



МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

**Институт информационных технологий (ИИТ)
Кафедра математического обеспечения и стандартизации
информационных технологий (МОСИТ)**

ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ
по дисциплине «Тестирование и верификация программного обеспечения»

Практическое занятие № 1

Студентки группы *ИКБО-50-23, Зернова К.А. Мудрак И. А.*

(подпись)

Преподаватель *Ильичев Г.П*

(подпись)

Отчет представлен « » сентября 2025 г.

Москва 2025 г.

Оглавление

Техническое задание	3
Описание внесённых ошибок в ПО "MatrixCalculator"	12
Техническое задание, проверяемого программного продукта	14
Результаты тестирования программного продукта	19
Анализ документации	29
Заключение	30

Техническое задание

Введение

Данный документ представляет собой техническое задание (ТЗ) на разработку консольного приложения «MatrixCalculator». Приложение предназначено для выполнения базовых и продвинутых математических операций с матрицами. Область применения включает учебный процесс (для студентов и преподавателей математических специальностей), инженерные расчеты и прототипирование алгоритмов линейной алгебры.

Основания для разработки

Разработка инициирована в связи с отсутствием простых, кроссплатформенных и не требующих графического интерфейса инструментов для работы с матрицами.

Назначение разработки

Приложение предназначено для выполнения базовых и продвинутых математических операций с матрицами. Область применения включает учебный процесс (для студентов и преподавателей математических специальностей), инженерные расчеты и прототипирование алгоритмов линейной алгебры.

Требования к программе

Функциональные требования

Приложение должно предоставлять пользовательское меню для выбора операций. Программа должна последовательно запрашивать у пользователя все необходимые данные.

Перечень функций:

1. Ввод матриц: Возможность ввода матриц с клавиатуры с указанием размерности.
2. Сложение и вычитание матриц: Операции выполняются над матрицами одинаковой размерности.
3. Умножение матрицы на число.
4. Перемножение матриц: С контролем согласованности размерностей (число столбцов первой матрицы должно равняться числу строк второй).
5. Транспонирование матрицы.
6. Вычисление определителя (детерминанта) матрицы: Для квадратных матриц.

7. Вычисление обратной матрицы: Для квадратных невырожденных матриц (с определителем $\neq 0$).

8. Возведение матрицы в степень: Для натуральных степеней и квадратных матриц.

9. Выход из программы.

Требования к надежности

Отказоустойчивость: Приложение не должно завершаться с ошибкой при вводе пользователем нечисловых значений, неверных размерностей матриц или при попытке выполнения недопустимой операции (например, деление на ноль при вычислении обратной матрицы). В таких случаях должно выводиться понятное сообщение об ошибке с предложением повторить ввод или выбрать другую операцию.

Восстановление после сбоев: После возникновения ошибки приложение должно возвращать пользователя в главное меню или на предыдущий шаг ввода.

Условия эксплуатации

Среда выполнения: Интерпретатор Python версии 3.8 или выше.

Аппаратные требования: Любой компьютер, способный запускать интерпретатор Python. Особых требований к объему оперативной памяти или процессору не предъявляется.

Требования к совместимости

Приложение должно быть кроссплатформенным и работать в любой операционной системе, где установлен корректный интерпретатор Python (Windows, Linux, macOS).

Взаимодействие с другими системами не предусмотрено.

Требования к интерфейсу

Интерфейс — консольный (командная строка). Взаимодействие осуществляется через последовательный вывод меню и запросов на ввод

данных.

Текстовое описание интерфейса:

1. При запуске отображается главное меню с нумерованным списком операций.
2. Пользователь вводит цифру, соответствующую операции.
3. Для операций, требующих ввода матриц, программа запрашивает:
Количество строк и столбцов.
Поэлементный ввод значений матрицы (по строкам).
4. После ввода всех необходимых данных программа выводит на экран исходные матрицы (если применимо) и результат операции в читаемом, форматированном виде (например, каждую строку матрицы с отступами).
5. После вывода результата программа возвращает пользователя в главное меню.

Критерии приемки

- Успешно выполняются 100% позитивных тест-кейсов (корректные данные) для всех заявленных операций. Результаты должны совпадать с расчетами, выполненными вручную.
- Успешно обрабатываются 100% негативных тест-кейсов (некорректные данные: буквы вместо цифр, неверные размерности и т.д.) с выводом адекватных сообщений об ошибках без завершения работы программы.
- Программа запускается и выполняет все заявленные функции на трех указанных ОС: Windows 10, Ubuntu 22.04, macOS Ventura.

Требования к документации

Обязательная документация

- Запрос пользователя с описанием требуемой системы
- Техническое описание программы
- Техническое задание
- Исходный код с соответствующими комментариями

Порядок контроля и приемки

Приемка будет проводиться путем тестирования методом «черного ящика» на соответствие критериям приемки (п.6). Разработчик предоставляет исполняемый `.ру` файл и документацию. Заказчик проводит тестирование по заранее подготовленным тестовым сценариям:

Тестирование включает в себя следующие группы тест-кейсов:

1. Позитивное тестирование (корректные данные и операции):

Сложение матриц.

Действия: Выбрать операцию сложения. Ввести две матрицы размерности 2x2 с целыми числами.

Ожидаемый результат: Программа выводит корректную сумму матриц.

Вычитание матриц.

Действия: Выбрать операцию вычитания. Ввести две матрицы размерности 2x2.

Ожидаемый результат: Программа выводит корректную разность матриц.

Умножение матрицы на число.

Действия: Выбрать операцию умножения на число. Ввести матрицу и числовой скаляр.

Ожидаемый результат: Программа выводит матрицу, каждый элемент которой умножен на скаляр.

Умножение матриц (согласованные размерности).

Действия: Выбрать операцию умножения матриц. Ввести матрицу А размерности 2×3 и матрицу В размерности 3×2 .

Ожидаемый результат: Программа выводит корректную матрицу-произведение размерности 2×2 .

Транспонирование матрицы.

Действия: Выбрать операцию транспонирования. Ввести прямоугольную матрицу (например, 3×2).

Ожидаемый результат: Программа выводит корректно транспонированную матрицу (2×3).

Вычисление определителя.

Действия: Выбрать операцию вычисления определителя. Ввести квадратную матрицу (например, 3×3) с известным определителем, не равным нулю.

Ожидаемый результат: Программа выводит корректное числовое значение определителя.

Вычисление обратной матрицы.

Действия: Выбрать операцию нахождения обратной матрицы. Ввести квадратную матрицу (2×2 или 3×3) с известным определителем, не равным нулю.

Ожидаемый результат: Программа выводит корректную обратную матрицу. Проверка: умножение исходной матрицы на полученную обратную дает единичную матрицу (проверка вручную или выбором операции умножения в программе).

Возведение матрицы в степень.

Действия: Выбрать операцию возведения в степень. Ввести квадратную матрицу и натуральную степень (например, 2 или 3).

Ожидаемый результат: Программа выводит корректно возведенную в степень матрицу.

2. Негативное тестирование (обработка ошибок):

Ввод нечисловых значений.

Действия: На любом шаге ввода (размерность, элемент матрицы) ввести символы, отличные от цифр (например, abc, 1.2.3).

Ожидаемый результат: Программа выводит понятное сообщение об ошибке и повторяет запрос, не завершаясь аварийно.

Операции с матрицами несовместимой размерности.

Действия:

а) Выбрать сложение/вычитание. Ввести матрицы разной размерности (например, 2×2 и 3×3).

б) Выбрать умножение матриц. Ввести матрицы, где число столбцов первой не равно числу строк второй (например, 2×2 и 3×3).

Ожидаемый результат: Программа выводит понятное сообщение об ошибке (например, "Ошибка: для сложения матрицы должны быть одного размера") и возвращает в меню или шаг ввода.

Операции, требующие квадратную матрицу, с неквадратной.

Действия: Выбрать операцию нахождения определителя/обратной матрицы/возведения в степень. Ввести прямоугольную матрицу (например, 2×3).

Ожидаемый результат: Программа выводит понятное сообщение об ошибке (например, "Ошибка: матрица должна быть квадратной").

Вычисление обратной матрицы для вырожденной матрицы (определитель = 0).

Действия: Выбрать операцию нахождения обратной матрицы. Ввести квадратную матрицу с нулевым определителем.

Ожидаемый результат: Программа выводит понятное сообщение об ошибке (например, "Ошибка: определитель матрицы равен нулю, обратной матрицы не существует").

ТК-Н-05: Ввод недопустимой размерности.

Действия: При запросе размерности ввести число меньше или равное нулю (например, 0, -5).

Ожидаемый результат: Программа выводит сообщение об ошибке

(например, "Размерность матрицы должна быть положительным числом") и повторяет запрос.

Этапы и сроки разработки

1. Проектирование архитектуры и алгоритмов: 1 день.
2. Реализация функций ввода-вывода и валидации данных: 2 дня.
3. Реализация базовых арифметических операций (+, -, на число): 1 день.
4. Реализация алгоритмов перемножения, транспонирования, нахождения определителя и обратной матрицы: 3 дня.
5. Тестирование и отладка: 2 дня.
6. Написание документации: 1 день.

Общий срок выполнения проекта: 10 рабочих дней.

Описание внесённых ошибок в ПО "MatrixCalculator"

1. Ошибка валидации размера матрицы при вводе

- Тип ошибки: Логическая/Интерфейсная
- Местоположение: Функция `input_matrix()` в основном файле
- Описание: Закомментирована проверка соответствия количества введенных элементов заданной размерности матрицы
- Способ обнаружения: При вводе строки с количеством элементов, не соответствующим заявленной размерности (например, для матрицы 2x2 ввести 3 элемента в строке)

2. Ошибка валидации операций с матрицами

- Тип ошибки: Логическая
- Местоположение: Функции `matrix_addition()`, `matrix_multiplication()`
- Описание: Закомментированы проверки совместимости размерностей матриц для операций сложения и умножения
- Способ обнаружения: Попытка сложить матрицы разной размерности или умножить матрицы с несовместимыми размерами

3. Ошибка обработки вырожденных матриц

- Тип ошибки: Логическая
- Местоположение: Функция `matrix_inverse()`
- Описание: Закомментирована проверка определителя на равенство нулю перед вычислением обратной матрицы
- Способ обнаружения: Попытка вычислить обратную матрицу для вырожденной матрицы (с определителем = 0)

4. Ошибка валидации степени матрицы

- Тип ошибки: Логическая
- Местоположение: Функция `matrix_power()`
- Описание: Закомментирована проверка, что степень является натуральным числом больше 1
- Способ обнаружения: Попытка возвести матрицу в степень 0 или отрицательную степень

5. Неполная обработка ошибок

- Тип ошибки: Логическая

- Местоположение: Функция ``matrix_inverse()`` в модуле `matrix.py`
- Описание: Закомментирована проверка на вырожденность матрицы в алгоритме обращения
- Способ обнаружения: Попытка обратить вырожденную матрицу

Техническое задание, проверяемого программного продукта

Введение

Поступил заказ для создания эмулятора Shell для языка оболочки UNIX-подобной операционной системы. Продукт должен реализовывать команды, содержать уже готовую конфигурацию и работать в режиме графического редактора.

Основания для разработки

Данный заказ поступил для проверки возможностей программистов в сфере разработки программ, на примере Эмулятора Shell. Проект должен продемонстрировать способности программистов и их компетентность в реализации сложных программ с использованием языка программирования Python. Для разработки предоставлены:

- Документация для работы в Python;
- Методические материалы для реализации заказа;
- ГОСТ 34.602-2020 для составления технического задания.

Назначение разработки

Целью разработки является создание простого Эмулятора Shell с конфигурацией в соответствии с заказом, а также определенными стандартными функциями для оболочки ОС. В итоге ожидается готовый безотказный продукт, способный продемонстрировать ожидаемый функционал.

Требования к программе

4.1 Функциональные требования

Эмулятор должен работать в режиме GUI и принимать образ виртуальной файловой системы в виде файла формата zip.

Должен быть создан конфигурационный файл в формате csv который будет содержать:

- Имя пользователя для показа в приглашении к вводу;
- Путь к архиву виртуальной файловой системы;
- Путь к старому архиву.

Эмулятор должен поддерживать команды:

- ls – отображение файлов и директорий внутри заданной;
- cd – перемещение по директориям;
- exit – выход из эмулятора;
- find – нахождение по шаблону файла/директории;
- chown – изменение владельца файла;
- tail – вывод последних строк текстового файла.

Запуск должен осуществляться через консоль (python main.py config.csv).

4.2 Требования к надежности

Эмулятор должен обрабатывать только команды, описанные в функциональных требованиях. Попытки ввести иные команды или случайный набор символов вне и внутри готовых должны сопровождаться сообщением об ошибке цвета, отличного от стандартного темного.

4.3 Условия эксплуатации

Операционная система Windows 10 или выше;

Минимальные требования персонального компьютера: современный процессор (любой), 256 Мб оперативной памяти, 1 Мб свободного места на диске.

4.4 Требования к совместимости

Программа требует предустановки языка программирования Python, так как реализована на данном языке (.py).

Требования к интерфейсу

Эмулятор должен имитировать оболочку UNIX-подобной ОС, где основными цветами являются: розовый (фон терминала), темно-синий/близкий к черному (текст), красный (сообщения об ошибках) и белый/custom (внешняя составляющая терминала).

Текст должен быть разборчивым, достаточно большим. Формат вывода консоли: <пользователь>:~<текущая директория> “команда”. (С новой строки – результат выполнения команды, если предусмотрен).

Критерии приемки

Продукт считается соответствующим настоящему ТЗ и готовым к приемке, если:

- Успешно пройдены все тест-кейсы, составленные на основе функциональных требований, указанных в разделе 4.1;
- Интерфейс программы соответствует требованиям раздела 5;
- Программа запускается и функционирует на целевой операционной системе, указанной в п. 4.3.

Требования к документации

В состав поставки программного продукта должна входить следующая документация:

- Краткое руководство пользователя (в формате README.md).

Порядок контроля и приемки

Тестирование программы будет проводиться методом "черного ящика" на основе требований, изложенных в настоящем техническом задании.

Приемочные испытания включают в себя:

- Функциональное тестирование всех элементов интерфейса;
- Тестирование корректности вывода при выполнении команд;
- Тестирование удобства использования.

Недостаточно подробно описан порядок приемки

Этапы и сроки разработки

1. Проектирование архитектуры эмулятора – 1 дня;
2. Разработка кода программы и конфигурации – 4 дня;
3. Написание сопроводительной документации – 1 день;
4. Внутреннее тестирование – 1 день.

Общий срок разработки – 7 дней.

Результаты тестирования программного продукта

В ходе тестирования терминального эмулятора было выявлено 9 дефектов.

Критические ошибки

1. Идентификатор.

ТС-001

2. Название.

Невозможность навигации по файловой системе командой `cd`.

3. Описание.

Убедиться, что команда `cd` корректно меняет текущую рабочую директорию.

4. Предварительные условия:

а) Открыта командная строка (терминал).

5. Шаги выполнения:

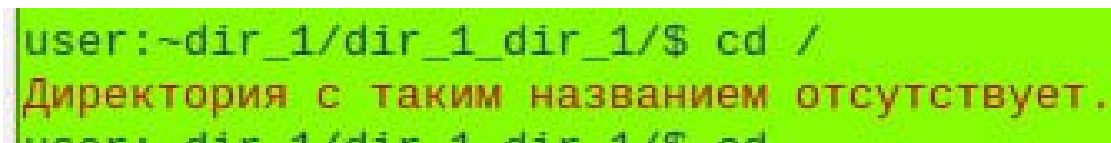
1. В командной строке ввести `cd /` для перехода в корневой каталог.

2. Нажать Enter.

6. Ожидаемый результат.

Текущая рабочая директория успешно меняется на корневую (`/`).

7. Фактический результат.



```
user:~dir_1/dir_1_dir_1/$ cd /  
Директория с таким названием отсутствует.
```

8. Статус.

Failed.

Средние ошибки

1. Идентификатор.

ТС-002

2. Название.

Команда `chown` некорректно обрабатывает несуществующего пользователя.

3. Описание.

Убедиться, что команда `chown` завершается с ошибкой при попытке сменить владельца на несуществующего пользователя.

4. Предварительные условия:

- а) Открыта командная строка.
- б) У пользователя есть права на выполнение команды `chown`.

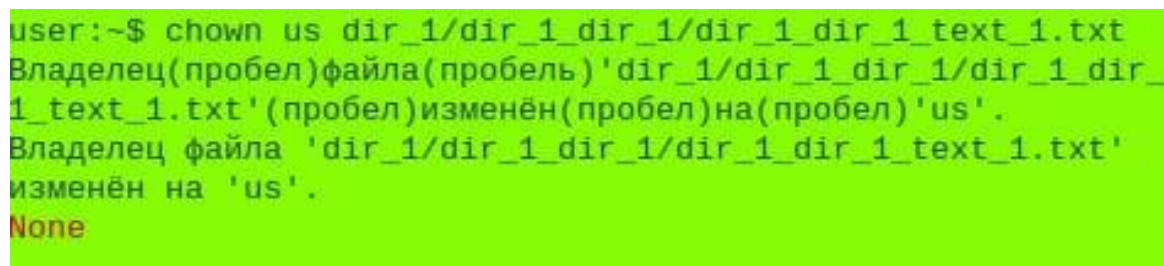
5. Шаги выполнения:

1. Попытаться сменить владельца на несуществующего пользователя: `chown us test.txt`.

6. Ожидаемый результат.

Команда завершается с ошибкой, выводя сообщение типа `chown: invalid user: 'nonexistentuser'`. Владелец файла не изменен.

7. Фактический результат.



```
user:~$ chown us dir_1/dir_1_dir_1/dir_1_dir_1_text_1.txt
Владелец(пробел)файла(пробель) 'dir_1/dir_1_dir_1/dir_1_dir_1_text_1.txt' (пробел)изменён(пробел)на(пробел) 'us'.
Владелец файла 'dir_1/dir_1_dir_1/dir_1_dir_1_text_1.txt'
изменён на 'us'.
None
```

8. Статус.

Failed

1. Идентификатор.

ТС-003

2. Название.

Команда `find` выводит лишние символы.

3. Описание.

Убедиться, что вывод команды `find` корректный.

4. Предварительные условия:

- а) Открыта командная строка.
- б) В текущей директории есть файлы и/или поддиректории.

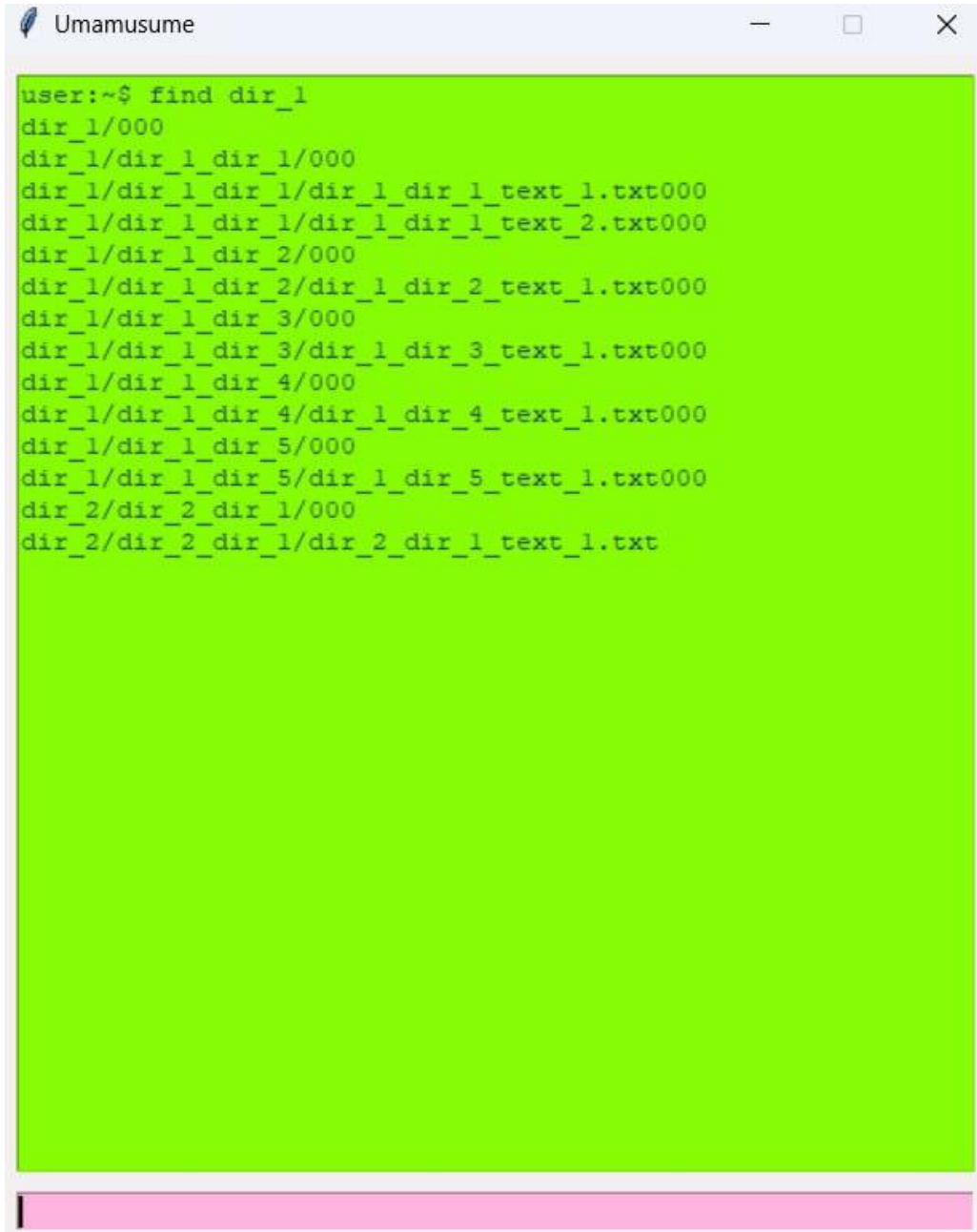
5. Шаги выполнения:

1. Ввести команду для поиска всех файлов и директорий: `find . dir1 ``.

6. Ожидаемый результат.

Выводятся абсолютные пути к найденным файлам и директориям в стандартном формате (например, `./dir/file.txt``).

7. Фактический результат.



```
user:~$ find dir_1
dir_1/000
dir_1/dir_1_dir_1/000
dir_1/dir_1_dir_1/dir_1_dir_1_text_1.txt000
dir_1/dir_1_dir_1/dir_1_dir_1_text_2.txt000
dir_1/dir_1_dir_2/000
dir_1/dir_1_dir_2/dir_1_dir_2_text_1.txt000
dir_1/dir_1_dir_3/000
dir_1/dir_1_dir_3/dir_1_dir_3_text_1.txt000
dir_1/dir_1_dir_4/000
dir_1/dir_1_dir_4/dir_1_dir_4_text_1.txt000
dir_1/dir_1_dir_5/000
dir_1/dir_1_dir_5/dir_1_dir_5_text_1.txt000
dir_2/dir_2_dir_1/000
dir_2/dir_2_dir_1/dir_2_dir_1_text_1.txt
```

8. Статус.

Failed.

1. Идентификатор.

ТС-004

2. Название.

Команда `'tail'` выводит неверное количество строк.

3. Описание.

Убедиться, что команда `'tail -n 1'` выводит ровно одну последнюю строку файла.

4. Предварительные условия:

а) Открыта командная строка.

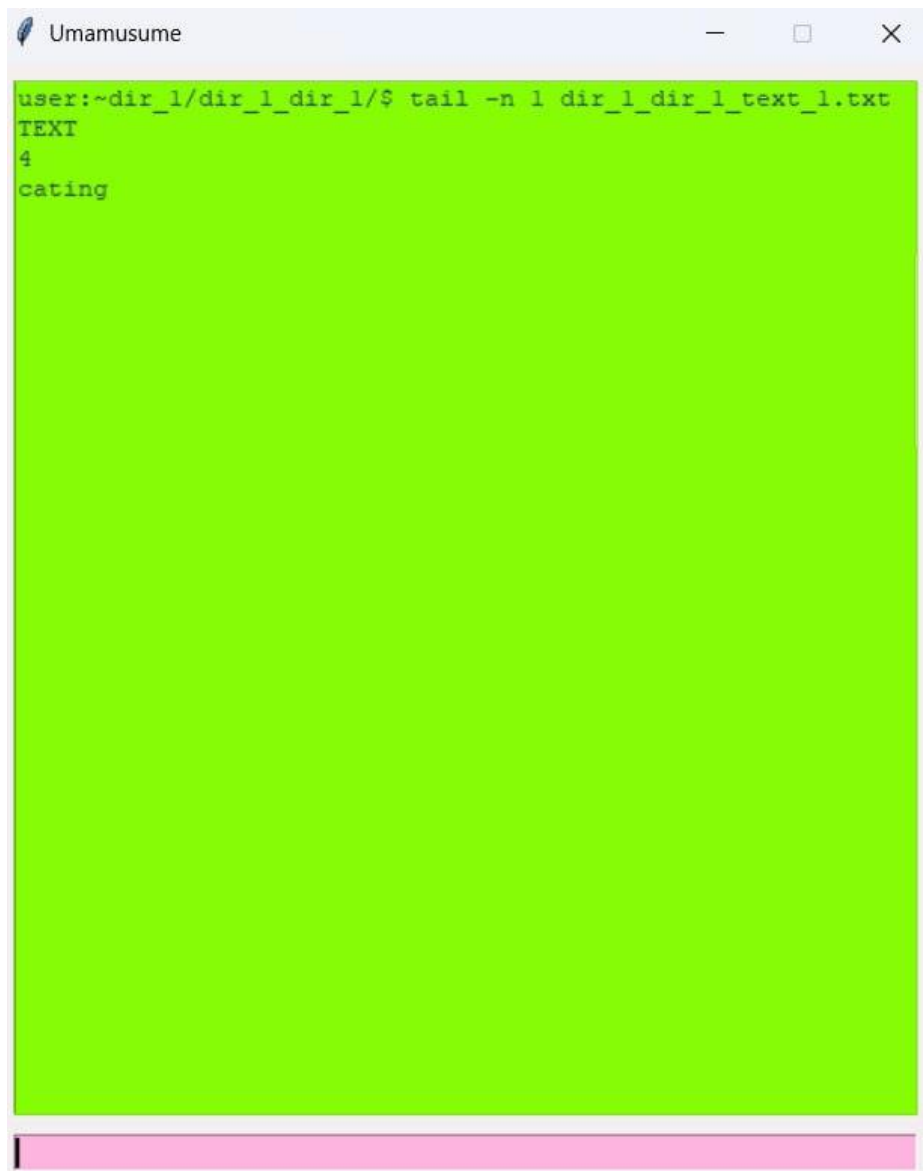
5. Шаги выполнения:

2. Выполнить команду для вывода одной строки: `'tail -n 1 text.txt'`.

6. Ожидаемый результат.

На стандартный вывод поступает ровно одна (последняя) строка из файла.

7. Фактический результат.

A terminal window titled 'Umamusume' with standard window controls. The prompt is 'user:~dir_1/dir_1_dir_1/\$'. The command 'tail -n 1 dir_1_dir_1_text_1.txt' has been executed. The output is displayed on three lines: 'TEXT', '4', and 'cating'.

```
user:~dir_1/dir_1_dir_1/$ tail -n 1 dir_1_dir_1_text_1.txt
TEXT
4
cating
```

8. Статус.

Failed.

Незначительные ошибки

1. Идентификатор.

ТС-005

2. Название.

Отсутствие информации о текущей директории в приглашении командной строки.

3. Описание

Убедиться, что в командной строке отображается текущая рабочая директория.

4. Предварительные условия:

а) Запущен терминал.

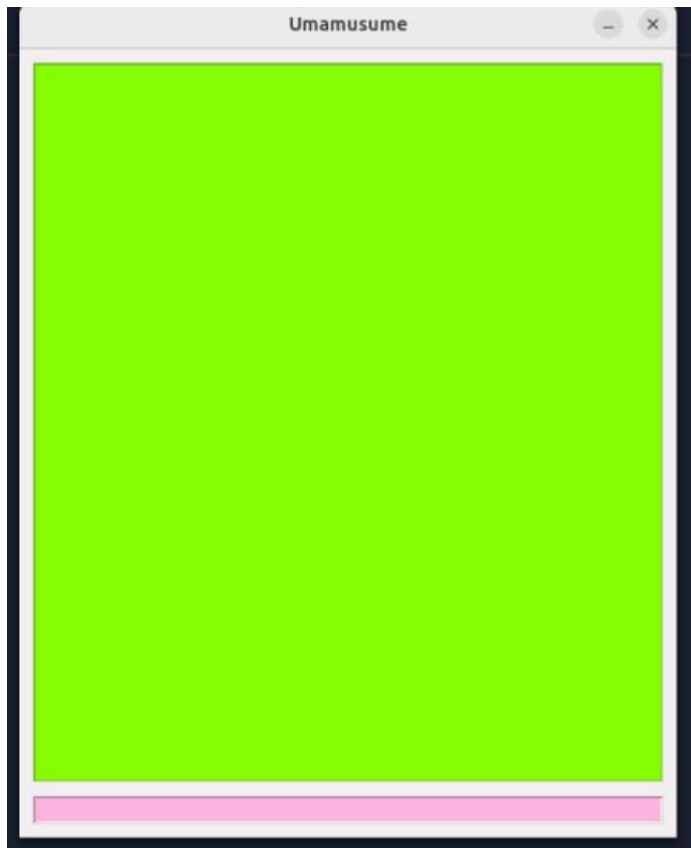
5. Шаги выполнения:

1. Посмотреть на командную строку.

6. Ожидаемый результат.

В приглашении командной строки отображается текущая рабочая директория (например, ``user@host:~$``).

7. Фактический результат.



8. Статус.

Failed.

1. Идентификатор.

ТС-006

2. Название.

Команда ``ls`` не выводит ошибку для несуществующей директории.

3. Описание.

Убедиться, что команда `ls` корректно сообщает об ошибке при запросе к несуществующей директории.

4. Предварительные условия:

а) Открыта командная строка.

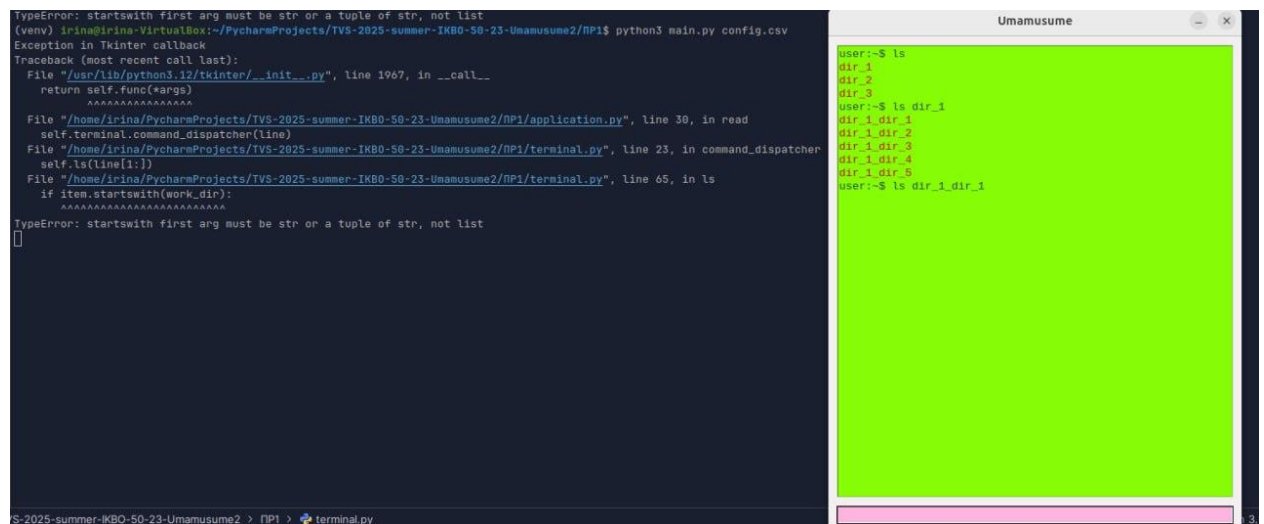
5. Шаги выполнения:

1. Ввести команду: `ls /несуществующая_директория`.

6. Ожидаемый результат.

Команда завершается с ошибкой, выводя стандартное сообщение, например: `ls: cannot access '/несуществующая_директория': No such file or directory`.

7. Фактический результат.



The image shows two side-by-side terminal windows. The left window displays a Python traceback for a `TypeError: startswith first arg must be str or a tuple of str, not list`. The right window, titled 'Umamusume', shows the output of the command `ls` in a green terminal environment, listing several directories: `dir_1`, `dir_2`, `dir_3`, `dir_1_dir_1`, `dir_1_dir_2`, `dir_1_dir_3`, `dir_1_dir_4`, `dir_1_dir_5`, and `dir_1_dir_1_dir_1`.

8. Статус.

Failed.

1. Идентификатор.

ТС-007

2. Название.

Команда `chown` выводит некорректное сообщение об ошибке для несуществующего файла.

3. Описание.

Убедиться, что команда `chown` выводит четкое сообщение об ошибке при попытке изменить владельца несуществующего файла.

4. Предварительные условия:

а) Открыта командная строка.

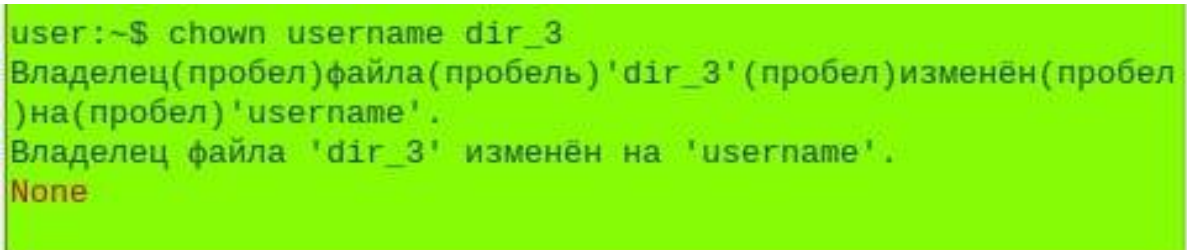
5. Шаги выполнения:

1. Попытаться сменить владельца для несуществующего файла: `chown user nonexistentfile.txt`.

6. Ожидаемый результат.

Четкое сообщение об ошибке, например: `chown: cannot access 'nonexistentfile.txt': No such file or directory`.

7. Фактический результат.



```
user:~$ chown username dir_3
Владелец(пробел)файла(пробель)'dir_3'(пробел)изменён(пробел)
)на(пробел)'username'.
Владелец файла 'dir_3' изменён на 'username'.
None
```

8. Статус.

Failed.

Ошибки оформления

1. Идентификатор.

ТС-008

2. Название.

Проверка корректности цвета фона терминала.

3. Описание.

Убедиться, что фон терминала соответствует цвету, заданному в техническом задании.

4. Предварительные условия:

а) Терминал запущен.

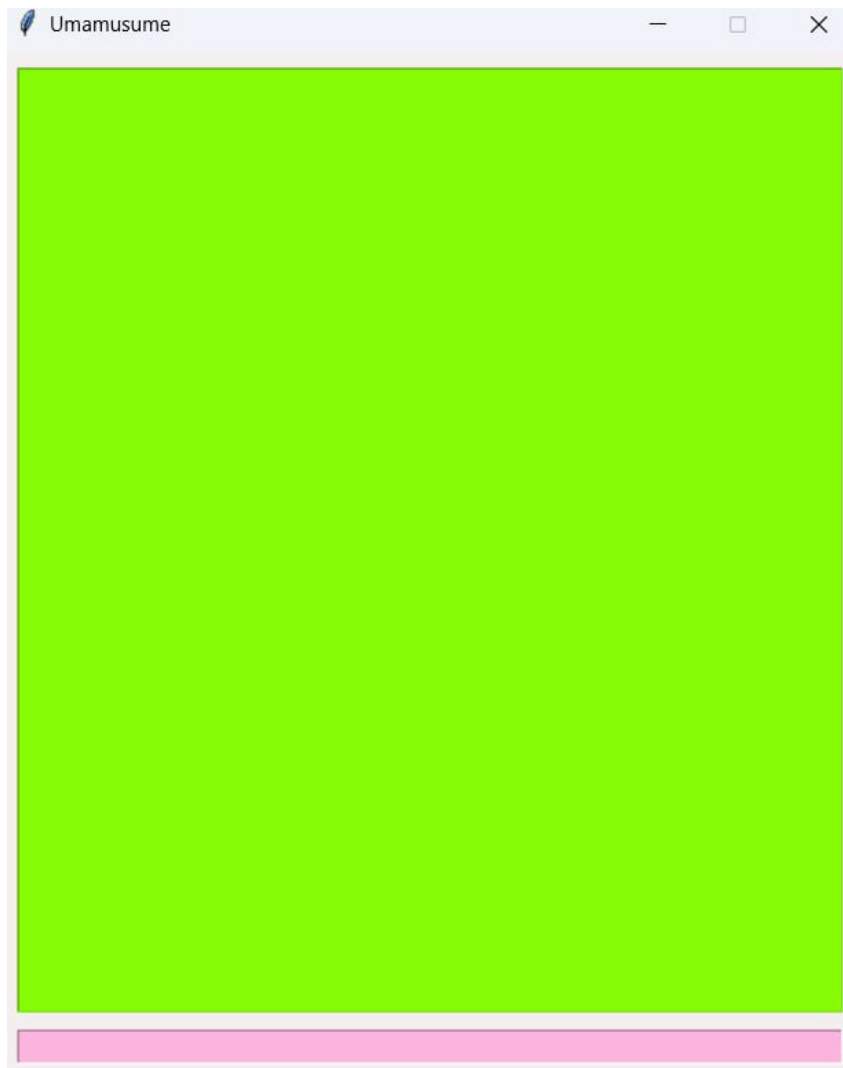
5. Шаги выполнения:

1. Визуально оценить цвет фона терминала.

6. Ожидаемый результат.

Фон терминала имеет розовый цвет (согласно ТЗ).

7. Фактический результат.



8. Статус.

Failed.

1. Идентификатор.

ТС-009

2. Название.

Проверка корректности цвета текста вывода команд.

3. Описание.

Убедиться, что цвет текста вывода команд соответствует спецификации ТЗ.

4. Предварительные условия:

а) Терминал запущен.

5. Шаги выполнения:

1. Выполнить любую команду с стандартным выводом (например, `ls`).

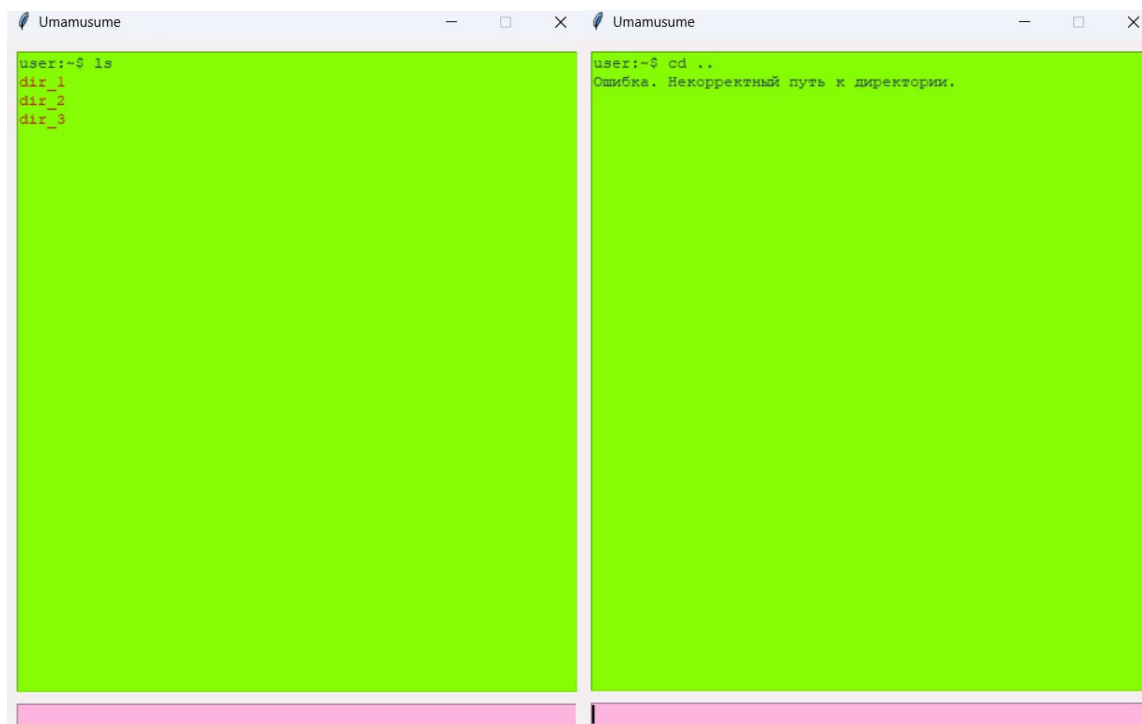
2. Выполнить любую команду, завершающуюся ошибкой (например, `ls /несуществующая_директория`).

3. Визуально оценить цвет текста в обоих случаях.

6. Ожидаемый результат.

Цвет текста вывода команд соответствует спецификации ТЗ (например, стандартный вывод - черный/белый, ошибки - красный).

7. Фактический результат.



```
user:~$ ls
dir_1
dir_2
dir_3

user:~$ cd ..
Ошибка. Некорректный путь к директории.
```

8. Статус.

Failed.

Анализ документации

В ходе анализа документации к эмулятору shell выявлен ряд существенных недочётов, которые могут привести к недопониманию и некорректной реализации проекта.

Отсутствие деталей — самая критическая проблема документации. Формат конфигурационного CSV-файла описан лишь в общих чертах.

Самые серьёзные ошибки содержатся в описании команд. Команда `tail`, которая предназначена для просмотра конца файла, ошибочно описана как инструмент для создания файла. Команда `chown`, меняющая владельца файла, якобы выводит его содержимое. Команда `find`, используемая для поиска файлов, по документации выводит содержимое файла. Эти грубые ошибки полностью искажают суть программного продукта.

Команды

В квадратных скобках опциональные настройки команд.

`ls <path>` - Список файлов и директорий

`cd <path>` - Смена директории

`exit` - Выход из эмулятора

`tail [-n int] <path>` - Создание файла

`chown <user> <path>` - Вывод содержимого файла

`find <name>` - Вывод содержимого файла

Дополнительным недостатком является отсутствие спецификации по уровню поддержки команд. Не ясно, реализованы ли все возможные флаги для `ls`, `find` и других утилит или лишь базовый функционал.

Заключение

На основе проведённого тестирования и анализа документации можно сделать вывод о том, что представленный программный продукт — эмулятор shell — имеет существенные недостатки как в реализации, так и в сопроводительной документации.

Общая оценка качества программного продукта является неудовлетворительной. Несмотря на частично реализованный базовый функционал, продукт содержит критические ошибки, которые нарушают его основное предназначение — корректную эмуляцию командной оболочки. Невозможность правильной навигации по файловой системе (команда `'cd'`) делает программу практически бесполезной. Серьёзные логические ошибки в таких ключевых командах, как `'chown'` (изменяет владельца на несуществующую учётную запись) и `'find'` (выводит мусорные данные), а также некорректная работа `'tail'` свидетельствуют о поверхностном тестировании и отсутствии валидации входных данных и результатов выполнения команд. Интерфейс продукта не соответствует требованиям технического задания по цветовой схеме, что указывает на пренебрежение к требованиям заказчика.

Качество документации также признаётся низким. README содержит грубые фактические ошибки в описании команд (`'tail'` — создание файла, `'chown'` — вывод содержимого), что вводит в заблуждение и свидетельствует о невнимательности при его составлении. Критическим недостатком является отсутствие деталей в ключевых разделах: не определён формат и структура конфигурационного CSV-файла, не конкретизирован ожидаемый уровень поддержки флагов команд. Требование к «режиму GUI» сформулировано двусмысленно и допускает множество трактовок.

Вывод о соответствии ТЗ: Программный продукт не соответствует требованиям, изложенным в техническом задании. Критерии приемки, включающие прохождение всех тест-кейсов и соответствие интерфейса, не выполнены.

Рекомендации по улучшению:

1. Для программного продукта:

- Высокий приоритет: Исправить критическую ошибку с командой `'cd'` для обеспечения базовой навигации.
- Высокий приоритет: Реализовать валидацию входных данных для команд `'chown'` (проверка существования пользователя) и `'find'`.
- Высокий приоритет: Привести цветовую схему интерфейса (фон, цвет текста, цвет ошибок) в строгое соответствие с ТЗ.

- Средний приоритет: Исправить логику работы команд `tail` и `find` для обеспечения корректного вывода.
- Низкий приоритет: Добавить отображение текущей директории в приглашении командной строки (prompt) для улучшения пользовательского опыта.

2. Для документации:

- Обязательно: Немедленно исправить описание команд `tail`, `chown` и `find` в README, приведя их в соответствие с общепринятой семантикой UNIX.
- Обязательно: Детально описать формат конфигурационного файла (например: «Файл `config.csv` содержит одну строку с тремя полями, разделёнными запятой: `username, path_to_zip_archive, path_to_old_archive`»).
- Обязательно: Конкретизировать требование к GUI, указав, что подразумевается под этим режимом (например: «Эмулятор должен запускаться в отдельном графическом окне, имитирующем терминал»).
- Рекомендуется: Уточнить в ТЗ уровень поддержки команд (например, указать, какие именно флаги должны реализовываться для `ls`, `find` и т.д.).

Продукт может быть допущен к повторной приемке только после устранения всех критических замечаний и предоставления исправленной, детализированной документации.