МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт Информационных технологий (ИТ)

Кафедра математического обеспечения и стандартизации информационных технологий (МОСИТ)

ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ № 1

по дисциплине

«Тестирование и верификация программного обеспечения»

Тема: «Тестирование программного продукта методом «чёрного ящика»

Выполнила группа студентов ИКБО-11-23

Романов Н.С.
Румянцев А.А.
Прокофьев И.Р.
Чернецов Е.М.
Емельянов А.С.
Кашпирев М.Д.

Принял преподаватель

Петренко А.А.

Практическая работа выполнена «12»09 2025 г. *(подпись студента)*

«Зачтено» «12»09 2025 г. *(подпись руководителя)*

Москва 2025

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Цель работы заключается в знакомстве студентов с процессом тестирования программного обеспечения, включая подготовку технической документации, выявление ошибок и их документирование. Развитие навыков командной работы при тестировании программного продукта.

Для достижения поставленной цели работы студентам необходимо выполнить ряд задач:

1) научиться разрабатывать программный продукт с учетом требований для дальнейшего тестирования;
2) овладеть методом тестирования «черного ящика»;
3) разработать и протестировать техническое задание и документацию;
4) научиться находить ошибки в чужом программном продукте и документировать их;
5) оформить итоговый отчет по проделанной работе в соответствии с установленными стандартами;
6) освоить работу с инструментами для управления тестированием и отслеживания багов;
7) изучить распределение ролей в команде тестировщиков.

ВЫПОЛНЕНИЕ ЗАДАНИЙ

1 Техническое задание на разработку калькулятора

Команда: Сапоги офлайн

Дата создания: 5 сентября 2025 г.

Версия: 1.0

1.1 Введение

1.1.1 Назначение программы

Программный продукт представляет собой **современный научный калькулятор с графическим интерфейсом**, разработанный на языке Python с использованием встроенной библиотеки Tkinter. Программа обеспечивает удобное и интуитивно понятное выполнение арифметических операций, поддерживает ввод десятичных дробей, корректную обработку приоритета операций и отображение результатов в реальном времени.

1.1.2 Область применения

Калькулятор предназначен для широкого круга пользователей:

– Учащиеся школ и студенты для выполнения математических расчётов.

– Инженеры и технические специалисты для быстрых вычислений.

– Пользователи ПК, нуждающиеся в простом и надёжном инструменте.

Программа может использоваться как в автономном режиме, так и в составе образовательных комплексов.

1.2 Основания для разработки

Разработка калькулятора была инициирована в рамках учебного проекта по программированию и пользовательскому интерфейсу. Цель — создание стабильного, функционального и визуально привлекательного приложения с соблюдением лучших практик разработки ПО.

Источники:

– Учебное задание по дисциплине.

– ГОСТ 19.201-78 «Техническое задание. Требования к содержанию и оформлению».

– Рекомендации по UX/UI-дизайну интерфейсов.

1.3 Назначение разработки

Цель разработки — создание **надёжного, удобного и функционального калькулятора**, который:

– Позволяет выполнять базовые арифметические операции с корректным приоритетом.

– Обеспечивает стабильную работу при любых корректных входных данных.

– Имеет современный, отзывчивый интерфейс с эффектами наведения.

– Корректно обрабатывает ошибки (например, деление на ноль).

– Адаптируется к различным разрешениям экрана.

1.4 Требования к программе

1.4.1 Функциональные требования

В таблице 1 представлены функциональные требования.

Таблица 1 — Функциональные требования

№	Функция	Описание
1	Ввод чисел	Поддержка ввода цифр от 0 до 9
2	Ввод операций	Поддержка: +, -, ×, ÷
3	Ввод десятичной точки	Возможность ввода дробных чисел через.
4	Вычисление результата	По нажатию = вычисляется выражение с учётом приоритета операций
5	Очистка экрана	Кнопка C полностью очищает выражение и сбрасывает дисплей
6	Отображение результата	Результат отображается на дисплее и может использоваться в дальнейших вычислениях
7	Поддержка последовательных операций	После результата можно продолжить вычисления
8	Обработка ошибок	При делении на ноль выводится сообщение 'Ошибка'

1.4.2 Требования к надёжности

- Программа не завершается аварийно.
- Все исключения обрабатываются корректно.
- При некорректном вводе (например, .., ÷+) отображается "Error".
- После ошибки можно начать новое вычисление.

1.4.3 Условия эксплуатации

- Операционные системы: Windows, Linux, macOS
- Требуется Python 3.7 или выше
- Минимальные системные требования: 1 ГБ ОЗУ, 10 МБ свободного места

1.4.4 Требования к совместимости

– Работает на всех ОС с установленным Python

– Использует только стандартные библиотеки (tkinter, math)

– Не требует прав администратора

– Поддерживает HiDPI-экраны

1.5 Требования к интерфейсу

1.5.1 Элементы интерфейса

В таблице 2 представлены элементы интерфейса.

Таблица 2 — Элементы интерфейса

Элемент	**Описание**
Дисплей	Поле ввода (только для чтения), шрифт 24 pt, выравнивание по правому краю
Цифровые кнопки (0–9)	Светлый фон, тёмный текст, реакция на наведение
Операционные кнопки (+, -, ×, ÷, =)	Цветовая индикация: синие (операции), зелёная (=), красная (C)
Кнопка .	Ввод десятичной точки
Кнопка C	Очистка выражения

1.5.1 Элементы интерфейса

1.5.2 Стиль интерфейса

– Цветовая схема: тёмная тема (#2c3e50 — фон, #ecf0f1 — дисплей);

– Шрифты: Arial, жирное начертание;

– Эффекты: при наведении кнопки осветляются;

– Размер окна: 350×500 пикселей, центрировано на экране;

– Отзывчивость: интерфейс корректно масштабируется при изменении размеров.

1.6 Критерии приемки

Программа считается принятой, если:

– Все функции работают в соответствии с требованиями.

– Результаты вычислений корректны (включая приоритет операций).

– Интерфейс стабилен и не ломается при использовании.

– Программа не падает при любых допустимых и недопустимых действиях.

– Все ошибки обрабатываются с выводом понятного сообщения.

– 100% тест-кейсов выполняются успешно.

1.7 Требования к документации

1) Техническое задание (настоящий документ)
2) Руководство пользователя
3) Руководство разработчика (по запросу)
4) Описание запуска

1.7.1 Порядок запуска

1) Убедитесь, что установлен Python 3.7 или выше.
2) Скачайте файл calculator.py.
3) Запустите команду: `python calculator.py`
4) Приложение запустится в отдельном окне.

1.8 Порядок контроля и приемки

1.8.1 Методы тестирования

Для проверки корректности работы программного продукта применяются ниже представленные методы тестирования.

Функциональное тестирование: проверка всех основных функций калькулятора: ввод чисел, арифметических операций, вычисление результата, сброс выражения. Примеры тестов:

– 5 + 3 = → ожидаемый результат: 8

– 9 ÷ 3 = → результат: 3

– Нажатие C → дисплей возвращается к значению 0

Граничные значения (Boundary Value Testing): проверка поведения при вводе предельных значений:

– 0, 1, 999999

– Дробные числа: 0.1, 9.999, 0.0001

– Проверка максимальной длины выражения

Негативное тестирование: проверка обработки некорректных или потенциально опасных действий:

– Деление на ноль: 5 ÷ 0 = → должно отображаться "Ошибка"

– Ввод недопустимых последовательностей: .., ++, ×÷, = =

– Попытка ввода нескольких точек в одном числе: 3.14.15

– Пустое выражение перед нажатием =

Интерфейсное тестирование (UI Testing): проверка визуальной составляющей и взаимодействия с пользователем:

– Все кнопки отображаются корректно, без обрезания текста

– Цвета соответствуют дизайну (тёмный фон, светлые цифры, цветные операции)

– Эффекты при наведении курсора работают плавно

– Окно центрируется на экране при запуске

– Шрифты читаемы, размеры элементов соответствуют стандартам UX

Регрессионное тестирование: после внесения изменений или исправления ошибок проводится повторное выполнение ранее пройденных тестов для обеспечения отсутствия новых дефектов.

Тестирование удобства использования (Usability Testing): проверка интуитивности интерфейса:

– Расположение кнопок соответствует привычной логике калькуляторов

– Кнопки имеют достаточный размер для точного нажатия

– Цвета не вызывают напряжения при длительной работе

1.8.2 Приемочные испытания

Приемочные испытания проводятся для подтверждения соответствия программного продукта требованиям, указанным в настоящем техническом задании. Испытания выполняются по утверждённому плану. В таблице 3 продемонстрированы тестовые сценарии.

Таблица 3 — Тестовые сценарии

№	Сценарий	Шаги	Ожидаемый результат
1	Корректное вычисление с приоритетом операций	Ввести 2 + 3 × 4 , нажать =	Результат: 14
2	Деление на ноль	Ввести 10 ÷ 0 , нажать =	Отображается "Ошибка"
3	Работа с десятичными дробями	Ввести 0.5 + 0.3 , нажать =	Результат: 0.8
4	Очистка выражения	Ввести любое выражение, нажать C	Дисплей сбрасывается до 0 , выражение очищается
5	Продолжение вычислений после результата	Ввести 5 + 2 = , затем + 3 =	Результат: 10
6	Ввод дробного числа	Ввести 1.5 , нажать + , затем 2.5 , нажать =	Результат: 4.0
7	Многократное использование =	Ввести 2 + 3 = , затем несколько раз нажать =	Каждое нажатие прибавляет 3: 8 , 11 , 14 и т.д.
8	Ввод только точки	Нажать .	Отображается 0. или ., но не допускается ..
9	Запуск на малом экране	Запустить программу на устройстве с разрешением 1024×768	Окно отображается полностью, не обрезается, центрировано
10	Наведение на кнопки	Навести курсор на любую кнопку	Кнопка плавно освещается (эффект hover)

1.8.3 Критерии успешной приемки

Программный продукт считается **успешно принятым**, если:

– Все тестовые сценарии (№1–№10) выполнены с ожидаемым результатом.

– Программа не завершается аварийно при любых действиях пользователя.

– Все ошибки обрабатываются штатно с выводом понятного сообщения.

– Интерфейс остаётся стабильным и отзывчивым в течение всего сеанса.

– Программа соответствует всем требованиям, указанным в разделах 4–7 настоящего ТЗ.

1.9 Этапы и сроки разработки

Разработка программного продукта выполняется в соответствии с утверждённым графиком. Ниже приведён план-график реализации проекта. В таблице 4 продемонстрированы этапы и сроки разработки.

Таблица 4 — Этапы и сроки разработки

Этап	**Наименование работы**	**Ответственные**	**Срок начала**	**Срок завершения**	**Статус**
1	Анализ требований и проектирование	Все члены команды	05.09.2025	05.09.2025	Выполнено
2	Проектирование интерфейса	Дизайнер, Frontend	06.09.2025	06.09.2025	Выполнено
3	Реализация графического интерфейса (GUI)	Frontend-разработчик	07.09.2025	07.09.2025	Выполнено
4	Реализация логики вычислений	Backend-разработчик	08.09.2025	08.09.2025	Выполнено
5	Валидация ввода и обработка ошибок	QA-инженер, Backend	09.09.2025	09.09.2025	Выполнено
6	Комплексное тестирование	QA-инженер	10.09.2025	10.09.2025	Выполнено
7	Подготовка документации	Технический писатель	11.09.2025	11.09.2025	Выполнено
8	Финальная проверка и релиз	Лидер команды	12.09.2025	12.09.2025	Выполнено

Общий срок разработки: 7 дней

Дата релиза: 12 сентября 2025 г.

Версия ПО: 1.0.0

2 Дополнительная документация к программному продукту

2.1 Руководство пользователя

2.1.1 Назначение

Данный документ предназначен для помощи пользователю в освоении и эффективном использовании калькулятора. Программа подходит для выполнения базовых арифметических операций и работы с десятичными числами.

2.1.2 Запуск программы

1) Убедитесь, что на компьютере установлен Python 3.7 или выше.
2) Скачайте файл calculator.py.
3) Откройте терминал или командную строку в папке с файлом.
4) Выполните команду: `python calculator.py`

2.1.3 Интерфейс и управление

В таблице 5 продемонстрированы основные элементы.

Таблица 5 — Основные элементы

Элемент	**Функция**
Дисплей	Отображает текущее выражение и результат
Кнопки 0–9	Ввод чисел
Кнопка .	Ввод десятичной точки
Кнопка C	Полная очистка дисплея и выражения
Кнопки +, −, ×, ÷	Арифметические операции
Кнопка =	Вычисление результата

Примеры использования:

– Сложение: 5 + 3 = → результат: 8

– Умножение: 4 × 7 = → результат: 28

– Деление: 10 ÷ 2 = → результат: 5

– Работа с дробями: 0.5 + 0.25 = → результат: 0.75

– Продолжение вычислений: 5 + 3 =, затем + 2 = → результат: 10

Обработка ошибок:

– При делении на ноль отображается "Ошибка".

– При некорректном вводе (например, ..) отображается "Error".

– После ошибки нажмите C, чтобы начать заново.

2.1.4 Горячие клавиши (опционально)

Программа поддерживает ввод с клавиатуры:

– Цифры: 0–9

– Операции: +, -, *, /

– Точка: .

– Равно: Enter

– Очистка: Esc или Delete

2.2 Архитектура системы

2.2.1 Общая структура

Калькулятор построен по принципу **MVC-подобной архитектуры** (Model-View-Controller), хотя реализован в одном классе для простоты.

2.2.2 Модули и компоненты

В таблице 6 продемонстрированы основные элементы.

Таблица 6 — Основные элементы

Компонент	**Назначение**
`Calculator.__init__()`	Инициализация окна, переменных, настройка геометрии
`create_widgets()`	Создание дисплея и сетки кнопок
`on_button_click(char)`	Обработка нажатий кнопок
`evaluate_expression()`	Парсинг и вычисление выражения с приоритетом
`on_enter(), on_leave()`	Эффекты при наведении на кнопки
`lighten_color()`	Утилита для визуального эффекта

2.2.3 Логика вычислений

Выражение обрабатывается в три этапа:

1) **Ввод** — символы добавляются в строку expression.
2) **Парсинг** — строка разбивается на токены (числа и операторы).
3) **Вычисление** — применяется приоритет операций (×, ÷ перед +, −).

Пример: Ввод: 2 + 3 × 4 Токены: [2, '+', 3, '×', 4] Вычисление: 3 × 4 = 12 → 2 + 12 = 14 Результат: 14

Ключевые особенности:

- Поддержка десятичных дробей.
- Корректная обработка приоритета операций.
- Продолжение вычислений после результата.
- Защита от деления на ноль.
- Блокировка ввода нескольких точек в одном числе.

2.3 Требования к системе

В таблице 7 продемонстрированы требования к системе.

Таблица 7 — Требования к системе

Параметр	**Требование**
Операционная система	Windows, Linux, macOS
Python	3.7 или выше
Библиотеки	tkinter (входит в стандартную поставку)
Память	100 МБ свободной оперативной памяти
Место на диске	1 МБ
Разрешение экрана	Минимум 1024×768

2.4 Поддержка и обратная связь

По вопросам, предложениям или сообщениям об ошибках обращайтесь: https://github.com/Relax1205/TIVPO.

3 Описание внесённых ошибок в программное обеспечение

3.1 Общее описание

В рамках учебного проекта в программный продукт **"Калькулятор"** было преднамеренно внесено **8 ошибок** различной природы. Ошибки добавлены для последующего тестирования другими командами и отработки навыков выявления дефектов методом "чёрного ящика".

Все ошибки являются **логическими** или **интерфейсными**, синтаксические ошибки отсутствуют, чтобы программа оставалась исполняемой и стабильной.

3.2 Список преднамеренных ошибок

В таблице 8 продемонстрирован список преднамеренных ошибок.

Таблица 8 — Список преднамеренных ошибок

№	Идентификатор	Название ошибки	Тип ошибки	Условия проявления	Способ обнаружения
1	BUG-001	Деление на ноль возвращает inf , а не 'Ошибка'	Логическая	При вводе выражения с делением на ноль, например: 5 ÷ 0	Выполнить 5 ÷ 0 = → ожидается 'Ошибка', фактически: inf
2	BUG-002	Кнопка C не всегда очищает дисплей	Логическая	После нажатия C → через 500 мс значение возвращается	Ввести 5+5 , нажать C , наблюдать за дисплеем — значение может вернуться
3	BUG-003	Операция × выполняет сложение вместо умножения	Логическая	При использовании умножения: 3 × 4	Ввести 3 × 4 = → ожидается 12 , фактически: 7
4	BUG-004	Вычисления выполняются слева направо, без приоритета операций	Логическая	При смешанных выражениях: 2 + 3 × 4	Ввести 2 + 3 × 4 = → ожидается 14 , фактически: 20
5	BUG-005	Первое нажатие кнопки = игнорируется	Логическая	При первом нажатии = после ввода выражения	Ввести 2 + 2 , нажать = , ничего не происходит; второе нажатие работает
6	BUG-006	Интерфейс не адаптируется к малому разрешению экрана	Интерфейсная	На экранах с разрешением < 1280×720	Запустить программу на малом экране — окно выходит за границы
7	BUG-007	Возможность ввода нескольких точек в числе	Логическая	При попытке ввести 1.2.3	Ввести 1.2.3 — калькулятор принимает ввод, хотя число некорректно
8	BUG-008	После результата нельзя продолжить вычисления	Логическая	После = , при попытке продолжить: + 2 =	5 + 3 = 8 , затем + 2 = → результат некорректный или не обновляется

3.3 Подробное описание каждой ошибки

3.3.1 BUG-001: Деление на ноль → inf

– Тип: Логическая

– Место в коде: on_button_click, блок except ZeroDivisionError

– Суть: Вместо вывода "Ошибка" при делении на ноль, программа отображает inf.

– Обнаружение: Выполнить 10 ÷ 0 = → результат inf, что вводит пользователя в заблуждение.

3.3.2 BUG-002: Кнопка C временно очищает дисплей

– Тип: Логическая

– Место в коде: on_button_click('C'), использование self.root.after(500, ...)

– Суть: После очистки через 500 мс дисплей возвращается к старому значению, если выражение не изменилось.

– Обнаружение: Быстро нажать C и не вводить новых символов — значение может восстановиться.

3.3.3 BUG-003: × выполняет сложение

– Тип: Логическая

– Место в коде: on_button_click, строка expr = self.expression.replace('×', '+')

– Суть: Перед вычислением символ × заменяется на +, что приводит к сложению.

– Обнаружение: 3 × 4 = 7 — явный признак ошибки.

3.3.4 BUG-004: Нарушение приоритета операций

– Тип: Логическая

– Место в коде: left_to_right_eval() — вычисления идут строго слева направо

– Суть: Программа не учитывает приоритет умножения и деления.

– Обнаружение: 2 + 3 × 4 = 20 вместо 14.

3.3.5 BUG-005: Игнорирование первого =

– Тип: Логическая

– Место в коде: on_button_click('='), проверка if not hasattr(self, 'equals_pressed')

– Суть: Первое нажатие = игнорируется, что нарушает ожидания пользователя.

– Обнаружение: После ввода выражения первое нажатие = не даёт результата.

3.3.6 BUG-006: Жёстко заданный размер окна

– Тип: Интерфейсная

– Место в коде: main(), строка root.geometry("350x500")

– Суть: Окно не адаптируется под малые экраны, выходит за границы.

– Обнаружение: Запуск на ноутбуке с разрешением 1024×768 — часть интерфейса не видна.

3.3.7 BUG-007: Нет валидации ввода точки

– Тип: Логическая

– Место в коде: Отсутствует проверка количества точек в числе

– Суть: Можно ввести 1.2.3, что не является корректным числом.

– Обнаружение: Ввести 1.2.3 — калькулятор принимает ввод без предупреждения.

3.3.8 BUG-008: Невозможность продолжения вычислений

– Тип: Логическая

– Место в коде: on_button_click — после = выражение не обновляется корректно

– Суть: После результата нельзя продолжить цепочку вычислений.

– Обнаружение: 5 + 3 =, затем + 2 = → результат не 10, а ошибка или старое значение.

3.4 Рекомендации по тестированию

Для выявления ошибок рекомендуется использовать:

– Метод граничных значений: 0, 0., 1.2.3

– Негативное тестирование: деление на ноль, пустые выражения

– Функциональное тестирование: 2 + 3 × 4, 5 × 5

– Интерфейсное тестирование: запуск на малом экране

– Последовательные действия: проверка продолжения вычислений

4 Результаты тестирования программного продукта другой команды

4.1 Ошибка 1: Деление на ноль возвращает inf, а не "Ошибка"

1) Идентификатор: TC-001

2) Название: Проверка деления на ноль

3) Описание: Убедиться, что калькулятор корректно обрабатывает попытку деления на ноль и выводит сообщение об ошибке, а не математическое значение.

4) Предварительные условия:

а) калькулятор запущен;

б) дисплей находится в состоянии ожидания ввода (значение 0 или пустое).

5) Шаги выполнения:

а) ввести число, например, 5;

б) нажать кнопку ÷;

в) ввести 0;

г) нажать кнопку =.

6) Ожидаемый результат: На дисплее отображается сообщение "Ошибка" или "Divide by zero".

7) Фактический результат: На дисплее отображается значение inf.

8) Статус: ❌ Провалено.

4.2 Ошибка 2: Операция умножения выполняет сложение

1) Идентификатор: ТС-003

2) Название: Проверка корректности операции умножения

3) Описание: Убедиться, что операция × выполняет умножение, а не сложение.

4) Предварительные условия

   а) калькулятор запущен;

   б) дисплей сброшен (отображает 0).

5) Шаги выполнения

   а) ввести 3;

   б) нажать кнопку ×;

   в) ввести 4;

   г) нажать кнопку =.

6) Ожидаемый результат: На дисплее отображается 12.

7) Фактический результат: На дисплее отображается 7 (результат сложения: 3 + 4).

8) Статус: ❌ Провалено.

4.3 Ошибка 3: Нарушение приоритета операций (вычисление слева направо)

1) Идентификатор: ТС-004

2) Название: Проверка приоритета операций: умножение перед сложением

3) Описание: Убедиться, что калькулятор учитывает приоритет операций: умножение и деление выполняются перед сложением и вычитанием.

4) Предварительные условия

   а) калькулятор запущен;

   б) дисплей сброшен.

5) Шаги выполнения:

а) ввести 2 + 3 × 4;

б) нажать кнопку =.

6) Ожидаемый результат: Результат: 14 (сначала 3 × 4 = 12, затем 2 + 12 = 14).

7) Фактический результат: Результат: 20 (вычисления выполняются слева направо: 2 + 3 = 5, 5 × 4 = 20).

8) Статус: ❌ Провалено.

4.4 Ошибка 4: Первое нажатие кнопки = игнорируется

1) Идентификатор: ТС-005

2) Название: Проверка реакции на первое нажатие кнопки =

3) Описание: Убедиться, что кнопка = сразу вычисляет результат при первом нажатии.

4) Предварительные условия

а) калькулятор запущен;

б) введено выражение, например, 2 + 2.

5) Шаги выполнения:

а) ввести 2 + 2;

б) нажать кнопку =.

6) Ожидаемый результат: На дисплее отображается 4.

7) Фактический результат: Ничего не происходит. Только при втором и последующих нажатиях = вычисление выполняется.

8) Статус: ❌ Провалено

4.5 Ошибка 5: Интерфейс ломается при малом разрешении экрана

1) Идентификатор: ТС-006

2) Название: Проверка адаптивности интерфейса на малом разрешении

3) Описание: Убедиться, что интерфейс корректно отображается на экранах с низким разрешением.

4) Предварительные условия:

а) экран с разрешением 1024×768 или ниже;

б) калькулятор запущен.

5) Шаги выполнения: запустить программу на устройстве с малым разрешением экрана.

6) Ожидаемый результат: Окно калькулятора отображается полностью, центрировано, все кнопки и дисплей видны.

7) Фактический результат: Окно частично выходит за границы экрана, так как размер жёстко задан как 350x500 без учёта размеров экрана.

8) Статус: ❌ Провалено.

ЗАКЛЮЧЕНИЕ

В ходе выполнения данной практической работы были успешно достигнуты поставленные цели. Мы ознакомились с процессом тестирования программного обеспечения методом «чёрного ящика», что включало в себя разработку технической документации и создание программного продукта — калькулятора — с преднамеренно внесёнными ошибками.

Были развиты навыки командной работы, а также освоены ключевые задачи тестировщика: разработка программного продукта с учётом требований для последующего тестирования, применение метода «черного ящика» для поиска дефектов в программном обеспечении другой команды, и документирование найденных ошибок в соответствии с установленными стандартами. Все восемь преднамеренно внесённых логических и интерфейсных ошибок были успешно обнаружены и задокументированы, что подтверждает эффективность проделанной работы.

ПРИЛОЖЕНИЯ

```
collected 6 items

test_calculator.py::test_division_by_zero FAILED
test_calculator.py::test_clear_button_bug PASSED
test_calculator.py::test_multiply_is_addition PASSED
test_calculator.py::test_wrong_operation_order FAILED
test_calculator.py::test_equals_first_press_ignored PASSED
test_calculator.py::test_fixed_window_size PASSED
```

Приложение 1 — Результат прогона тестов Pytest