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»

## Практическая работа № 1

Студенты группы *ИКБО-50-23, в составе подгруппы «Team5»,*
*Астахов С.П., Бурыхин И.Е.,*
*Враженко Д.О., Петруничев А.А.*

_____
(подпись)

Преподаватель *Ильичев Г.П.*

_____
(подпись)

Отчет представлен «___»_____2025 г.

Москва 2025 г.

# 1. ТЕХНИЧЕСКОЕ ЗАДАНИЕ

## 1.1. Введение

Настоящее техническое задание (ТЗ) определяет цели, требования и условия разработки настольного приложения "Универсальный конвертер". Программа предназначена для быстрого и удобного пересчета различных физических величин. Продукт предоставляет возможность конвертации валют, единиц длины и массы. Целевая аудитория продукта — широкий круг пользователей, включая путешественников, инженеров, студентов и сотрудников малого бизнеса, нуждающихся в проведении различных расчетов.

## 1.2. Основания для разработки

Разработка программного продукта "Универсальный конвертер" инициирована необходимостью автоматизации процесса конвертации различных физических величин для сотрудников малого бизнеса и образовательных учреждений, работающих с международными клиентами и проектами, с целью исключения ошибок при ручном расчете и экономии времени.

В качестве нормативной базы для разработки используются:

- Настоящее техническое задание.
- Стандарт ГОСТ 34.602-2020 на составление технического задания.
- Общепринятые принципы удобства использования и проектирования пользовательских интерфейсов (UI/UX).

## 1.3. Назначение разработки

Целью разработки является создание простого и надежного инструмента для конвертации различных физических величин, который позволит пользователям мгновенно получать расчеты без необходимости использования онлайн-сервисов или ручных вычислений. Ожидаемый эффект — экономия времени пользователя на 100% по сравнению с ручным расчетом на калькуляторе и снижение количества ошибок при конвертации.

## 1.4. Требования к программе

**Функциональные требования:**

Программа должна предоставлять графический интерфейс пользователя (GUI) с системой вкладок для различных типов конвертации.

Программа должна предоставлять возможность конвертации трех категорий величин:

- Валюты
- Единицы длины
- Единицы массы

Для каждой категории пользователь должен иметь возможность:

- Ввести числовое значение для конвертации
- Выбрать исходную единицу измерения из выпадающего списка
- Выбрать целевую единицу измерения из выпадающего списка
- Инициировать процесс конвертации нажатием на кнопку "Конвертировать"

Программа должна отображать результат конвертации в понятном формате (<Значение> <Исх. единица> = <Результат> <Целевая единица>).

Поддерживаемые валюты: Российский рубль (RUB), Доллар США (USD), Евро (EUR), Фунт стерлингов (GBP), Индонесийская рупия (IDR), Казахстанский тенге (KZT).

Поддерживаемые единицы длины: Метр, Километр, Сантиметр, Миля, Фут, Дюйм, Ярд, Морская миля.

Поддерживаемые единицы массы: Килограмм, Грамм, Фунт, Унция, Тонна, Карат, Стоун.

**Требования к надежности:**

Программа должна обрабатывать ввод только числовых значений в поля ввода. Ввод нечисловых символов должно обрабатываться с выводом соответствующего сообщения об ошибке.

Программа должна быть устойчива к попытке конвертации до того, как пользователь выбрал валюты.

**Условия эксплуатации:**

Операционная система: Windows 10 и новее.

Минимальные аппаратные требования: Любой современный процессор, 512 МБ ОЗУ, 10 МБ свободного места на диске.

**Требования к совместимости:**

Программа является самостоятельным исполняемым (.exe) файлом и не требует установки дополнительного программного обеспечения для работы.

## 1.5. Требования к интерфейсу

Интерфейс должен быть простым и интуитивно понятным.

- Программа должна использовать интерфейс с вкладками для переключения между различными типами конвертации (валюты, длины, массы).
- Поля для ввода значений должны быть достаточно широкими для ввода чисел до 10 знаков.
- Выпадающие списки для выбора единиц измерения ("Из" и "В") должны быть четко подписаны.
- Кнопка "Конвертировать" должна быть размещена на каждой вкладке и иметь понятную текстовую метку.
- Поле для вывода результата должно быть расположено в нижней части каждой вкладки и иметь достаточный размер для отображения результата.
- Окно программы должно открываться по центру экрана и не иметь возможности изменения размеров.

## 1.6. Критерии приёмки

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

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

## 1.7. Требования к документации

В состав поставки программного продукта должна входить следующая документация:

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

## 1.8. 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 вычислений на основе заранее подготовленных тестовых данных с известным ожидаемым результатом.
- Тестирование удобства использования.

## 1.9. Этапы и сроки разработки

| № | Наименование этапа | Срок выполнения |
|---|---|---|
| 1 | Проектирование архитектуры и UI/UX | 2 дня |
| 2 | Разработка кода программы | 2 дня |
| 3 | Сборка исполняемого файла | 1 день |
| 4 | Написание сопроводительной документации | 2 дня |
| 5 | Внутреннее тестирование и отладка | 1 день |
| | **Итого:** | **8 дней** |

# 2. ДОПОЛНИТЕЛЬНАЯ ДОКУМЕНТАЦИЯ

## 2.1. Руководство пользователя

**Назначение программы:**

Программный продукт «Универсальный конвертер» предназначен для быстрого пересчета денежных сумм между различными валютами и перевода между единицами длины и массы. Программа работает автономно и не требует подключения к сети Интернет.

**Системные требования:**

- Операционная система: Windows 10 или новее,
- Дополнительное программное обеспечение: не требуется.

**Установка и запуск:**

- Скачать исполняемый файл converter.exe из репозитория проекта,
- Сохранить файл в любую папку на локальном диске,
- Запустить программу двойным щелчком мыши по файлу converter.exe.

**Использование:**

- В поле «Сумма» ввести числовое значение для конвертации,
- В выпадающем списке «Из» выбрать исходную валюту,
- В выпадающем списке «В» выбрать целевую валюту,
- Нажать кнопку «Конвертировать»,
- Результат конвертации будет отображён в нижней части окна.

**Поддерживаемые валюты:**

- RUB — Российский рубль,

- USD — Доллар США,
- EUR — Евро,
- GBP — Фунт стерлингов,
- IDR — Индонезийская рупия,
- KZT — Казахстанский тенге.

**Поддерживаемые длины:**

- Метр,
- Километр,
- Сантиметр,
- Миля,
- Фут,
- Дюйм,
- Ярд,
- Морская миля.

**Поддерживаемые массы:**

- Килограмм,
- Грамм,
- Фунт,
- Унция,
- Тонна,
- Карат,
- Стоун.

**Особенности:**

- Автономная работа без подключения к Интернету,
- Интуитивно понятный графический интерфейс,
- Не требует установки дополнительных библиотек.

## 2.2. Описание архитектуры системы

Программный продукт реализован как настольное приложение с графическим интерфейсом пользователя.

Архитектура системы условно разделена на следующие модули:

- **Интерфейс пользователя (UI)** — форма ввода данных, элементы управления (поле ввода суммы, выпадающие списки выбора валют, кнопка «Конвертировать»).
- **Модуль обработки данных** — реализация алгоритма пересчета суммы на основе зафиксированных коэффициентов конверсии.
- **Модуль отображения результата** — вывод информации в графическое окно программы.

## 2.3. Схема архитектуры

На рисунке 1 представлена упрощённая схема архитектуры приложения.

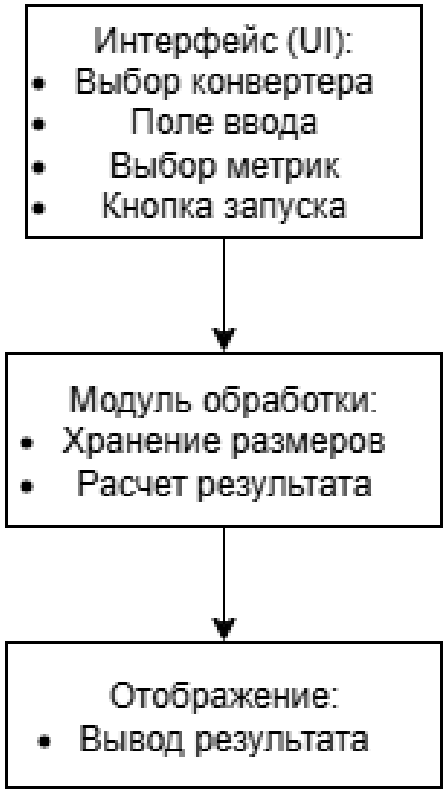


Рисунок 1 - Архитектура программного продукта «Универсальный конвертер»

# 3. ОПИСАНИЕ ВНЕСЕННЫХ ОШИБОК

**1. Жёсткая привязка конвертации к рублю**

- **Тип ошибки:** логическая,
- **Описание:** все операции пересчёта выполняются относительно рубля, что делает невозможным прямую конвертацию между другими валютами,
- **Способ обнаружения:** выявляется посредством возможного несинхронного обновления курсов рубля и остальных валют относительно него.

**2. Необновляемые курсы валют**

- **Тип ошибки:** логическая,
- **Описание:** курсы валют заданы статически в коде и не обновляются из внешнего источника,
- **Способ обнаружения:** выявляется путём сравнения курсов в программе с реальными рыночными курсами на текущую дату.

**3. Некорректная формула для перевода**

- **Тип ошибки:** логическая.
- **Описание:** положения переменных, ответственных за индексы валют, перепутаны местами.
- **Способ обнаружения:** ошибка проявляется при выполнении конвертации валют.

**4. Отсутствие ограничения длины результата конвертации**

- **Тип ошибки:** интерфейсная.
- **Описание:** программа выводит результат конвертации без ограничений на количество отображаемых знаков после запятой, что снижает читаемость и затрудняет восприятие.
- **Способ обнаружения:** выявляется при вводе сумм, которые приводят к длинным дробным результатам.

**5. Отсутствие проверки корректности вводимых данных**

- **Тип ошибки:** логическая/интерфейсная.
- **Описание:** программа допускает конвертацию отрицательных значений, что противоречит логике финансовых операций.
- **Способ обнаружения:** выявляется при вводе отрицательной суммы в поле «Сумма».

# 4. ТЗ, ДОКУМЕНТАЦИЯ СТОРОННЕГО ПО И ЕГО АНАЛИЗ

## 4.1. ТЗ и документация стороннего ПО

Ниже приведено с сохранением авторской структуры, орфографии, пунктуации и проч. ТЗ и документация ПО другой команды:

Техническое задание на разработку таск-трекера.

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.

Общие сведения

Назначение

Программное приложение To-Do List.

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.

**Разработчики:**

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

**Заказчики:**

Преподаватель

**Основание для разработки:**

Договор № 123.45 от 06.09.2024 на разработку автоматизированной системы.

**Назначение и цели создания системы**

**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.

**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 сбоев 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.

## 4.2. Анализ и рекомендации

**Анализ полноты и логичности ТЗ**

Представленное ТЗ является в достаточной мере полным и охватывает все ключевые аспекты, предусмотренные стандартами. В документе последовательно отражены назначение системы, цели её создания и основные функциональные возможности. Включены требования как функционального, так и нефункционального характера, так же включены разделы, посвящённые характеристике объекта автоматизации, порядку разработки и приёмке системы, а также формулировке требований к документации. Логика изложения материала не нарушена: документ следует от общей постановки задачи к

конкретным характеристикам и условиям эксплуатации. Используемые средства реализации адекватны заявленным целям — созданию автономного настольного приложения с устойчивой производительностью и удобным интерфейсом. В работе можно выделить следующие противоречия: в разделе, описывающем условия эксплуатации, наряду с характеристиками локального настольного приложения указаны требования к интернет-соединению, серверному оборудованию и платформе Spring Boot, что не соответствует выбранной технологии реализации и создаёт двусмысленность в интерпретации.

**Замечания**

Несмотря на полноту охвата, ТЗ содержит отдельные методологические и содержательные неточности. В частности, раздел «Введение» акцентирован на вопросах тестирования, тогда как в соответствии с ГОСТ он должен определять актуальность, цели и задачи разработки. В характеристике объекта автоматизации допущено смешение требований к настольному приложению и серверной системе на основе Java Spring Boot, что снижает внутреннюю согласованность документа. Описание функциональных возможностей является подробным, однако не всегда представлено в виде чётких формализованных требований. Требования к интерфейсу обозначены, но не закреплены в строгой форме (например, отсутствуют описания структуры экранных форм). Отдельные нефункциональные характеристики (надёжность, безопасность, доступность) изложены в общем виде и требуют большей детализации (в части обработки ошибок, сохранности данных и защиты информации).

**Рекомендации**

Целесообразно скорректировать раздел «Введение» в соответствии с требованиями ГОСТ, выделив в нём цели и актуальность разработки, а описание методологии тестирования перенести в отдельный раздел документации. Необходимо устранить противоречия в разделе, посвящённом условиям эксплуатации, исключив упоминание требований к интернет-соединению, серверному оборудованию и Java Spring Boot, так как они не применимы к настольному приложению на Python. Следует уточнить

функциональные и нефункциональные требования, оформив их в виде чётких формализованных формулировок, исключающих двусмысленность. Рекомендуется расширить раздел «Требования к интерфейсу», включив описание экранных форм и элементов управления. Требования к надёжности и безопасности следует конкретизировать: определить механизмы обработки ошибок, сценарии восстановления данных при сбое и меры по защите локально хранимых файлов. Также обоснованно дополнить ТЗ разделом о расширяемости системы, учитывая указанные перспективы развития (поддержка подзадач, напоминаний, интеграции с календарём).

# 5. Результаты тестирования программного продукта другой команды

В ходе тестирования «черным ящиком» проекта другой команды выявлено ряд различных ошибок.

Критические:

- Не работает изменение приоритета в окне редактирования
- Нет проверки содержания в поле даты
- Выбор приоритета полностью не работает
- Кнопка "+Добавить" не работает
- При резервном копировании в .csv у задач пропадает дата

Средние:

- Горячие клавиши просто не работают при включенном капсе и при смене языка
- Все колонки на главном экране перепутаны местами
- Задачу после "выполнения" нельзя заново сделать активной

Незначительные:

- Неправильное название в приложении. Окно на создание задачи называется "редактировать задачу" и наоборот (окно редактирования называется "новая задача")
- При изменении размера интерфейса до минимальной верхней части интерфейса программы уменьшается настолько, что перестаёт правильно отображаться

- Номер задачи не сбрасывается при удалении задачи
- При удалении задач отображаемая статистика справа снизу перестаёт правильно считать задачи

Сначала мы начали тестировать программу на корректность слов, опечаток и других орфографических ошибок и сразу же наткнулись на неправильное название в создании задачи. Далее решили изменить размер интерфейса до удобного и заметили, приложение не удобно показывает во свернутом режиме, вернули в обратный режим и решили добавить задачу на другом языке. Отсюда и узнали, что горячие клавиши не работают на другом языке, кроме основного, которого написали авторы. При ряде действий узнали, что ожидаемый результат в виде некоторых кнопок и даты просто не работают или работают неправильно, а еще увидели, что колонки все перепутаны.

# 6. Анализ документации другой команды

Техническое задание обладает четкой структурой и охватывает ключевые аспекты разработки приложения для управления задачами. Однако, в документе присутствуют отдельные положения, допускающие неоднозначную трактовку, что может привести к разночтениям в процессе реализации проекта.

Основное содержание задания сфокусировано на описании конечных целей и функциональности продукта, в то время как детализированные аспекты технической реализации некоторых требований требуют дополнительной конкретизации.

**1. Неактуальная информация / Внутренние противоречия**

**Пункт 4. Архитектура:**

> `"Файл проекта — единый исполняемый скрипт todo_app.py."`
>
> **Анализ:** это требование противоречит заявленной архитектуре со слоями (`TaskRepo`, `ToDoApp`, `TaskDialog`). Размещение всего кода в одном файле плохая практика для любого проекта, кроме самого простого. Это приведёт к спагетти-коду, сложностям в разработке и поддержке. Скорее всего, это устаревшее или ошибочное пожелание.

**2. Двусмысленные и размытые формулировки**

**Пункт 2.1. Управление задачами:**

> `"Редактирование существующих задач."`
>
> **Анализ:** формулировка слишком общая. Можно ли редактировать все поля? Что происходит при редактировании? Сбрасывается ли дата создания? Меняется ли дата редактирования? Это важно для логики приложения.

**Пункт 2.1. Управление задачами:**

> `"Массовые операции: ... Изменить статус нескольких задач одновременно."`
>
> **Анализ:** как пользователь должен выделять несколько задач? Через Ctrl+Click, Shift+Click? Через флажки? Интерфейсный механизм не описан.

**Пункт 2.2. Отображение и навигация:**

> `"Цветовая подсветка: ... Задачи «скоро» (до 3 дней) — зелёный текст"`

**Анализ:** "Скоро" — от какого дня отсчитывается? От текущей даты? Например, задача со сроком `завтра` — это "скоро"? А задача со сроком `послезавтра`? А если срок пустой?

**Пункт 2.2. Отображение и навигация:**

`"Фильтрация по: ... Категории (Все / конкретная категория)"`

**Анализ:** откуда берутся "конкретные категории"? Пользователь вводит их вручную в поле при создании задачи? Есть ли выпадающий список с существующими категориями? Можно ли создать новую категорию на лету?

**Пункт 3. Нефункциональные требования:**

`"Поддержка контекстного меню (ПКМ)."`

**Анализ:** крайне размытая формулировка. Что должно быть в этом меню? Все действия (редактировать, удалить, отметить, как выполненное)? Только часть? Контекстное меню должно появляться только когда выделена задача?

**3. Отсутствие важных деталей**

**Пункт 2.1. Управление задачами / Пункт 2.3. Дополнительные функции:**

**Анализ:** полностью отсутствует описание логики импорта из CSV. Что происходит при импорте?

Добавляются новые задачи или обновляются существующие?
Что если в импортируемом файле есть задача с таким же ID?
Что если формат файла не совпадает с экспортируемым?
Нужна ли валидация данных при импорте?

**Пункт 2.2. Отображение и навигация:**

**Анализ:** не указано, как должен выглядеть интерфейс фильтров и поиска. Это отдельное окно? Панель инструментов? Где она расположена?

**Пункт 2.3. Дополнительные функции:**

`"Подсчёт статистики (отображается внизу): ... Просроченные"`

**Анализ:** как считается "Просроченные"? Это задачи с статусом "Активна" и сроком выполнения меньше текущей даты? Или сюда включаются и выполненные просроченные задачи?

**Пункт 4. Архитектура:**

**Анализ:** описание архитектуры очень высокоуровневое. Нет описания полей и методов классов. Что хранится в `TaskRepo`? (`get_all_tasks`, `add_task`, `update_task`, `delete_task`...). Какие поля есть у задачи? (должен быть `id`, `created_date`).

# 7. Заключение, оценка качества программного продукта и документации, выводы и рекомендации

На основании проведённого тестирования и анализа документации можно сделать вывод о низком качестве программного продукта.

**Оценка качества программного продукта:** Неудовлетворительная.

Приложение содержит критические ошибки, делающие ключевой функционал (добавление, редактирование задач, работу с приоритетами) неработоспособным. Наличие средних и незначительных ошибок указывает на поверхностное тестирование со стороны разработчиков и отсутствие процесса контроля качества. Продукт не готов к использованию.

**Оценка качества документации (ТЗ):** Удовлетворительная, с существенными недостатками.

Техническое задание хорошо структурировано и покрывает основные требования к функционалу. Однако наличие размытых формулировок и отсутствие важных деталей (критериев приемки, описания алгоритмов, спецификации модели данных) напрямую способствует к ошибкам в реализации. Документация задала направление, но не обеспечила однозначности трактовки требований.

**Выводы и рекомендации:**

1. Вывод: Большинство критических ошибок в программе являются следствием не только ошибок кодирования, но и фундаментальных пробелов в проектировании и спецификации требований.

   Рекомендация: немедленно исправить критические ошибки. Перед этим — дополнить ТЗ недостающими спецификациями (логика импорта/экспорта, валидация данных, механизм выделения, математические формулы для условий).

2. Вывод: Команда не провела полноценное собственное тестирование, включая проверку на разных раскладках клавиатуры и разрешениях экрана.

   Рекомендация: внедрить регрессионное и кроссплатформенное тестирование. Составить чек-листы и тест-кейсы, покрывающие все сценарии использования, указанные в ТЗ.

3. Вывод: Архитектурное решение о размещении кода в одном файле негативно сказалось на качестве кода и сопровождаемости.

   Рекомендация: провести рефакторинг кода, разделив его на логические модули (классы работы с БД, GUI, бизнес-логики) в соответствии с заявленной в ТЗ архитектурой.

4. Вывод: ТЗ не содержало четких критериев приемки, что позволило команде сдать нерабочий продукт.

   Рекомендация: на будущее для каждого проекта детализировать ТЗ, добавляя для каждого функционального требования конкретные критерии приемки, которые будут использоваться для приемочного тестирования.