



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)

ПРАКТИЧЕСКАЯ РАБОТА №1

по дисциплине

«Тестирование и верификация программного обеспечения»

Выполнили:

Студенты группы ИКБО-50-23

Зиненко М. А.

Бобров Т. Д.

Кувабин К. М.

Петрокин Д. С.

Котков Д.И.

Проверил:

ассистент Ильичев Г. П.

Москва 2025 г.

СОДЕРЖАНИЕ

«Тестирование и верификация программного обеспечения»	1
ФОРМУЛИРОВКА ЗАДАНИЯ	4
ВЫПОЛНЕНИЕ ЗАДАНИЯ	6
Техническое задание на разработку To-Do-List.	6
1. Введение	6
2. Общие сведения	6
2.1 Назначение	6
2.2 Краткий обзор	7
3. Назначение и цели создания системы	8
3.1 Назначение системы	8
3.2 Цели создания системы	9
4. ХАРАКТЕРИСТИКА ОБЪЕКТА АВТОМАТИЗАЦИИ	10
4.1.1 Объект автоматизации	10
4.1.2 Условия эксплуатации и характеристики среды	10
5. Требования к автоматизированной системе	11
5.1 Функциональные требования	11
5.1.1 Управление задачами (CRUD)	11
5.1.2 Поиск, фильтрация и сортировка задач	11
5.1.3 Визуальная индикация статусов и сроков	11
5.1.5 Управление интерфейсом	12
5.2 Нефункциональные требования	12
5.2.1 Надежность	12
5.2.2 Доступность	12
5.2.3 Безопасность	12
6. Состав и содержание работ по созданию (развитию) системы	13
7. Порядок разработки автоматизированной системы	14
8. Порядок контроля и приемки автоматизированной системы	14
9. Требования к составу и содержанию работ по подготовке объекта автоматизации к вводу автоматизированной системы в действие	14
10. Требования к документации	15
11. Источники разработки	15

ФОРМУЛИРОВКА ЗАДАНИЯ

Часть 1

1. Для выполнения задания, группа должна разделиться на команды численность 2-5 человек и придумать себе название. Команда выбирается на весь курс дисциплины.

2. Каждая из команд для выполнения 1 части практического задания подготавливает следующие материалы:

- рассматриваемый Программный продукт;
- техническое задание под рассматриваемый программный продукт

и дополнительную документацию при необходимости.

Спецификации на предыдущие материалы:

- программный продукт должен быть исполняемым, удобным и читаемым для выполнения другой командой;
- должно присутствовать полное и достаточное описание для запуска программного продукта в случае его нетривиального исполнения;
- программный продукт должен содержать в себе определенное количество ошибок при исполнении от 5-8;
- техническое задание должно полностью описывать функциональные возможности программного продукта;
- описание в каждом пункте технического задания должно быть кратким, лаконичным и не вызывать двояких трактовок;
- техническое задание составляется по шаблону.

Часть 2

На момент выполнения второй части практического задания у команд в группе должно быть выполнена часть 1 практического задания.

1. Команда берет программный продукт, созданный другой командой.
2. Каждая команда изучает программный продукт, выбранный для тестирования, изучает ТЗ, документацию.
3. Анализирует полноту и качество описания ПП в ТЗ и документации.
4. Команда должна протестировать ПП, используя метод «черного ящика».
5. Исходя из конечного результата тестирования командой составляется отчет, по проделанной работе.
6. Командная работа подразумевает четкое разделение ролей. При необходимости каждый студент должен объяснить свою роль, в выполнении практической работы.

ВЫПОЛНЕНИЕ ЗАДАНИЯ

Техническое задание на разработку To-Do-List.

1. Введение

Тестирование программного обеспечения охватывает множество аспектов, включая функциональное, нагрузочное и регрессионное тестирование. Каждый из этих типов тестирования имеет свои цели и методы, что позволяет выявить различные виды дефектов на разных этапах разработки. Например, функциональное тестирование направлено на проверку корректности работы всех функций программы — таких как добавление, редактирование и удаление задач, фильтрация по категориям и приоритетам, а также экспорт и импорт данных в формате CSV — и их соответствие заявленным требованиям. Нагрузочное тестирование, хотя и менее критично для локального приложения, может быть применено для оценки производительности интерфейса при работе с тысячами задач. Современные инструменты автоматизации тестирования значительно повышают эффективность процесса проверки: автоматизированные UI-тесты позволяют быстро воспроизводить сценарии взаимодействия пользователя, выявлять регрессии после изменений кода и минимизировать влияние человеческого фактора. Однако, несмотря на все преимущества автоматизации, ручное тестирование по-прежнему остаётся важной частью QA-процесса — особенно при оценке удобства интерфейса, логики навигации, корректности отображения цветовых подсказок (просроченные, сегодня, высокий приоритет) и интуитивности контекстного меню и горячих клавиш. Только сочетание автоматизированных и ручных методов позволяет обеспечить высокое качество пользовательского опыта и надёжность работы приложения в реальных условиях эксплуатации.

2. Общие сведения

2.1 Назначение

Программное приложение To-Do List.

2.2 Краткий обзор

To-Do List на Python с Tkinter — это настольное приложение с графическим интерфейсом, позволяющее пользователю эффективно управлять личными и рабочими задачами. С помощью интуитивно понятного визуального интерфейса пользователь может добавлять, редактировать и удалять задачи, отмечать их как выполненные, фильтровать по категориям и приоритетам, а также сортировать по дате или названию. Дополнительно реализованы функции поиска, пакетных операций (например, массовое изменение статуса или удаление выполненных задач), а также экспорт и импорт данных в формате CSV для резервного копирования или переноса между устройствами. Приложение создано для пользователей, знакомых с базовыми принципами работы на компьютере и стандартными элементами управления оконными приложениями. Помимо этого, приложение обеспечивает стабильную работу даже при большом количестве задач благодаря использованию SQLite для хранения данных и оптимизированному отображению в таблице. Также предусмотрены горячие клавиши и контекстное меню для ускорения работы — например, Ctrl+N для быстрого добавления задачи или ПКМ для вызова меню действий. Для опытных пользователей доступна возможность ручного редактирования CSV-файлов, что расширяет гибкость управления данными.

2.3 Разработчики

Состав команды: Бобров Т. Д., Кувабин К. М., Петрокин Д. С., Зиненко М. А., Котков Д.И..

2.4 Заказчики

Преподаватель

2.5 Основание для разработки

Договор № 123.45 от 06.09.2024 на разработку автоматизированной системы.

3. Назначение и цели создания системы

1.1 Назначение системы

Приложение «To-Do List» на Tkinter предназначено для помощи пользователям в организации личных и профессиональных задач, управлении временем и повышении продуктивности за счёт централизованного хранения и визуального контроля списка дел. Основные функции приложения включают:

1. Управление задачами: Пользователь может добавлять новые задачи, редактировать существующие, удалять ненужные и отмечать выполненные — обеспечивая гибкий и интуитивный контроль над своим списком дел.

2. Фильтрация и сортировка: Приложение позволяет фильтровать задачи по статусу (активные/выполненные), приоритету (низкий, средний, высокий) и категории, а также сортировать их по дате создания, сроку выполнения или названию — что помогает быстро находить нужные задачи даже в большом списке.

3. Визуальная индикация: Для удобства восприятия реализована цветовая подсветка: просроченные задачи выделяются красным, задачи со сроком «сегодня» — оранжевым, задачи с высоким приоритетом — жирным шрифтом. Это позволяет пользователю мгновенно оценить срочность и важность дел.

4. Пакетные операции и горячие клавиши: Для ускорения работы реализованы пакетные действия — например, массовое удаление выполненных задач или изменение статуса у нескольких задач одновременно. Также доступны горячие клавиши (Ctrl+N, Ctrl+E, Delete, Space и др.) и контекстное меню по правому клику — что делает взаимодействие с приложением быстрым и удобным.

Приложение предназначено для пользователей, которым важно структурировать свои задачи, отслеживать сроки и контролировать прогресс выполнения дел. Оно обеспечивает простоту использования, стабильность работы благодаря локальному хранению данных в SQLite, и не требует

подключения к интернету — что делает его надёжным инструментом для ежедневного использования как дома, так и в офисе.

1.2 Цели создания системы

Целями создания приложения «To-Do List» на Tkinter являются:

- Обеспечение пользователей удобным и централизованным инструментом для управления ежедневными задачами, позволяющим быстро добавлять, отслеживать и завершать дела без необходимости использования сложных систем или облачных сервисов.
- Автоматизация рутинных операций по управлению задачами, таких как сортировка по приоритету, фильтрация по категориям, массовое изменение статуса или удаление выполненных задач — что позволяет пользователю сосредоточиться на выполнении дел, а не на их организации.
- Повышение точности и прозрачности контроля выполнения задач за счёт визуальной индикации сроков (просрочено, сегодня, скоро) и приоритетов (жирный шрифт для High), а также сохранения полной истории изменений в локальной базе данных.
- Обеспечение пользователей возможностью быстрого восстановления и переноса данных благодаря поддержке импорта и экспорта в формате CSV — что гарантирует сохранность информации даже при смене устройства или переустановке приложения.

Для достижения этих целей бот должен:

- Иметь простой, интуитивно понятный графический интерфейс, адаптированный под работу с клавиатурой и мышью, с поддержкой горячих клавиш и контекстного меню для ускорения взаимодействия.
- Быть стабильным и производительным даже при работе с тысячами задач, благодаря использованию SQLite для хранения данных и оптимизированному отображению в компоненте Treeview.
- Обеспечивать надёжное сохранение данных локально на устройстве пользователя, исключая риск потери информации из-за сбоя сети или сторонних сервисов.
- Быть легко расширяемым для добавления новых функций в будущем — таких как напоминания, синхронизация между устройствами, поддержка подзадач или интеграция с календарём — благодаря модульной архитектуре и чистому коду.

4. ХАРАКТЕРИСТИКА ОБЪЕКТА АВТОМАТИЗАЦИИ

1.2.1 Объект автоматизации

Объектом автоматизации является клиентское приложение для управления задачами, реализованное на Python с использованием Tkinter и SQLite. Система обеспечивает персистентное хранение данных, гибкие механизмы поиска и фильтрации, интерактивную сортировку, многоуровневое управление статусами задач и поддержку пользовательских настроек через графический интерфейс. Программа не требует установки внешних зависимостей и работает автономно.

1.2.2 Условия эксплуатации и характеристики среды

Программное обеспечение предназначено для эксплуатации в условиях домашней или офисной среды на персональных компьютерах. Взаимодействие с системой осуществляется через графический интерфейс пользователя (GUI), реализованный на базе библиотеки Tkinter. Программа не требует подключения к интернету и может работать полностью автономно.

Система разработана на языке Python 3 с использованием стандартных библиотек: tkinter, sqlite3, csv, datetime. Для хранения данных используется встроенная SQLite-база данных, размещаемая локально в рабочей директории приложения. Внешние зависимости отсутствуют — программа распространяется и запускается как один исполняемый файл.

Система рассчитана на эксплуатацию в стандартных домашних, без особых требований к окружающей среде. Основные требования включают:

- Наличие стабильного интернет-соединения для доступа к веб-интерфейсу системы.

- Работа на серверном оборудовании, соответствующем требованиям для запуска приложений на платформе Java Spring Boot.
- Совместимость с операционными системами, поддерживающими Java.

5. Требования к автоматизированной системе

1.3 Функциональные требования

5.1.1 Управление задачами (CRUD)

- Пользователь должен иметь возможность создавать новые задачи с указанием названия, описания, категории, приоритета и срока выполнения.
- Пользователь должен иметь возможность удалять задачи как по одной, так и пакетно (например, массовое удаление выполненных задач).

5.1.2 Поиск, фильтрация и сортировка задач

- Пользователь должен иметь возможность сортировать задачи по любому полю (название, категория, приоритет, срок, статус, дата создания) — как по возрастанию, так и по убыванию — через клик по заголовку столбца в таблице.
- Пользователь должен иметь возможность фильтровать задачи по статусу («Активные», «Выполненные», «Все»), приоритету («Low», «Medium», «High») и категории.

5.1.3 Визуальная индикация статусов и сроков

- Система должна визуально подсвечивать задачи в зависимости от срока выполнения
- Задачи с приоритетом «High» должны отображаться жирным шрифтом.
- Выполненные задачи должны отображаться затенённым текстом.

5.1.4 Экспорт и импорт данных

- Пользователь должен иметь возможность экспортировать все отображаемые задачи (с учётом фильтров и поиска) в файл формата CSV (UTF-8) для резервного копирования или переноса.
- Пользователь должен иметь возможность импортировать задачи из CSV-файла, с автоматической валидацией формата даты и приоритета.
- При импорте система должна игнорировать некорректные записи, но уведомлять пользователя о количестве успешно импортированных задач.

5.1.5 Управление интерфейсом

- Пользователь должен иметь доступ к контекстному меню (ПКМ) в списке задач с быстрыми действиями: добавить, редактировать, удалить, отметить, экспорт, импорт.
- Пользователь должен иметь возможность просматривать статистику по задачам: общее количество, активные, выполненные, просроченные — в реальном времени.

1.4 Нефункциональные требования

5.1.6 Надежность

- Приложение должно корректно запускаться и работать на всех поддерживаемых операционных системах (Windows, macOS, Linux) при наличии установленного Python 3.7+.
- Программа должна устойчиво обрабатывать ошибки ввода данных (например, некорректные даты, пустые названия задач) без аварийного завершения — с выводом понятных пользователю предупреждений.
- Приложение должно корректно восстанавливать состояние интерфейса (фильтры, сортировка, поиск) после перезапуска.

1.4.1 Доступность

- Приложение должно быть полностью автономным — не требует подключения к интернету, серверов, облачных сервисов или внешних API.

- Программа должна запускаться как один файл без необходимости установки дополнительных библиотек или зависимостей (кроме стандартной поставки Python с Tkinter).
- Приложение должно быть доступно для использования сразу после запуска — без регистрации, авторизации или настройки ролей.

1.4.2 Безопасность

- Поскольку приложение является однопользовательским и локальным, система ролей (ADMIN/CLIENT) не предусмотрена и не требуется.
- Все данные хранятся локально в файле tasks.db в директории запуска — доступ к ним определяется правами операционной системы.
- Программа не передаёт данные во внешние сети и не содержит уязвимостей, связанных с удалённым доступом или инъекциями (SQL-инъекции предотвращены использованием параметризованных запросов).
- При импорте данных из CSV предусмотрена валидация входных данных для предотвращения повреждения структуры базы.

2. Состав и содержание работ по созданию (развитию) системы

- Разработка технического задания:
 - Содержание: Определение требований, создание и согласование технического задания.
 - Результат: Утвержденное техническое задание.
- Проектирование системы:
 - Содержание: Разработка архитектуры системы, проектирование базы данных и интерфейсов.
 - Результат: Документация по архитектуре, проект базы данных, макеты интерфейсов.
- Разработка программного обеспечения:
 - Содержание: Написание и тестирование кода.
 - Результат: Рабочее программное обеспечение.
- Интеграция и тестирование:

- Содержание: Интеграция модулей, функциональное и нагрузочное тестирование.
- Результат: Протестированная система.
- Внедрение системы:
 - Содержание: Установка, настройка, обучение пользователей.
 - Результат: Рабочая система и обученные пользователи.
- Поддержка и сопровождение:
 - Содержание: Техническая поддержка, обновления.
 - Результат: Стабильная работающая система.

3. Порядок разработки автоматизированной системы

Процесс создания системы проходит через несколько ключевых фаз: анализ требований, проектирование, программирование, тестирование и внедрение. Перед началом каждого нового этапа требуется получение одобрения от заказчика. Важным аспектом являются промежуточные проверки для обеспечения соответствия системы установленным требованиям.

Необходимые этапы при разработки автоматизированной системы: аналитический, проектный, программный, тестирование и внедрение.

4. Порядок контроля и приемки автоматизированной системы

Контроль и приемка системы включают функциональные, интеграционные и нагрузочные испытания, проводимые в соответствии с действующими стандартами. Испытания охватывают проверку всех компонентов системы на соответствие требованиям технического задания. Приемка выполняется поэтапно: документация проверяется и утверждается в установленные сроки, а участие в приемке принимает заказчик, разработчик и, при необходимости, сторонние эксперты. Приемочная комиссия, состоящая

из представителей заказчика и разработчика, утверждает результаты и принимает решение о готовности системы к эксплуатации.

5. Требования к составу и содержанию работ по подготовке объекта автоматизации к вводу автоматизированной системы в действие

Для подготовки объекта автоматизации к внедрению системы необходимо выполнить ряд ключевых мероприятий. Это включает в себя: подготовку информации для обработки в системе, внесение изменений в объект автоматизации, обеспечение соответствующих условий для функционирования системы, создание необходимых служб и подразделений, а также организацию обучения персонала. Эти мероприятия должны быть согласованы с заказчиком и выполнены до начала эксплуатации системы, а их детали уточняются на стадии разработки и опытной эксплуатации.

6. Требования к документации

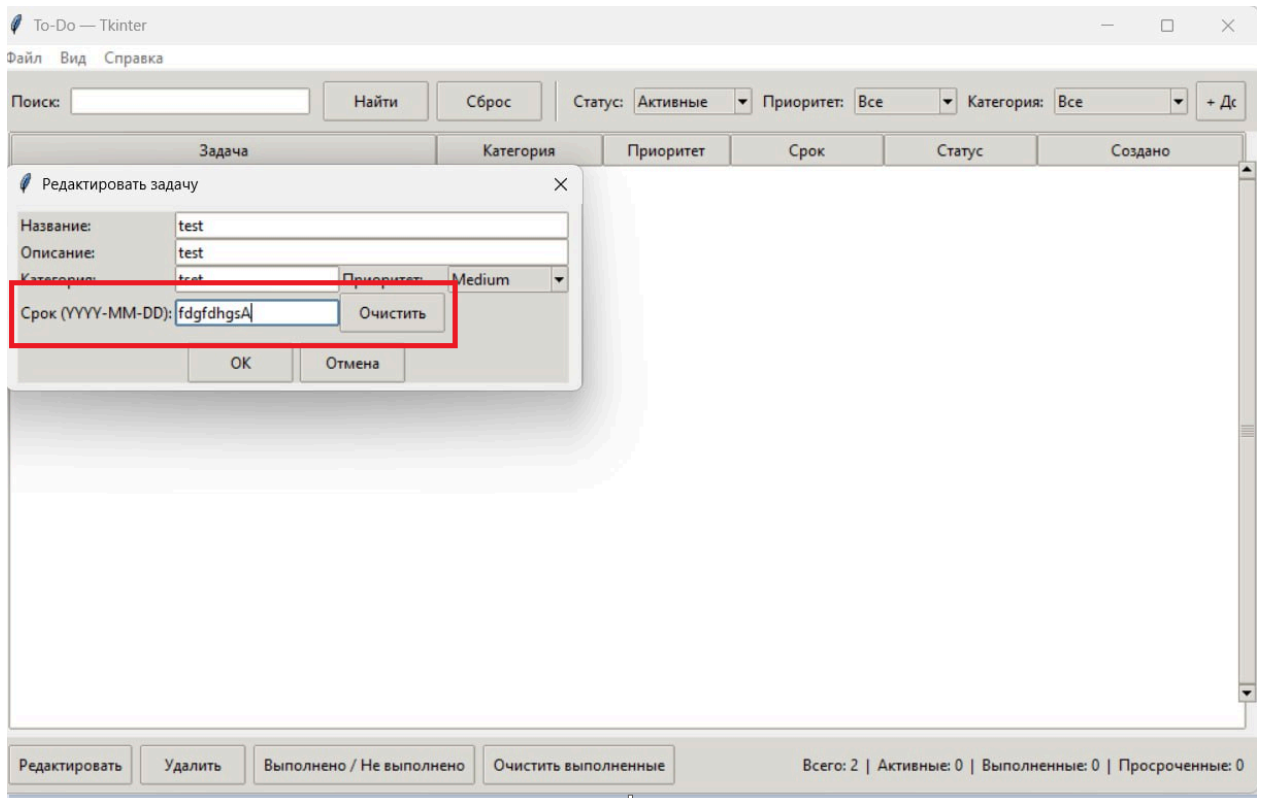
Для автоматизированной системы требуется разработка и поддержка следующих документов: техническое задание, проектная документация, руководство пользователя и эксплуатационная документация. Все документы должны соответствовать стандартам ГОСТ и внутренним нормативам. Документация будет предоставляться как в бумажном, так и в электронном виде.

7. Источники разработки

- Договор № 123.45 от 06.09.2024;
- ГОСТ 34.602 – 2020 "Техническое задание на создание автоматизированной системы".

Ошибки, заложенные в программном продукте

Пользователь может ввести некорректную дату, которая сохраняется и отображается неправильно (в столбце "Статус"), из-за отсутствия строгой валидации и ошибки в отображении данных. (рисунок 1).



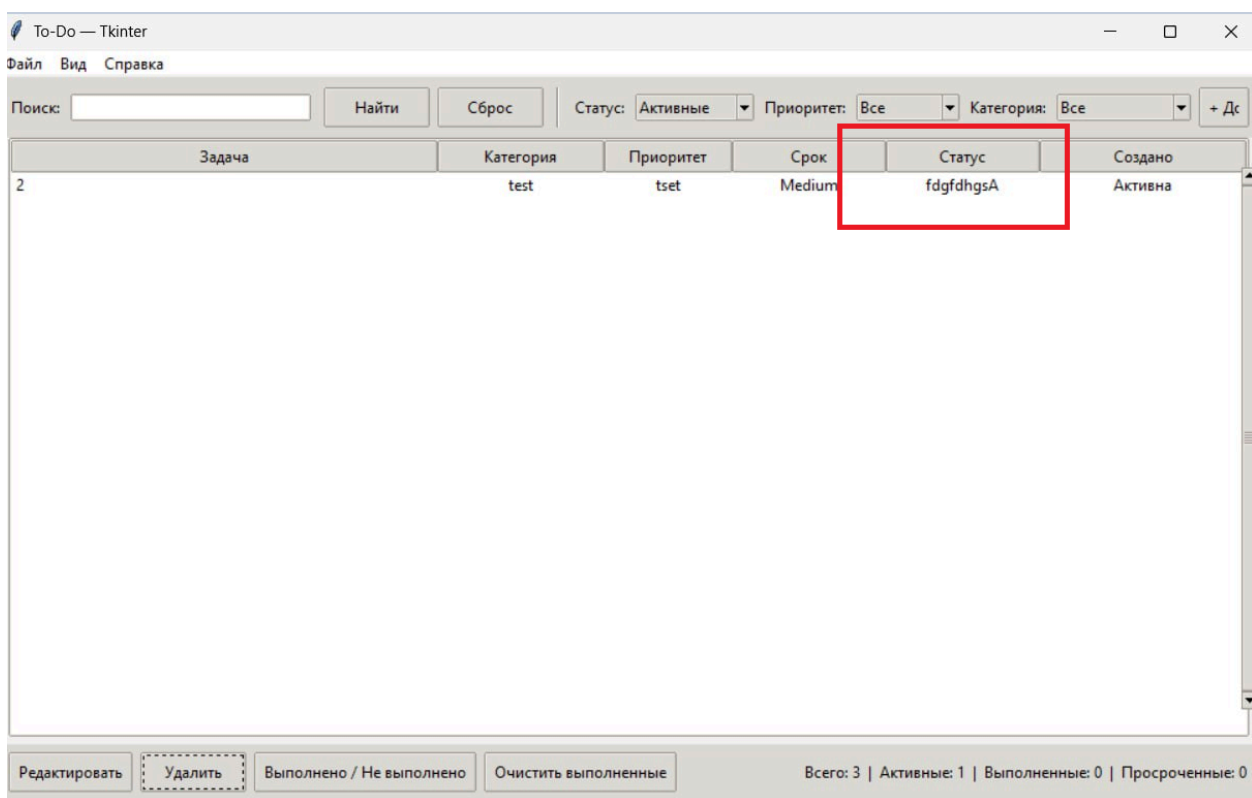


Рисунок 1 – Пример первой некорректной работы

Некорректное отображение общего количества задач, сумма выполненных, активных и просроченных отображалось неверно в графе "всего" (рисунок 2).

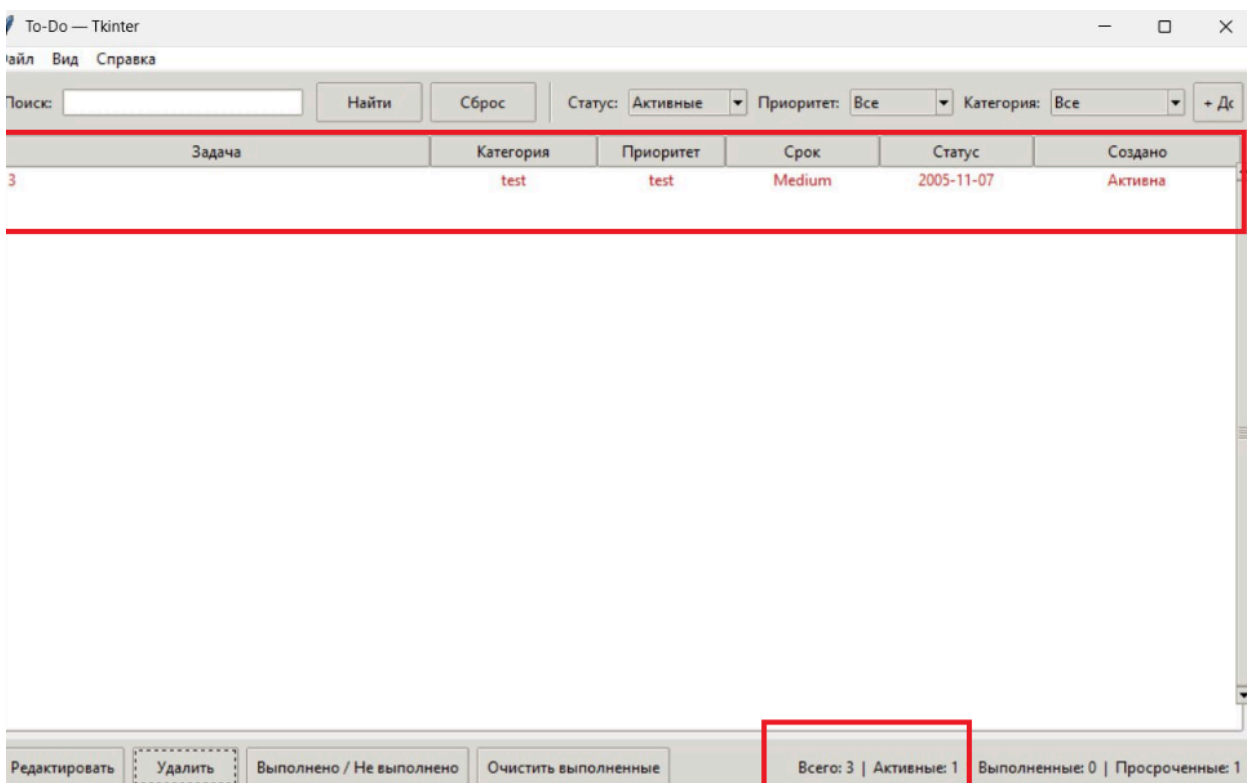


Рисунок 2 – Пример второй некорректной работы

Некорректное отображение колонок, значения сдвинуты на одну колонку вправо (рисунок 3).

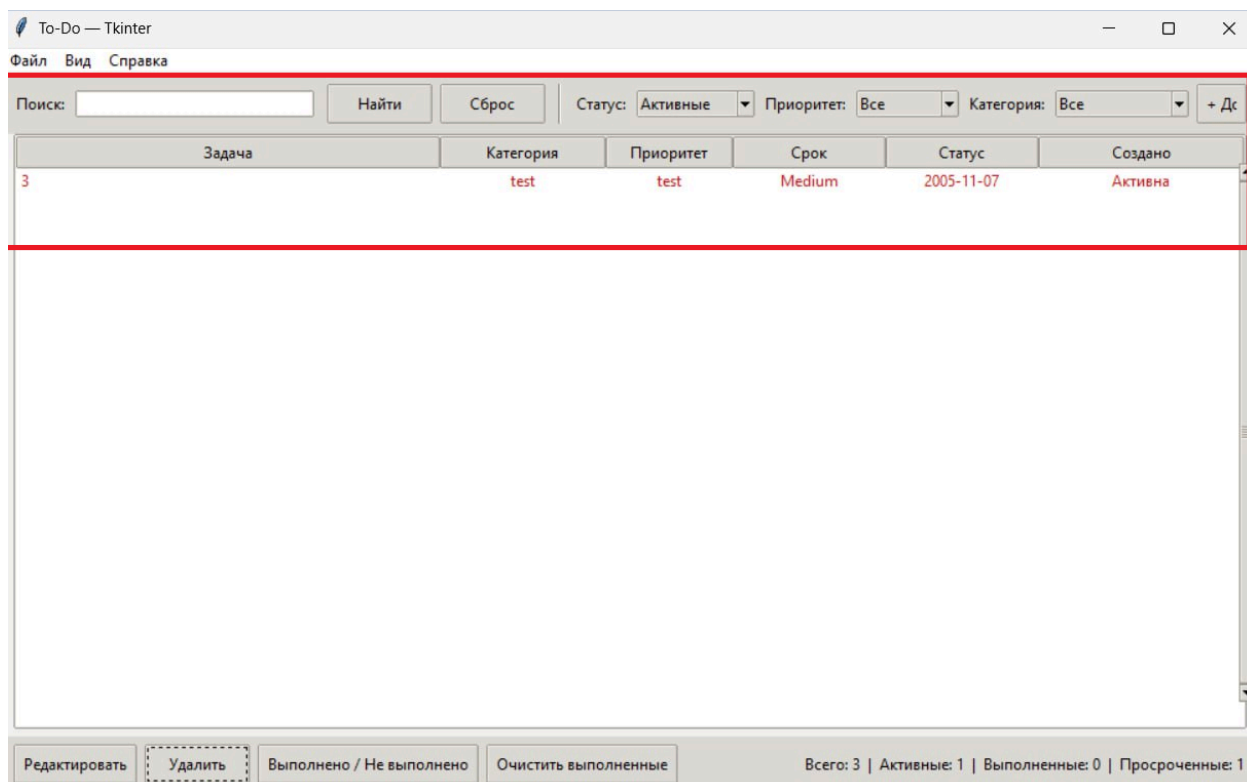
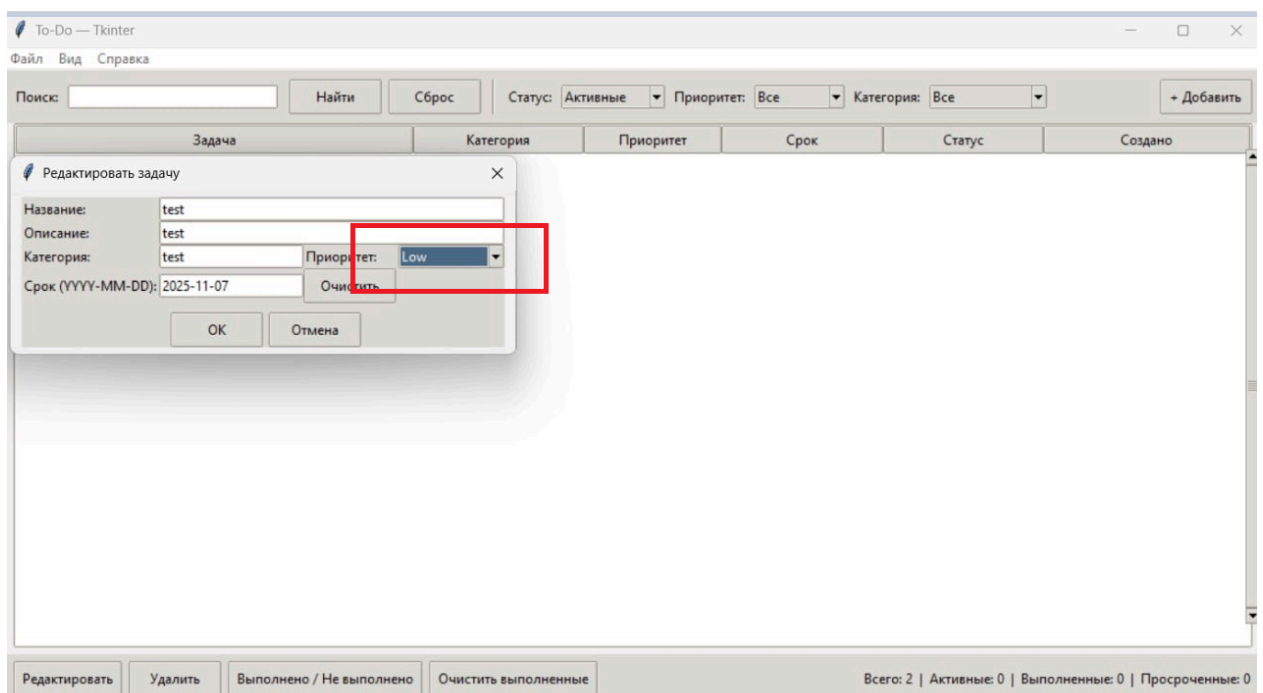


Рисунок 3 – Пример третьей некорректной работы

Несмотря на то что пользователь выбирает значение приоритета "Low" при редактировании задачи, в списке задач в столбце "Срок" отображается значение "Medium", а не выбранный приоритет.

Это указывает на ошибку логики отображения данных — значение приоритета некорректно отображается в столбце "Срок", что создает путаницу и снижает надежность интерфейса. Возможно, произошла путаница между полями "Приоритет" и "Срок" при отображении данных в таблице (рисунок 4).



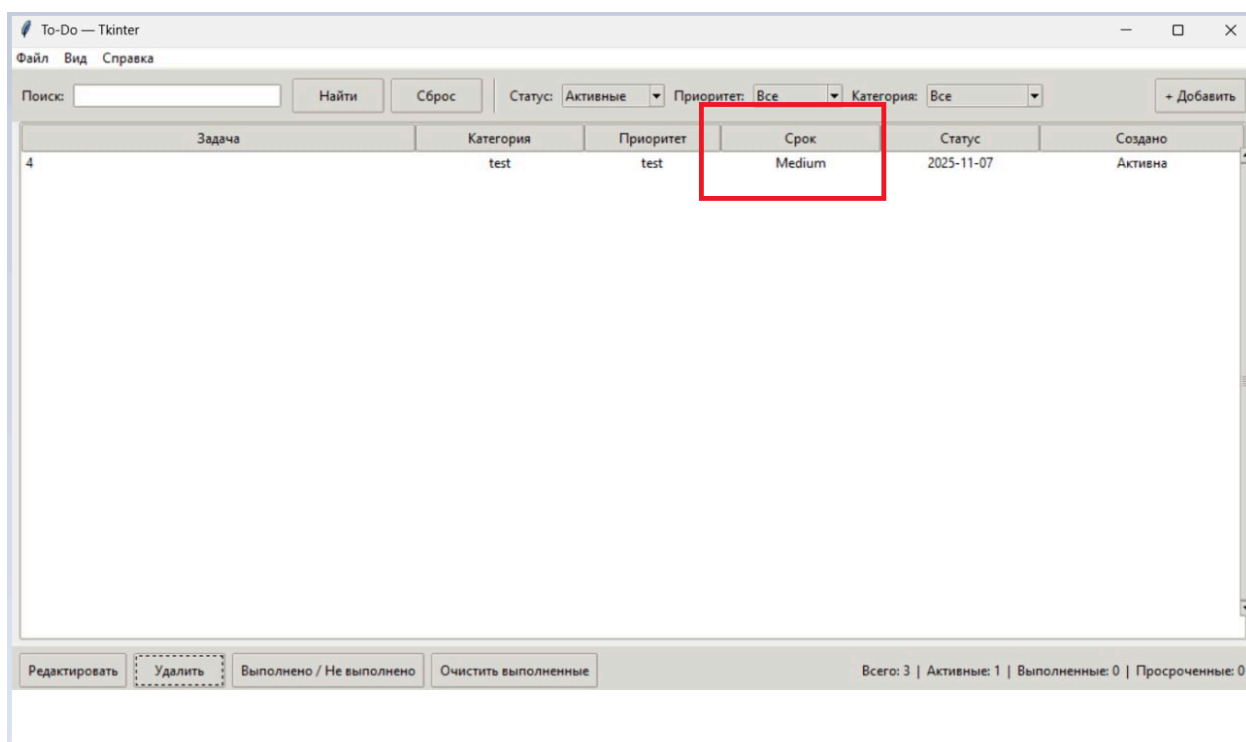


Рисунок 4 – Пример четвертой некорректной работы

При нажатии на кнопку “Добавить” ничего не происходит (рисунок 5).

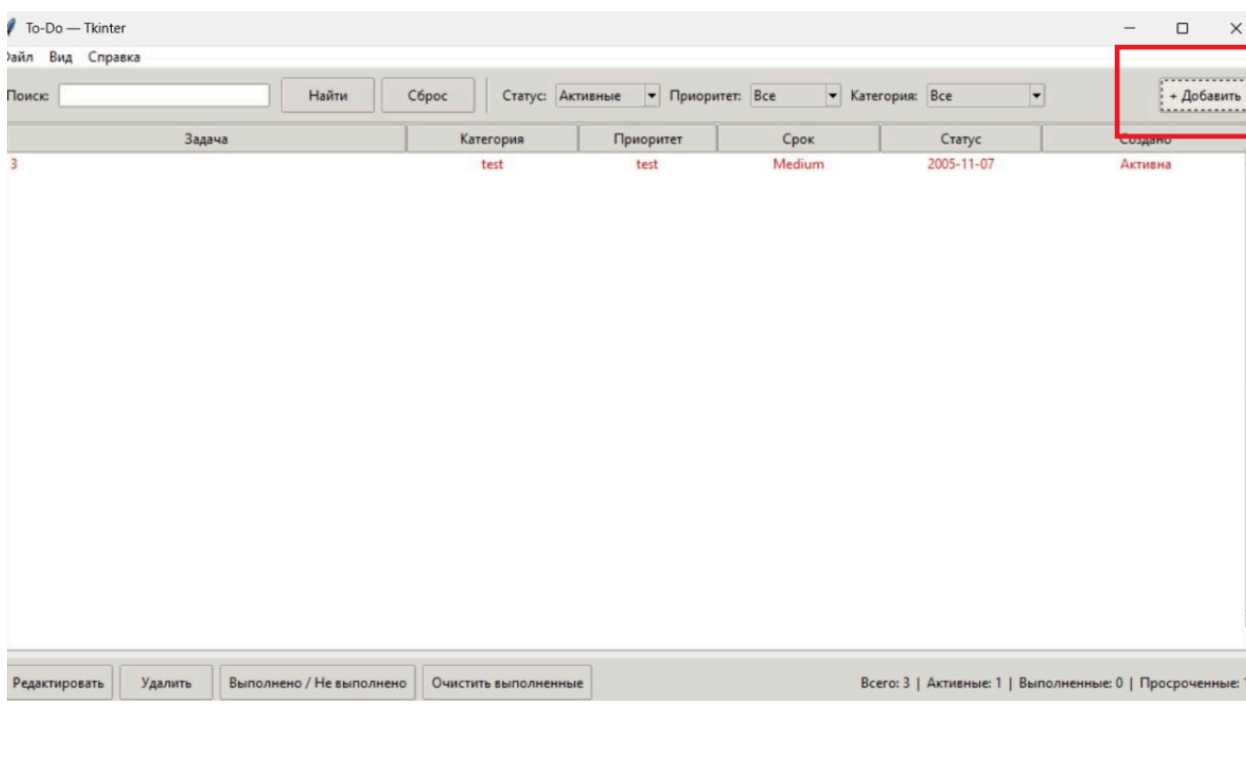


Рисунок 5 – Пример пятой некорректной работы

ПРОВЕРКА ТЗ И ДОКУМЕНТАЦИИ ПРОГРАММНОГО ПРОДУКТА ДРУГОЙ КОМАНДЫ

Введение

Программа "Калькулятор участка" представляет собой десктопное приложение с графическим интерфейсом. Она предназначена для выполнения расчетов, связанных с земельными участками, включая определение площади, стоимости участка, перевода площади в сотки, расчета максимальной площади застройки и стоимости ограждения.

Области применения:

- Недвижимость и оценка имущества
- Планирование строительства
- Ландшафтный дизайн и благоустройство:
- Сельское хозяйство и землевладение
- Образовательные цели

Основания для разработки

Разработка программы "Калькулятор участка" была инициирована в ответ на потребность в простом и доступном инструменте для выполнения расчетов, связанных с земельными участками. Проект направлен на упрощение и ускорение процесса принятия решений в области управления земельными ресурсами, а также на повышение точности расчетов за счет минимизации человеческого фактора.

Исходные документы:

- Документация Python и Tkinter
- Методические рекомендации по определению стоимости земельных участков

Назначение разработки

Программа предназначена для автоматизации расчётов параметров земельного участка, что позволит:

- Сократить время расчёта стоимости, площади и других характеристик участка по сравнению с ручными вычислениями.
- Исключить ошибки при вычислении площади, стоимости и параметров ограждения.
- Обеспечить наглядное представление результатов расчётов (площадь, стоимость, максимальная площадь застройки).
- Предоставить удобный интерфейс для работы с различными типами расчётов (площадь, сотки, стоимость, ограждение, застройка).

Требования к программе

Функциональные требования

- Вычисление площади по заданным длине и ширине (в метрах)
- Конвертация площади из квадратных метров в сотки
- Вычисление общей стоимости на основе площади и стоимости квадратного метра.
- Расчёт допустимой площади застройки (30% от общей площади).
- Вычисление стоимости ограждения по периметру участка и цене за метр.
- Выбор команды через выпадающее меню для активации соответствующей формы ввода.
- Отображение результатов расчётов в текстовом формате.
- Динамическое обновление интерфейса при выборе команды.

Требования к надежности

- При некорректном вводе данных (пустые поля, текст вместо числа, отрицательные значения) программа не завершается аварийно, а выдаёт сообщение об ошибке.
- Ошибки обработки данных (например, деление на ноль, ввод нечисловых значений) должны перехватываться обработчиками исключений.
- Программа должна сохранять работоспособность при любых пользовательских ошибках во вводе данных.
- В случае возникновения ошибок программа должна предоставлять пользователю возможность продолжить работу без перезапуска.
- Логика расчётов должна гарантировать, что полученные результаты всегда корректны для введённых значений.

Условия эксплуатации

- Операционная система: Windows 7/8/10/11, Linux, MacOS
- Минимальные системные требования: процессор 1 ГГц и выше; оперативная память 512 МБ; дисковое пространство не менее 100 МБ; экран с разрешением от 1024×768.
- Обязательное наличие интерпретатора Python версии 3.6 и выше
- Наличие библиотеки tkinter в составе Python

Требования к совместимости

- Совместимость с операционными системами семейства Windows, Linux, MacOS
- Поддержка кодировки UTF-8 для корректного отображения символов
- Независимость от дополнительного программного обеспечения (кроме Python)

Требования к интерфейсу

Основные элементы интерфейса

Главное окно с заголовком «Калькулятор участка».

Выпадающий список для выбора команды:

«1» — расчет стоимости участка;

«2» — перевод площади в сотки;

«3» — расчет стоимости по площади и цене за кв.м;

«4» — расчет максимальной площади застройки;

«5» — расчет стоимости ограждения.

Текстовые поля для ввода параметров (длина, ширина, площадь, стоимость). Поля должны быть активными и доступны для ввода только числовых значений.

Кнопка «Рассчитать» — заметная, расположена в нижней части окна, активируется нажатием мыши.

Область для отображения результата должна иметь увеличенный шрифт (не менее 12 pt) и быть расположена под кнопкой.

Все сообщения об ошибках должны отображаться в отдельном диалоговом окне с информативным текстом.

Интерфейс должен поддерживать работу в стандартных разрешениях экрана (от 1024×768).

Критерии приемки

- Успешное выполнение не менее 95 % тест-кейсов (проверка корректных и некорректных вводов).
- Корректный расчет площади, стоимости и других параметров в пределах допустимой погрешности (не более 0.01).
- Программа устойчива к ошибкам ввода и не завершает работу аварийно.
- Интерфейс отображается корректно на поддерживаемых ОС (Windows, Linux, macOS).
- Время отклика программы на действие пользователя не превышает 1 секунды.

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

Методы тестирования

- Функциональное тестирование всех элементов интерфейса
- Тестирование корректности вычислений
- Тестирование обработки ошибок и исключительных ситуаций
- Тестирование пользовательского интерфейса на удобство использования

Приемочные испытания

- Проверка на соответствие техническому заданию
- Тестирование на различных операционных системах
- Проверка работы при различных разрешениях экрана
- Оценка удобства интерфейса целевой аудиторией

Этапы и сроки разработки

1. Проектирование архитектуры приложения – 1 день
2. Разработка графического интерфейса – 2 дня
3. Реализация логики вычислений – 2 дня
4. Реализация обработки ошибок и исключений – 1 день
5. Тестирование и отладка программы – 2 дня
6. Написание документации – 1 день

Общий срок разработки: 9 рабочих дней

Дополнительная документация на программный продукт "Конвертер величин"

1. Руководство пользователя

Обзор приложения

«Калькулятор участка» — это графическое приложение для выполнения расчетов, связанных с земельными участками. Оно поддерживает следующие функции:

- Расчет площади участка по длине и ширине.
- Перевод площади из квадратных метров в сотки.
- Расчет стоимости участка по площади и стоимости 1 кв.м.
- Определение максимальной площади застройки (30% от площади участка).
- Расчет стоимости ограждения по периметру.

Приложение позволяет вводить данные в поля ввода, выбирать команду из списка и получать результат в удобном виде.

Установка и запуск

Требования: Python 3.x с установленной библиотекой Tkinter (обычно входит в стандартную поставку Python).

Запуск:

- Сохраните код в файл main.py.
- Выполните команду `python main.py` в терминале или двойным щелчком по файлу.
- Откроется окно программы.

Интерфейс пользователя

- Меню выбора команды: список с номерами 1–5 (описание каждой команды отображается под меню).
- Поля ввода: зависят от выбранной команды (например, для команды 1 — «Длина» и «Ширина»).
- Кнопка «Рассчитать»: выполняет расчет и отображает результат.
- Раздел результата: показывает вычисленное значение.

- Окна ошибок: при некорректном вводе (пустое поле, текст вместо числа, отрицательные значения) программа сообщает пользователю об ошибке и предлагает исправить данные.

Примеры использования

Расчет площади участка:

- Выберите команду «1».
- Введите длину = 20, ширину = 15.
- Нажмите «Рассчитать».
- Результат: «Площадь: 300.00 кв.м».

Перевод в сотки:

- Выберите команду «2».
- Введите площадь = 1500.
- Нажмите «Рассчитать».
- Результат: «Участок: 0.15 соток».

Стоимость участка:

- Выберите команду «3».
- Введите площадь = 500, стоимость кв.м = 2000.
- Нажмите «Рассчитать».
- Результат: «Стоимость: 1000000.00 руб».

Максимальная площадь застройки:

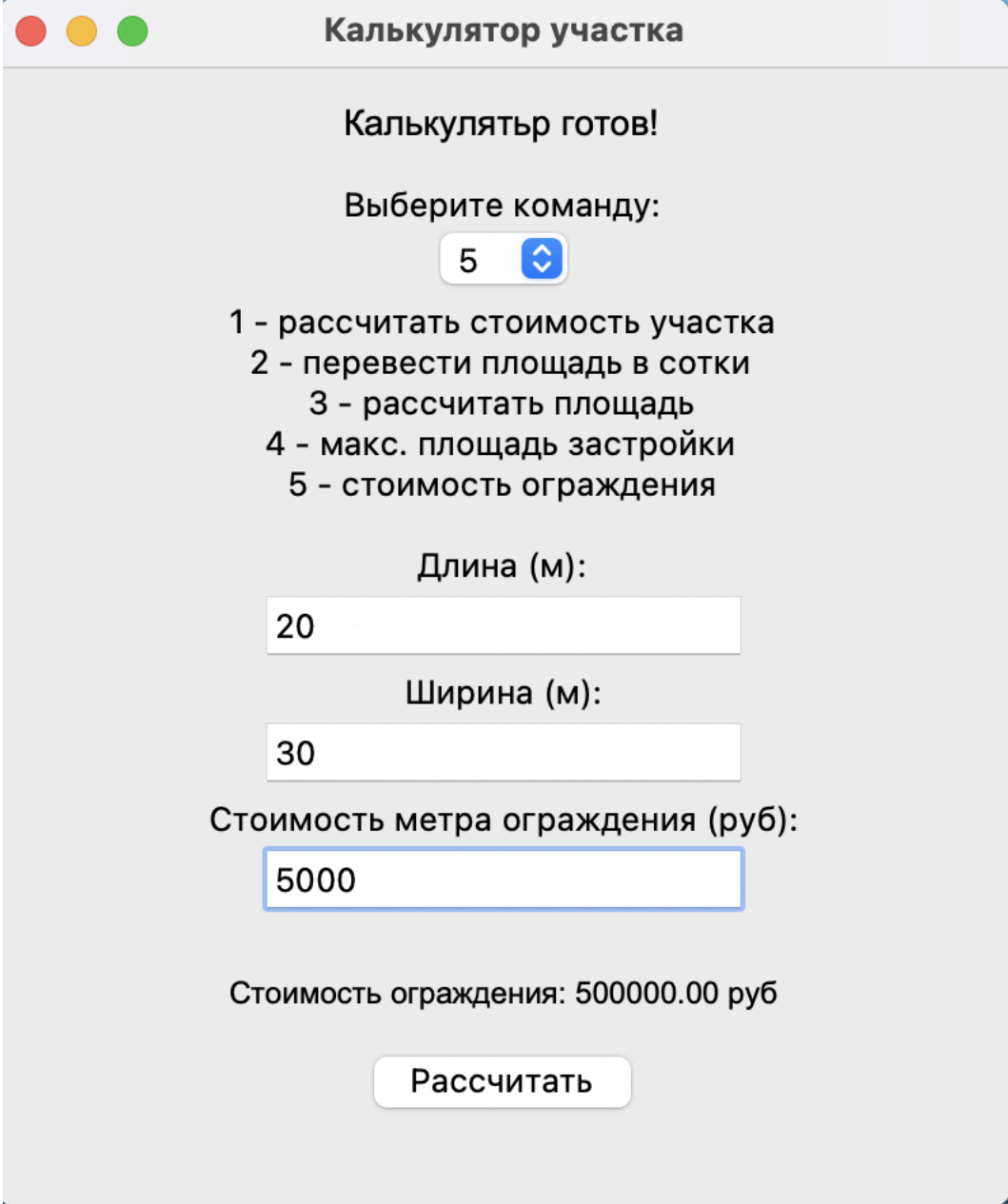
- Выберите команду «4».
- Введите длину = 40, ширину = 30.
- Результат: «Макс. площадь застройки: 360.00 кв.м».

Стоимость ограждения:

- Выберите команду «5».
- Введите длину = 25, ширину = 15, стоимость метра ограждения = 500.
- Результат: «Стоимость ограждения: 20000.00 руб».

Советы

- Вводите только положительные числа.
- При ошибке программа не закроется, а покажет сообщение с подсказкой.
- Для повторного расчета можно просто заменить значения и нажать «Рассчитать» снова.



Калькулятор участка

Калькулятор готов!

Выберите команду:

5

1 - рассчитать стоимость участка
2 - перевести площадь в сотки
3 - рассчитать площадь
4 - макс. площадь застройки
5 - стоимость ограждения

Длина (м):

20

Ширина (м):

30

Стоимость метра ограждения (руб):

5000

Стоимость ограждения: 500000.00 руб

Рассчитать

Рисунок 1 - Интерфейс

Отчёт по тестированию

Выявленные ошибки:

Входные значения	Ожидаемый результат	Полученные значения
Расчет стоимости участка	Стоимость участка	Расчет площади. Рисунок 1
Расчет площади участка	Площадь участка	Стоимость участка. Рисунок 2
Расчет площади участка в сотках	Площадь участка в сотках	Площадь участка в сотках выводится с ошибкой, также не указать цену за квадратный метр участка. Рисунок 3
Перевод площади в сотки	Площадь в сотках	Неправильный расчет соток. Рисунок 4
Максимальная площадь застройки	Максимальная площадь застройки в квадратных метрах	Неправильный расчет площади. Рисунок 5

Приложение

Рисунки с ошибками протестированного приложения

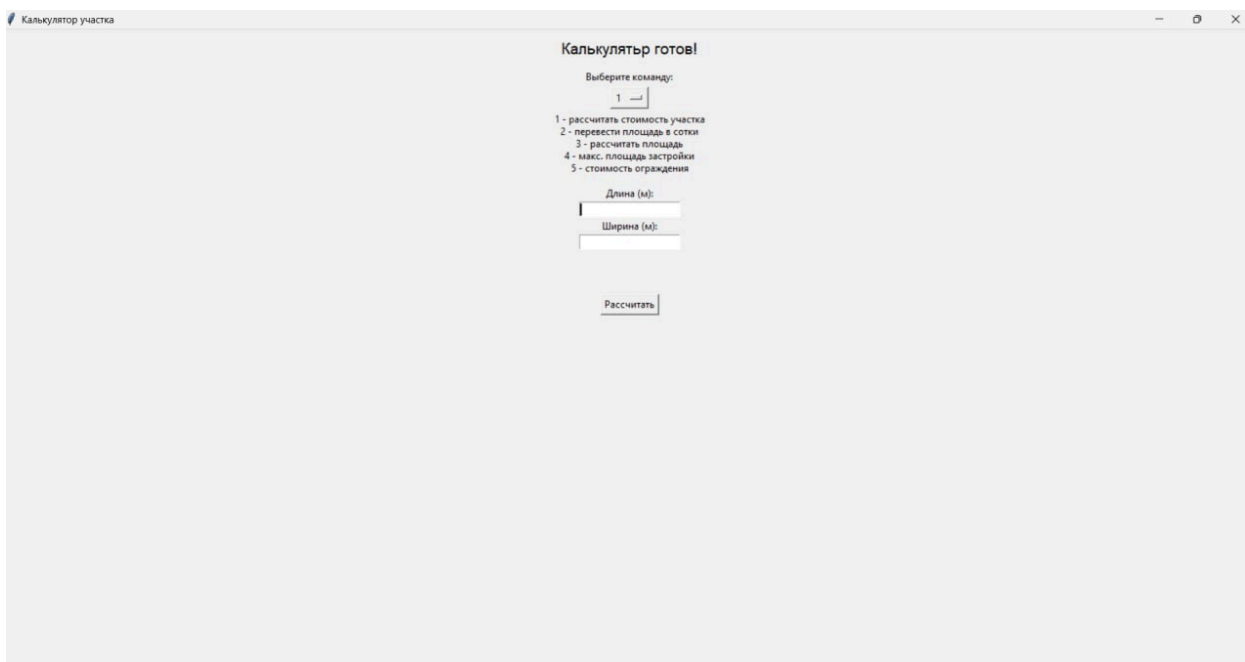


Рисунок 1 - Расчет стоимости участка

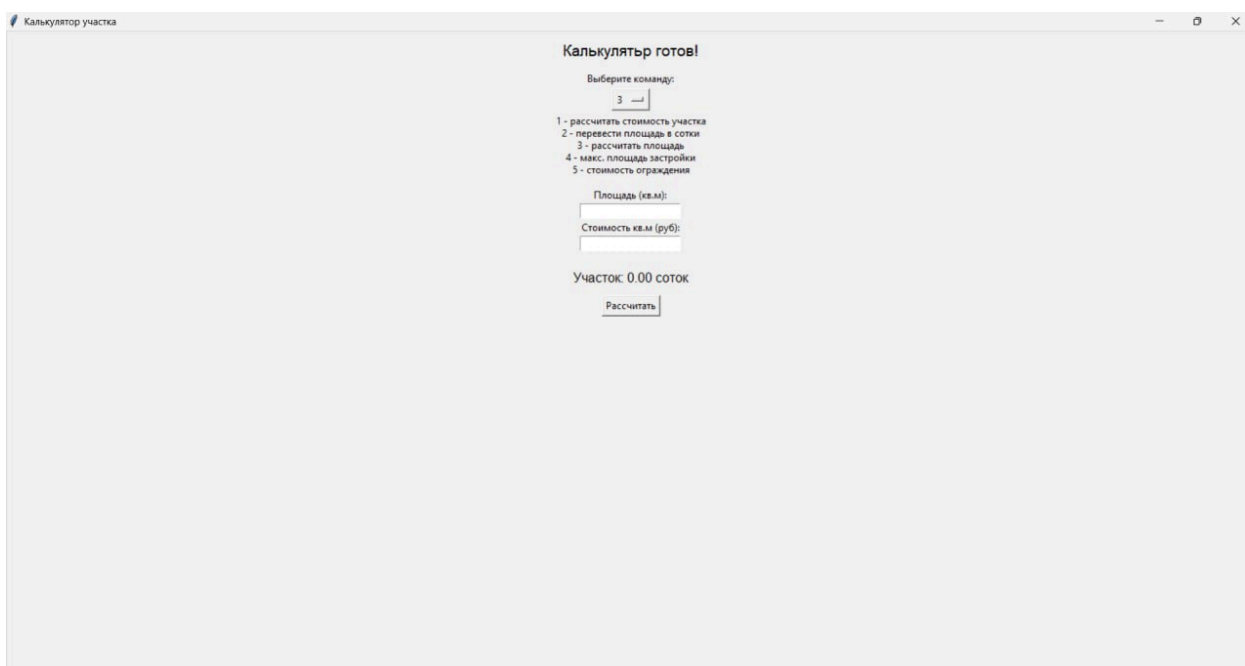


Рисунок 2 - Расчет площади участка

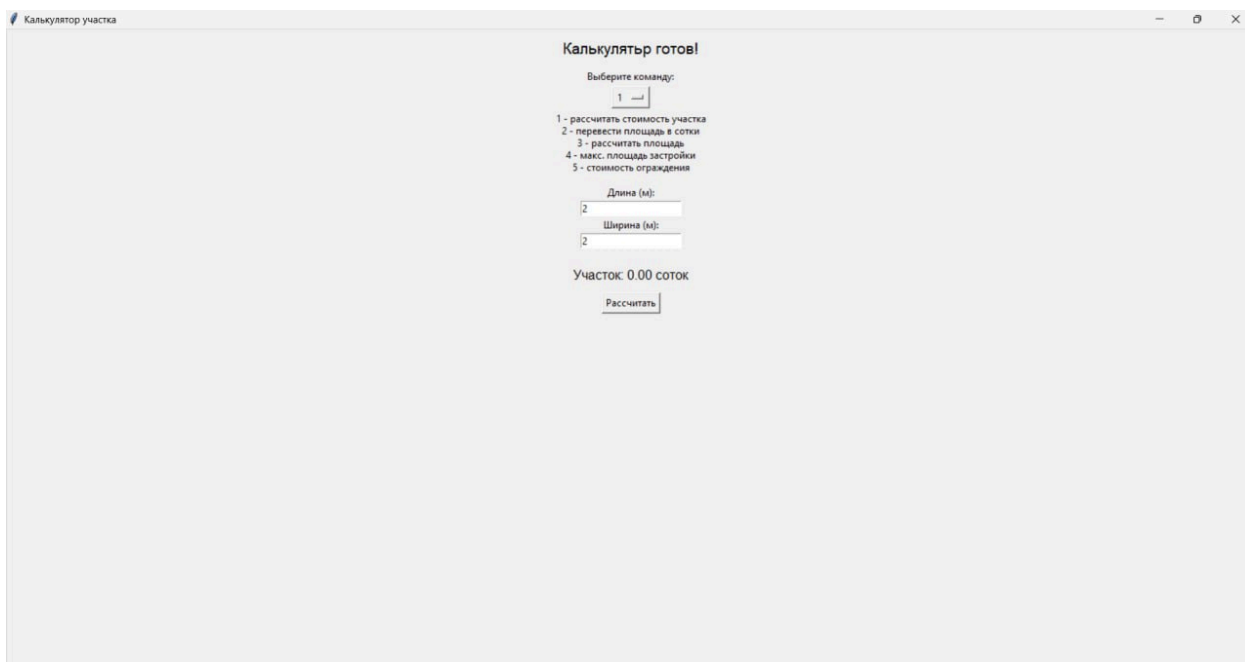


Рисунок 3 - Расчет площади участка в сотках

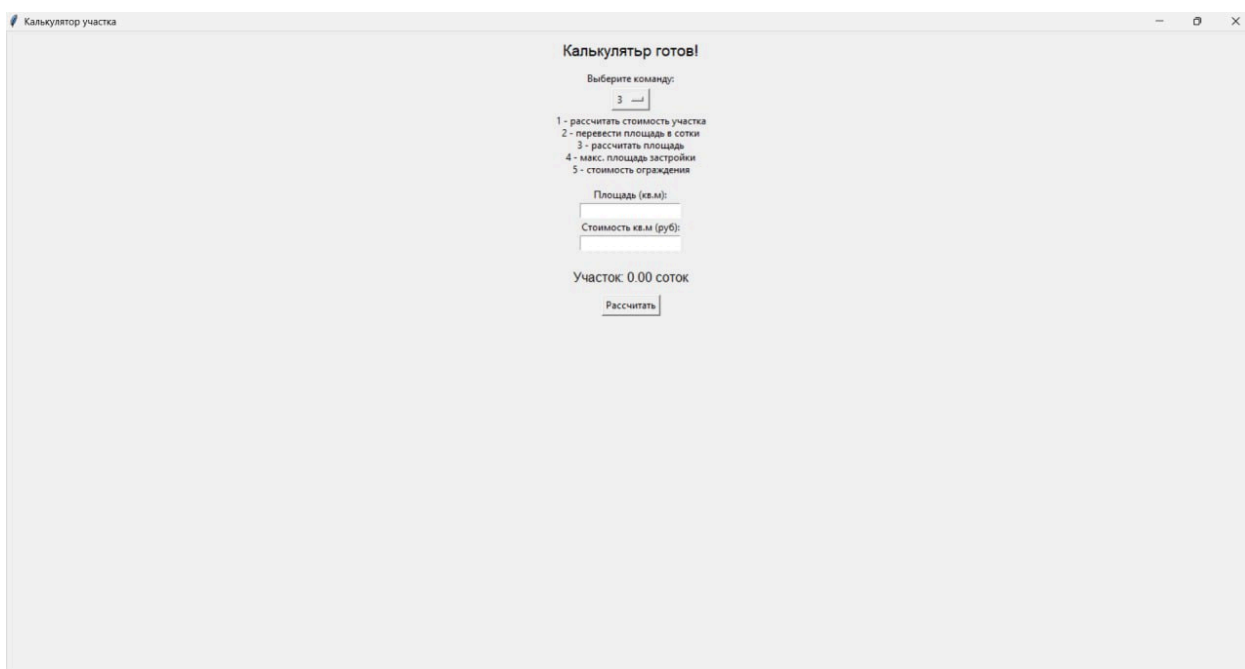


Рисунок 4 - Перевод площади в сотки

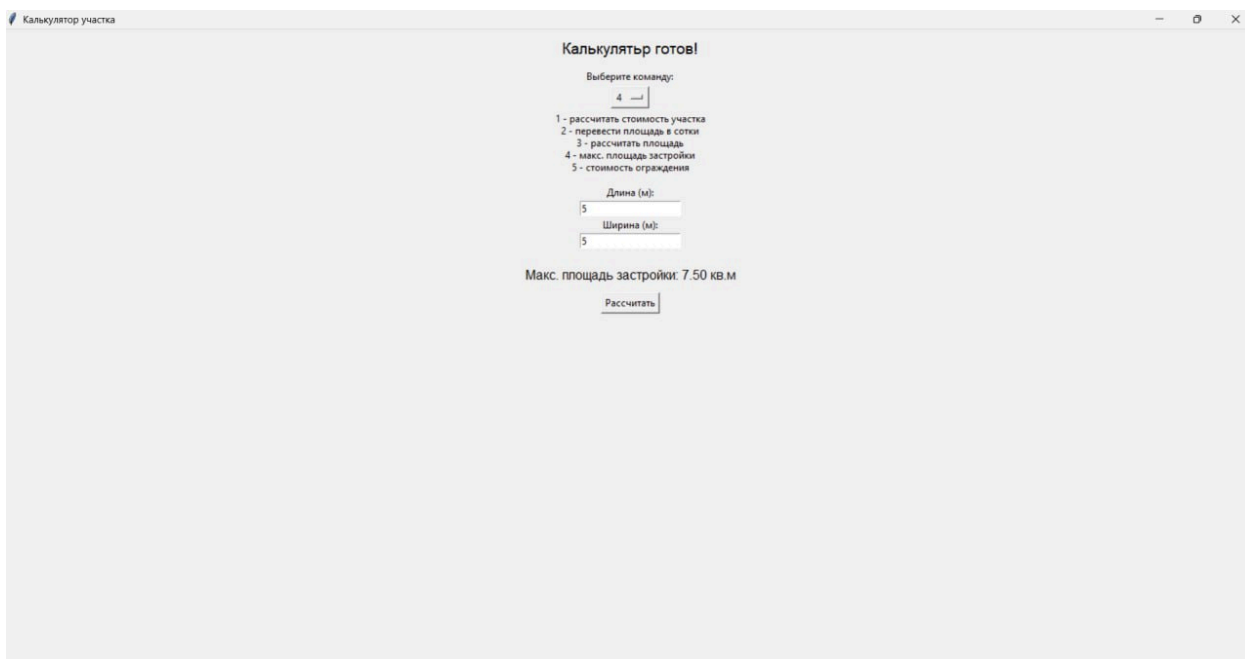


Рисунок 5 - Максимальная площадь застройки

Итог: В результате тестирования были выявлены некоторые ошибки, которые препятствуют корректной работе системы в соответствии с техническим заданием. Проблемы возникают при выполнении ключевых функций, таких как обработка путей, создание и управление папками, загрузка и скачивание файлов, а также операции с облачным хранилищем. Эти ошибки требуют дальнейшей проработки и устранения для обеспечения стабильной работы системы

Вывод

В результате тестирования программного продукта были выявлены 5 ошибок. Программный продукт не соответствует некоторым требованиям.

ЗАКЛЮЧЕНИЕ

В ходе выполнения данной практической работы были изучены аспекты структуры технического задания для программного обеспечения. Также были приобретены практические навыки по созданию документации и написанию ТЗ для собственного продукта.