Пара №7

Задание №1 *Спецификация*

1.Предусловия:

1. Входной параметр numbers должен быть списком (или итерируемым объектом), содержащим числовые значения.
2. Если список numbers пуст, функция должна корректно обработать этот случай и вернуть 0.

2.Постусловия:

1. Функция возвращает среднее арифметическое всех положительных элементов в списке numbers.
2. Если в списке numbers нет положительных элементов, функция возвращает 0.

3.Инвариант цикла:

В данном коде есть инвариант цикла. На каждой итерации цикла for num in numbers: выполняются условия - total cумма всех положительных элементов, обработанных на текущий момент. Count -Количество положительных элементов, обработанных на текущий момент.

Таким образом, на каждой итерации цикла если текущий элемент num положительный, то total увеличивается на значение этого элемента, а count увеличивается на 1 или если текущий элемент num не положительный, то total и count остаются неизменными.

Задание №2 *Ручное тестирование*

* Тестовый случай 1: Обычный случай (есть положительные числа)

Входные данные : [1, -2, 3, -4, 5]

Ожидаемый результат: 3.0

Фактический результат: 3.0

* Тестовый случай 2: Граничный случай (нет положительных чисел)

Входные данные : [-1, -2, -3, -4, -5]

Ожидаемый результат: 0

Фактический результат: 0

* Тестовый случай 3: Некорректные входные данные (нарушение предусловия)

Входные данные: [1, 2, 'three', 4, 5]

Ожидаемый результат: Исключение TypeError

Фактический результат: Исключение TypeError

Задание №3 *Анализ корректности и ошибок*

*1)* Деление на ноль гарантированно не произойдет, потому что перед этой строкой стоит условие if count > 0:. Это условие проверяет, что count больше нуля, и только в этом случае выполняется деление. Если count равен нулю, выполнение переходит в блок else, где average присваивается значение 0.

2) Да, функция корректно обрабатывает случай, когда на вход подается пустой список. Да, это соответствует постусловию. Согласно постусловию, если в списке нет положительных элементов, функция должна вернуть 0. В случае пустого списка, положительных элементов нет, поэтому функция возвращает 0, что соответствует спецификации.

3) Если numbers не является списком (например, если numbers является строкой), то при попытке итерации по numbers в цикле for num in numbers: возникнет ошибка TypeError, так как строки не поддерживают итерацию в контексте числовых операций.

* Проверка типа входных данных:

Перед началом обработки можно проверить, является ли numbers списком. Если нет, выбросить исключение с понятным сообщением об ошибке.

for num in numbers: if not isinstance(num, (int, float)):

raise TypeError("Все элементы списка должны быть числовыми.")

* Использование аннотаций типов и документации

Добавление аннотаций типов и более подробной документации может помочь пользователям функции лучше понимать ожидаемые типы входных данных и возможные исключения.

def calculate\_positive\_average(numbers: List[Union[int, float]]) -> float:

Задание №4 *Вывод*

1. Коректность функции

Да, функцию можно считать корректной относительно её спецификации.

*Результаты тестов:*

Обычный случай:

Функция корректно вычисляет среднее арифметическое положительных чисел.

*Граничный случай:*

Функция корректно обрабатывает случай, когда в списке нет положительных чисел, возвращая 0.

*Некорректные входные данные:*

Функция выбрасывает исключение TypeError, если входные данные не соответствуют ожидаемым типам.

Анализ кода:

Функция корректно обрабатывает пустой список, возвращая 0, что соответствует спецификации.

Условие if count > 0: гарантирует, что деление на ноль не произойдет, и функция возвращает 0, если положительных чисел нет.

Функция проверяет типы входных данных и выбрасывает исключения при нарушении предусловий, что делает её более надежной.

2. Метод проверки

Наиболее эффективным методом проверки для обнаружения ошибки деления на ноль в строке A, если бы не было проверки count > 0, было бы **тестирование**.

*Тестирование:*

Тестирование позволяет непосредственно выполнить код с различными входными данными и наблюдать его поведение. В данном случае, тестирование с пустым списком или списком без положительных чисел сразу бы выявило ошибку деления на ноль.

*Статический анализ:*

Статический анализ может выявить потенциальные ошибки без выполнения кода. Однако он может не всегда точно определить все возможные случаи деления на ноль, особенно если они зависят от динамических условий выполнения.

*Формальная верификация:*

Формальная верификация может доказать корректность программы относительно её спецификации. Однако этот метод требует значительных усилий и экспертизы для создания формальных спецификаций и доказательств.

Итог: тестирование является наиболее эффективным методом для обнаружения ошибки деления на ноль в данном случае, так как оно позволяет быстро и непосредственно выявить проблему при выполнении кода с различными входными данными.