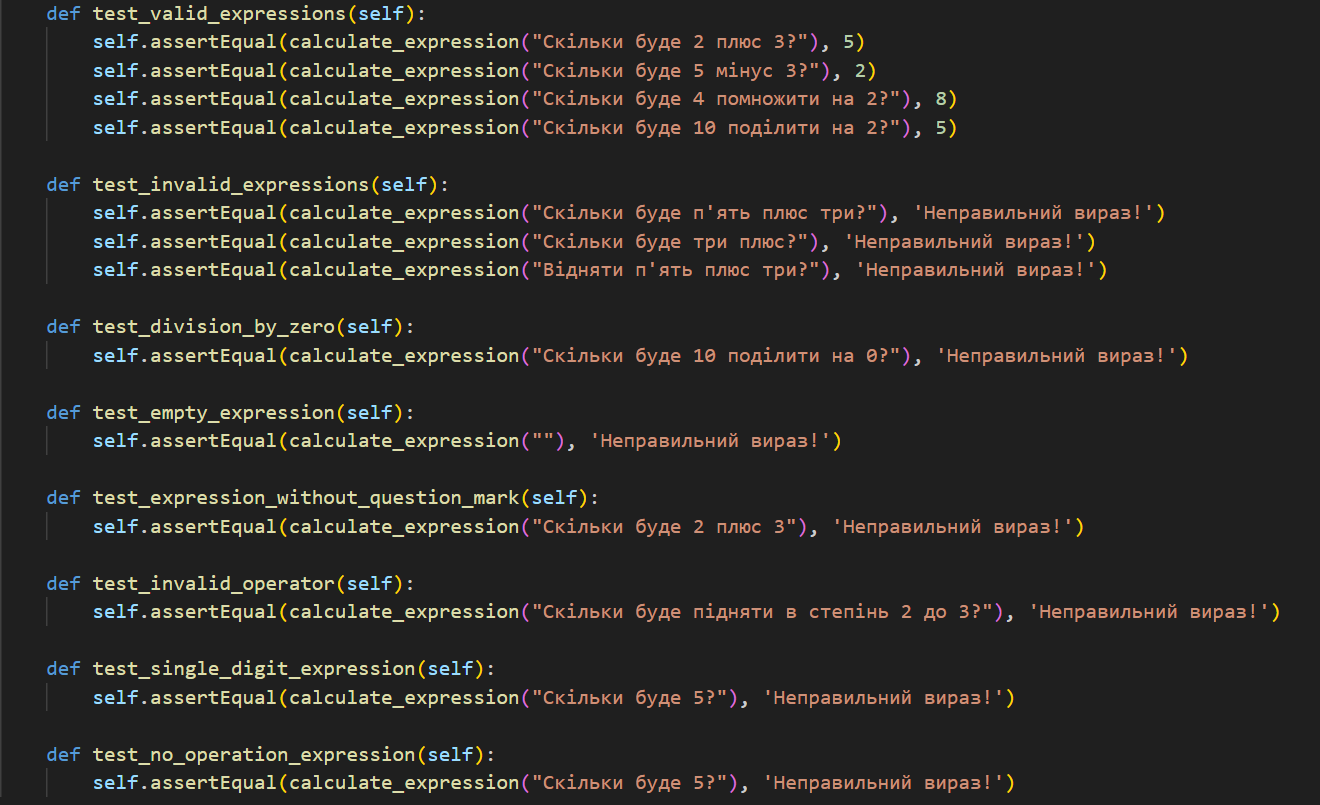


- Коментарі:

Після того, як код пройшов усі тести чату і cms було вирішено потестити здатність чату досягнути певного відсотку покриття. Тести були повністю неробочими, бо використовували синтаксис украхнської мови, а не передбачений вид речень. Зате було введено тести зі словесним поданням цифер.

Промпт 8, 9:

* \*скопійований код\* Here are some ready tests can you correct them?
* No, Скільки буде додати 2 плюс 3? is not a valid expression, since ist`s operator-digit-operator-digit
* Додані тести:

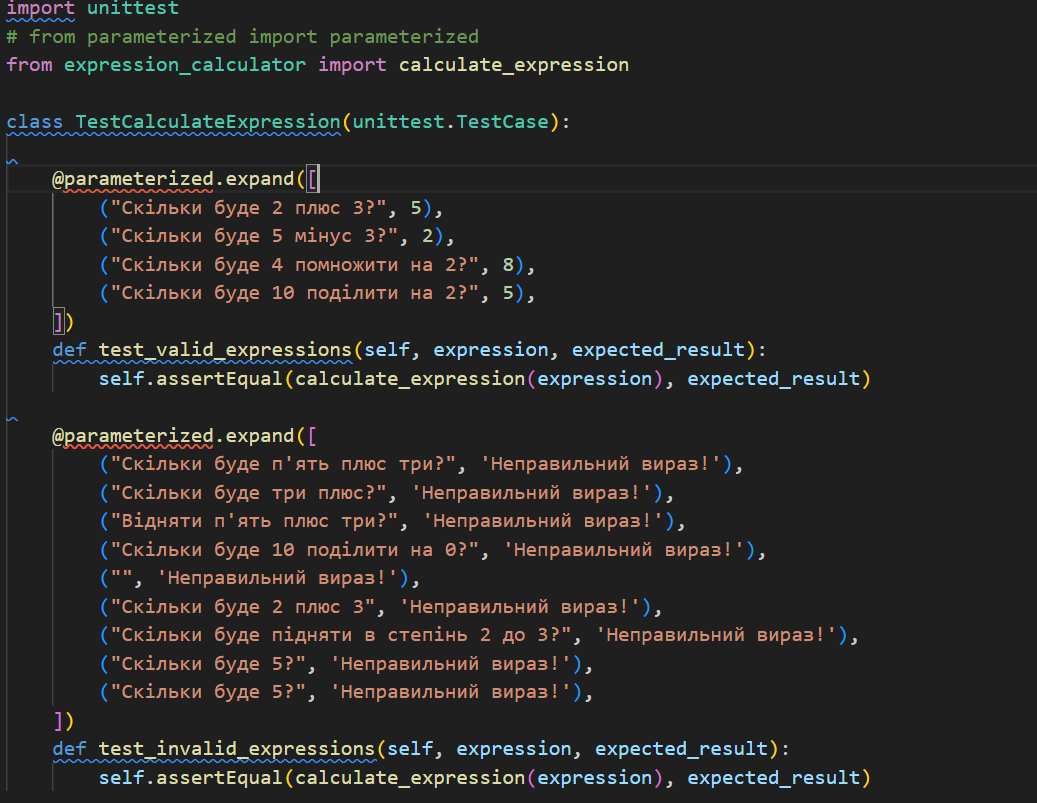


- Коментарі:

Не з першого разу, але ШІ зміг виправити неправильні тести на правильні

Промпт 10-11:

* Now optimize these tests
* Aren`t there any unnecessary tests?
* Додані тести:



- Коментарі:

ШІ переписав тести за допомогою допоміжного модуля, але, навіть з промптами-підказками не забрав тест, що повторювався.

Висновки:

За допомогою умови завдання і додаткових промптів Chat GPT добре справився із завданням створити юніттести для коду. Якщо ж дати йому вже написаний код і попросити його покрити, то без умови ШІ губиться і не створює хороших тестів

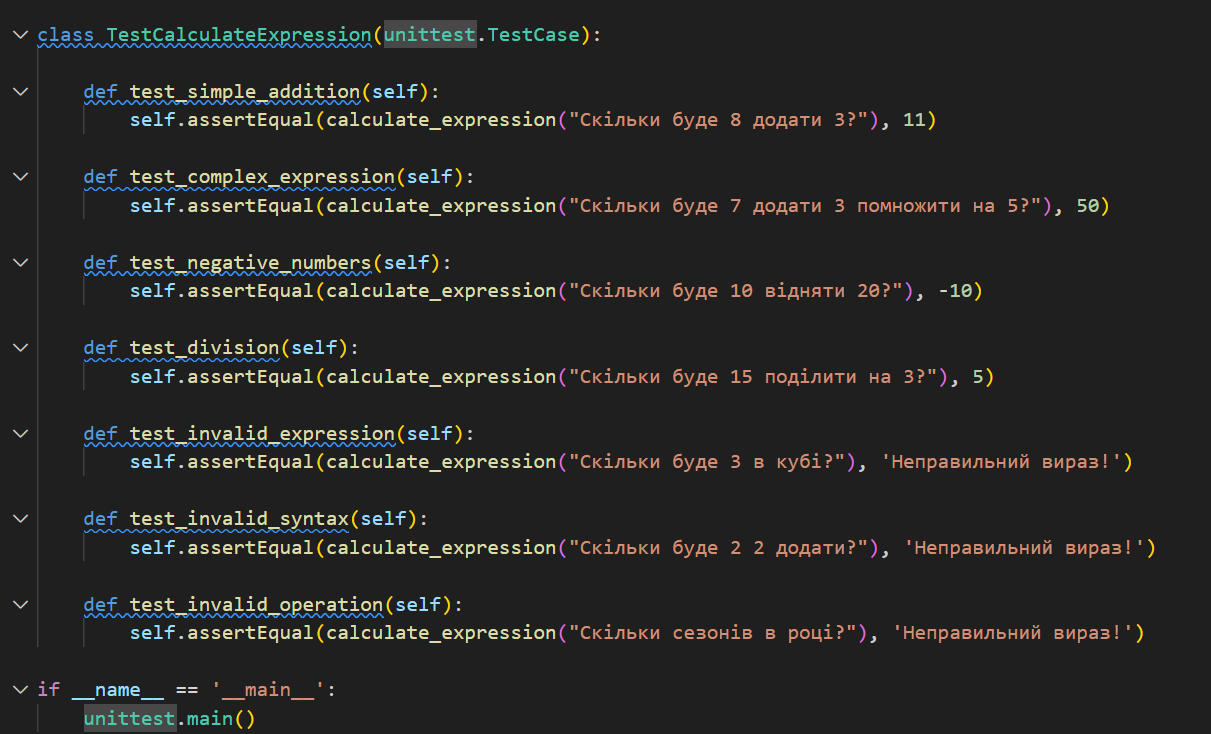
Недостатньо охоплені випадки ШІважко визначити, для цього потрібно багато додатковх підказок. Виправлення помилок пройшло більш успішно, хоч і також не без підказок.

Аналіз покритття тестами у Chat-у досить влучний, адже він покривав код майже повністю, якщо того вимагав промпт.

Зайві тести ШІ не бачить і ігнорує, задля оптимізації пропонує додаткові модулі і бібліотеки.

Gemini:  
 Промпт 1:  
 - \*скопійована задача\* This is a description of a task, I need to code in Python. Your task is to write unittest module to cover all the necessary functions and edgecases

-Запропоновані тести:

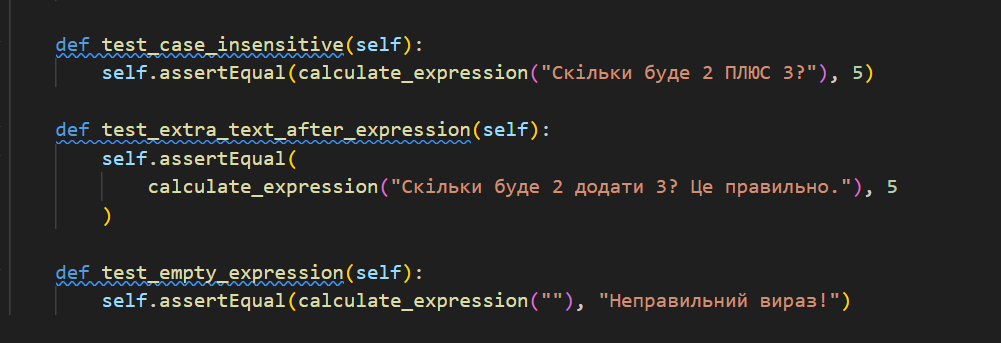


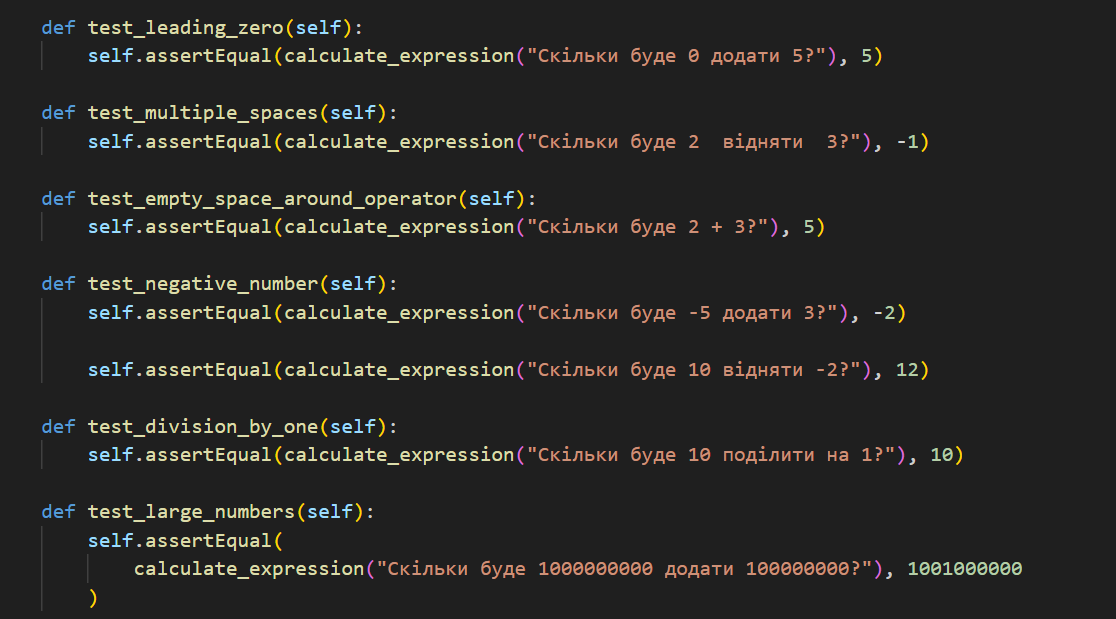
- Коментарі:

Як і Chat GPT, Gemini в першій пропозиції видав лише ті тести, які були вже вказані в умові

Промпт 2, 3:  
 - Now include more possible edge cases

-Додані тести:





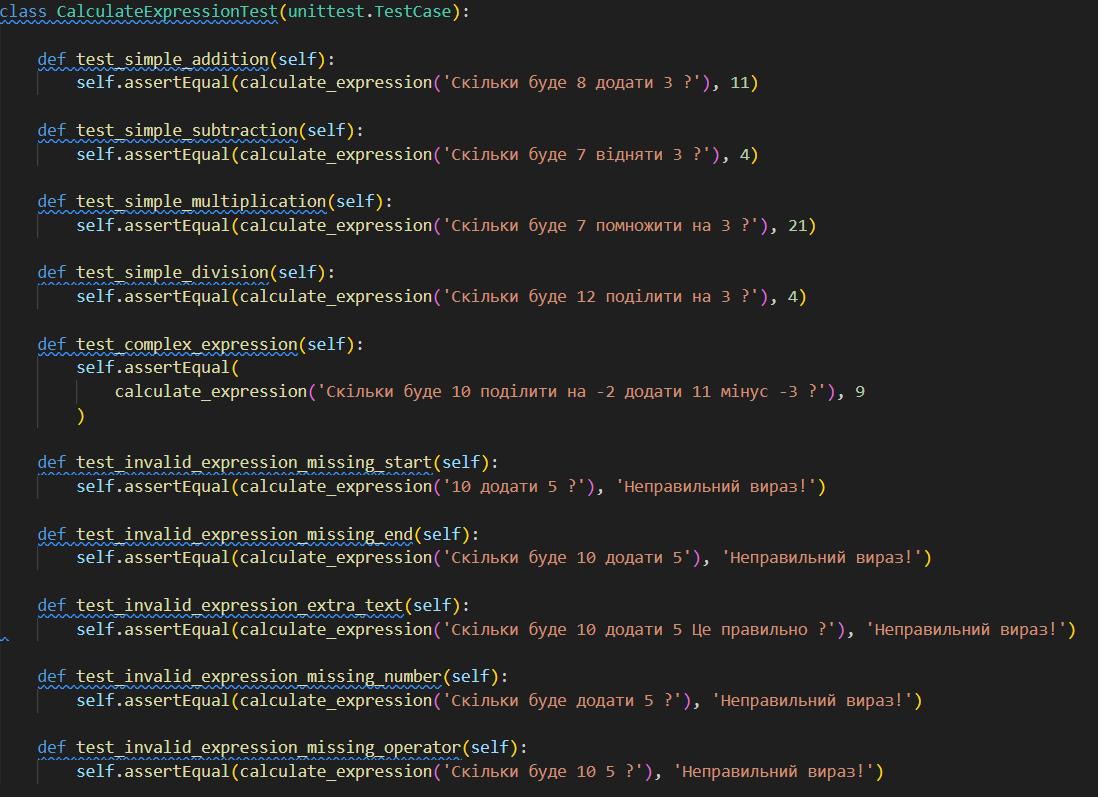
- Коментарі:

Знову ж таки, як і Chat GPT, Gemini додав більше тестів, з тою відмінністю, що він не врахував ділення на 0, але виділив тест на від’ємні числа. На повторному використанні промпту ШІ видав корисні тести, але вони не співпадали з вимогами CMS.

Промпт 3:  
 - You forgot division by 0 and a few more possible edgecases  
 - Коментарі:  
 Тут Gemini спробував додати ще, але переставав генерувати після останнього вже готового тесту. Тому я облишила це заняття.

Промпт 4, 5:

* This is my code, write unittests to cover it. \*скопійований код\*
* Add more tests to make coverage 100%
* Додані тести:



- Коментарі:

ШІ досить добре справився із завданням. Більшість тестів валідні і навіть відповідають вимогам cms. Найкраще покриття коду, якого мені вдалося добитися – 90%.

Промпт 6:

* \*скопійований код\* These are my already written tests for the task. Correct them
* Коментарі:
  + Тут ШІ сказав, що код справді потребує допрацювання і видав такий самий код, як був 10/10

Промпт 7:

* No, Скільки буде додати 2 плюс 3? is not a valid expression, since ist`s operator-digit-operator-digit
* Коментарі:
  + У відповідь Gemini пояснив, що з цим реченням не так (2 з 4 причин від висмоктав з пальця). Нового коду добитись не вдалось.

Промпт 8:

* \*скопійований код\* optimize this code
* Виданий код:



* Коментарі:
  + ШІ запхав деякі з тестів у цикли, але не помітив повторюваного і майже не оптимізував нічого.

Висновки:

ШІ виконав досить добру роботу у створенні автоматичних тестів як за умовою, так і за готовим кодом. З допомогою деяких доробок можна було отримати повністю функціональні тести. На жаль, промпт про збільшення тестів працював лише раз, далі Gemini починав ламатись.

Аналіз покриття тестами у Gemini також на високому рівні. Отриматні в результаті тести покривали 90 і більше відсотків коду.

А от з оптимізацією готових тестів і виправлення помилок ШІ зовсім не справився.