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 №1

по дисциплине «Тестирование и верификация программного обеспечения»

Тема: Тестирование программного продукта методом «черного ящика»

***Группа*:** *ИКБО-66-23*

Состав команды: Смирнов А. Ю.
Ковалев А.Э.
Гордильо Ариса

Дата выполнения: 22 сентября 2025 г.

Москва 2025

Пункт 1. Разработка ТЗ и ПП

1. ВВЕДЕНИЕ

Техническое задание (ТЗ) определяет цели, требования и условия разработки настольного приложения «Список покупок». Программа предназначена для удобного создания, хранения и управления списками необходимых покупок. Продукт предоставляет возможность добавления и редактирования товаров, указания количества и категорий, а также отметки приобретённых позиций. Целевая аудитория продукта — широкий круг пользователей, включая домохозяек, студентов, офисных работников и представителей малого бизнеса, которым важно быстро и эффективно организовывать процесс покупок и контролировать расходы.

2. ОСНОВАНИЯ ДЛЯ РАЗРАБОТКИ

Разработка программного продукта **«Список покупок»** инициирована необходимостью автоматизации процесса планирования и ведения покупок для широкого круга пользователей. Использование приложения позволит исключить ошибки, возникающие при ручном составлении бумажных списков, уменьшить вероятность забытых товаров, а также сэкономить время и средства за счёт удобного контроля над покупками. 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).

3. НАЗНАЧЕНИЕ РАЗРАБОТКИ

Целью разработки является создание простого и надёжного инструмента для составления и ведения списков покупок, который позволит пользователям

быстро формировать перечень необходимых товаров, отмечать приобретённые позиции и организовывать покупки без использования бумажных носителей или сторонних заметок.

Ожидаемый эффект — экономия времени пользователя за счёт автоматизации процесса планирования покупок, снижение вероятности забыть нужные товары, а также повышение удобства и эффективности при совершении покупок.

4. ТРЕБОВАНИЯ К ПРОГРАММЕ

4.1. Функциональные требования

Программа должна предоставлять графический интерфейс пользователя (**GUI**) с удобной навигацией для работы со списками покупок.

Программа должна предоставлять возможность работы с тремя основными категориями данных:

- Товары (наименование покупок)
- Количество (единицы измерения и число)
- Категории товаров (например: продукты, бытовая химия, одежда и пр.)

Для каждой позиции пользователь должен иметь возможность:

- Ввести название товара
- Указать количество и единицу измерения
- Выбрать категорию товара из выпадающего списка
- Удалить или отредактировать позицию

Программа должна отображать список покупок в удобном формате: <Наименование товара> — <Количество> <Единица измерения> [Категория].

Поддерживаемые единицы количества: штука (шт.), килограмм (кг), грамм (г), литр (л), миллилитр (мл), упаковка (уп.), коробка (кор.).

4.2. Требования к надежности

** Программа должна обрабатывать корректность ввода данных в поля добавления товара. Ввод недопустимых символов (например, в поле количества — букв или специальных символов) должен обрабатываться с выводом соответствующего сообщения об ошибке.**

** Программа должна быть устойчива к попыткам добавления товара без указания обязательных параметров (например, без наименования или количества).**

** Все внесённые данные должны сохраняться корректно и не теряться при закрытии или перезапуске приложения (если реализуется механизм сохранения).**

4.3. Условия эксплуатации

Операционная система: Android 11 и новее.

4.4. Требования к совместимости

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

5. ТРЕБОВАНИЯ К ИНТЕРФЕЙСУ

Интерфейс должен быть простым и интуитивно понятным.

- Программа должна использовать вкладки или панели для переключения между различными категориями товаров.
- Поля для ввода наименования товара и количества должны быть достаточно широкими для удобного ввода текста и чисел.
- Выпадающий список выбора единицы измерения (шт., кг, л и др.) и категории товара должен быть чётко подписан.
- Кнопка **«Добавить»** должна быть расположена рядом с полями ввода и иметь понятную текстовую метку.

- Список покупок должен отображаться в центральной части окна в табличной или карточной форме с возможностью отметки купленных товаров (например, галочкой).
- Окно программы должно открываться по центру экрана и иметь фиксированный размер, обеспечивающий читаемость списка.

6. КРИТЕРИИ ПРИЕМКИ

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.

7. ТРЕБОВАНИЯ К ДОКУМЕНТАЦИИ

В состав поставки программного продукта должна входить следующая документация:

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

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.

9. ЭТАПЫ И СРОКИ РАЗРАБОТКИ

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

Пункт 2. Эксплуатационная документация

2.1 Инструкция по запуску

- Установите приложение «Список покупок» на устройство с операционной системой Android 11 или выше.
- После установки запустите приложение, нажав на иконку «Список покупок» в меню приложений.
- При первом запуске создайте учётную запись или выполните вход в существующую.

2.2 Инструкции по управлению

- Добавление продукта: Введите название, количество и вес товара в соответствующие поля и выберите категорию. Подтвердите добавление кнопкой «Добавить».
- Редактирование товара: Нажмите на элемент списка и измените его параметры (название, количество, вес, категорию).
- Удаление товара: Нажмите кнопку «Удалить» или выполните свайп по карточке товара. Последняя удалённая позиция сохраняется в базе данных и может быть восстановлена

Пункт 3. Описание внесения багов в своё ПО

Описание бага №1: (инвертированная проверка логина).

Кратко: В модуле авторизации реализована некорректная логика проверки данных. Условие проверки инвертировано: при правильном вводе логина и пароля система выдает ошибку, а при неверных данных допускает вход в приложение.

Пример:

Ввод	Ожидаемый результат	Фактический результат
Log:Jose Password: hochuautomat	Регистрация прошла успешно	Неправильные данные

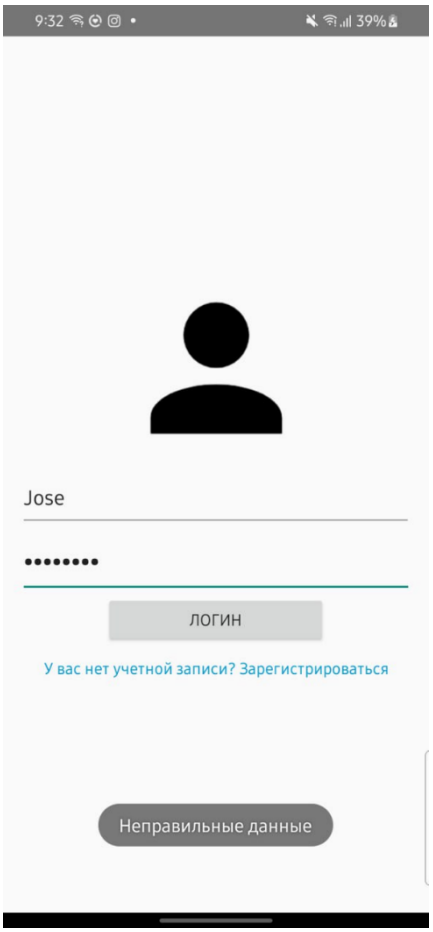

Рисунок 1 – Демонстрация ошибки ввода правильных данных

Описание бага №2: Несоответствие ключей адаптера и данных.

Кратко: Эффект: SimpleAdapter продолжает искать «name», из-за чего имена списков не отображаются и могут быть пустыми в UI/диалогах.

Пример:

Ввод	Ожидаемый результат	Фактический результат
Создать список с именем Продукты на неделю	На главном экране появляется новый список с именем: Продукты на неделю	На главном экране отображается пустая строка вместо названия списка

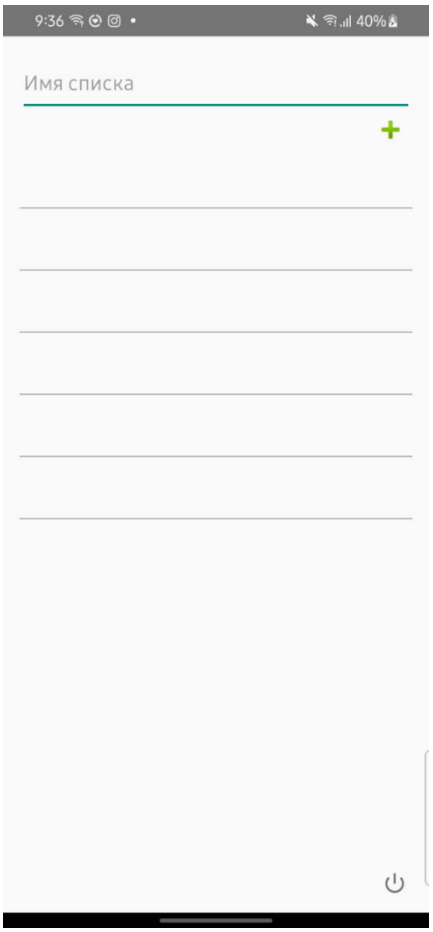

Рисунок 2 – Демонстрация ошибки с добавлением продуктов

Описание бага №3: кнопка Undo больше не восстанавливается.

Кратко: Кнопка **Undo** после удаления товара больше не выполняет восстановление элемента — удалённые позиции не возвращаются в список, что нарушает ожидаемое поведение функции отмены.

Пример:

Ввод	Ожидаемый результат	Фактический результат
Удалить товар Молоко и нажать кнопку **Undo**.	Товар Молоко возвращается в список.	Товар не восстанавливается, список остаётся без Молоко.

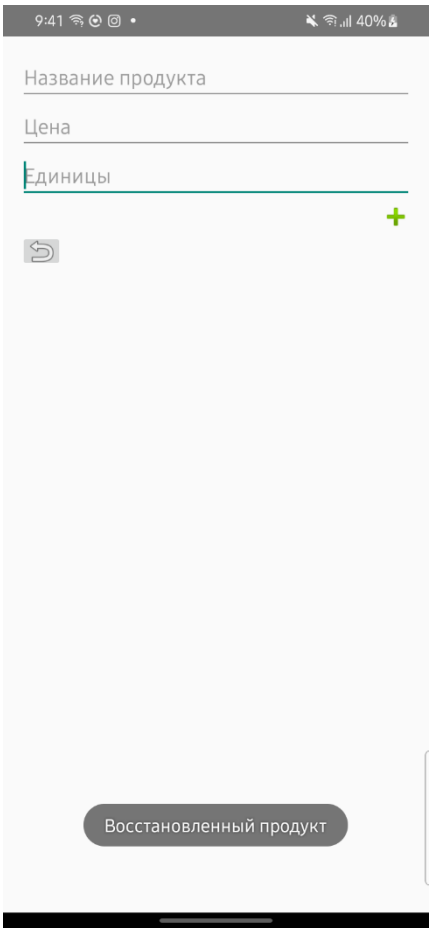

Рисунок 3 – Демонстрация ошибки произведения

Описание бага №4: Кнопка “Войти” недоступна.

Кратко: Кнопка Войти недоступна , что ломает сценарий входа.

Пример:

Ввод	Ожидаемый результат	Фактический результат
На экране входа ввести корректные логин и пароль.	Кнопка **«Войти»** становится активной, и пользователь может авторизоваться.	Кнопка **«Войти»** остаётся недоступной, вход невозможен.

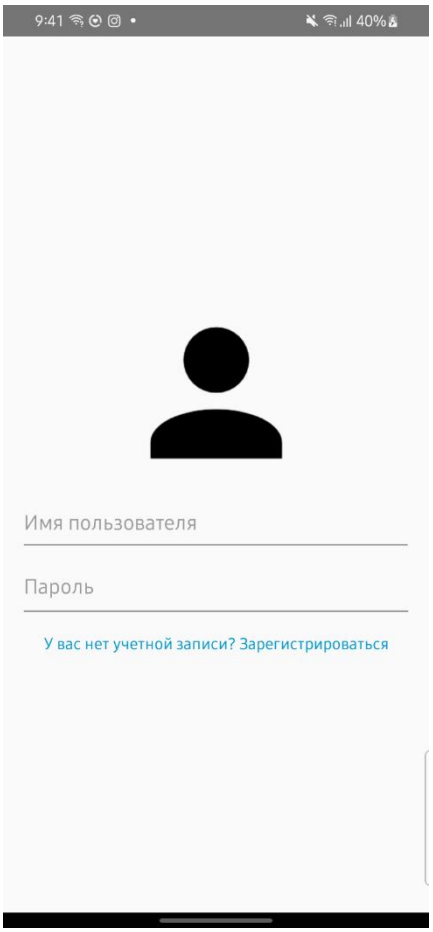

Рисунок 4 – Демонстрация ошибки кнопки "Войти"

Описание бага №5: Некорректный парсинг цен с десятичной точкой.

Кратко: :при наличии цены с десятичной точкой формат парсинга сломается (NumberFormatException). Даже при парсинге удачных значений ID становится нестабильным (ломает стабильность Adapter IDs).

Пример:

Ввод	Ожидаемый результат	Фактический результат
Добавить товар с ценой 2.50	Товар успешно добавляется в список, цена отображается корректно, Adapter сохраняет стабильные ID элементов	Отображение списка нарушается, возможны сбои при обновлении или удалении товаров

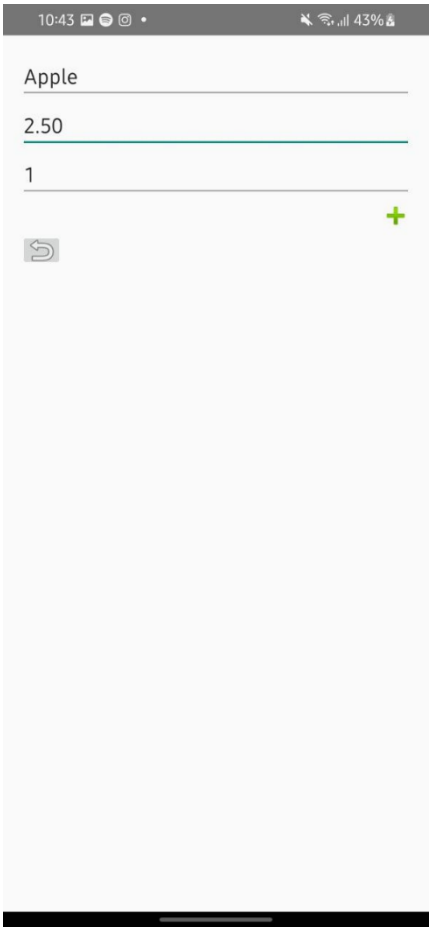

Рисунок 5 – Демонстрация ошибки возведения в степень

Пункт 4. ТЗ и ПП другой команды

Настоящее техническое задание определяет цели, требования и порядок разработки консольного приложения «Конвертер валют» (далее — Программа). Программа реализуется на языке Kotlin и предназначена для выполнения операций по переводу денежных сумм между различными валютами.

Приложение будет использоваться в учебных целях для демонстрации принципов разработки и последующего тестирования программного обеспечения методом «чёрного ящика».

---

1. Основания для разработки

Разработка проводится в рамках практической работы по дисциплине «Тестирование и верификация программного обеспечения».

Основания для создания продукта:

- необходимость предоставления исполняемого приложения для тестирования другой командой;
- закрепление навыков составления технической документации в соответствии с ГОСТ;
- отработка методологии «чёрного ящика».

---

2. Назначение разработки

Программа предназначена для пользователей, которым требуется быстрый инструмент для перевода денежных сумм между несколькими валютами (рубль, доллар США, евро, юань).

Основные задачи разработки:

- реализовать простой и удобный интерфейс;
- предоставить возможность конвертации валют в обоих направлениях;
- отработать процесс тестирования ПП с внесёнными ошибками.

---

3. Требования к программе

3.1 Функциональные требования

Программа должна обеспечивать:

1. Ввод исходной суммы в выбранной валюте.
2. Выбор валюты для перевода.
3. Отображение результата пересчёта в целевой валюте.
4. Возможность повторного ввода без перезапуска программы.
5. Поддержку не менее четырёх валют: RUB, USD, EUR, CNY.

3.2 Требования к надёжности

- При некорректном вводе программа должна выводить сообщение об ошибке без аварийного завершения.
- Результаты вычислений должны быть воспроизводимыми при одинаковом вводе.

3.3 Требования к условиям эксплуатации

- Операционная система: Windows, Linux, macOS.
- Необходим установленный Kotlin (версии 1.8 и выше) или JVM (версии 11 и выше).

3.4 Требования к совместимости

Программа должна запускаться в консоли и не требовать сторонних библиотек.

3.5 Требования к интерфейсу

- Текстовое меню выбора исходной и целевой валюты.
- Поле для ввода суммы.
- Отображение результата в формате: 100 RUB = 1.05 USD

---

4. Критерии приёмки

Программа считается принятой, если:

1. Реализованы все функции из раздела 4.1.
2. Программа корректно выполняет пересчёт валют и выдаёт результат без критических ошибок.
3. Интерфейс соответствует заявленным требованиям.
4. Программа устойчива к некорректному вводу данных.

---

5. Порядок контроля и приёмки

Контроль качества осуществляется методом функционального тестирования («чёрного ящика»).

- **Этап 1:** предоставление исполняемого файла и инструкции по запуску.
- **Этап 2:** проведение тестирования: ввод различных сумм, проверка корректности пересчёта, реакция на ошибки ввода.
- **Этап 3:** фиксация найденных дефектов в отчёте.

---

6. Этапы и сроки разработки

№	Этап разработки	Срок выполнения	Примечание
1	Проектирование структуры ПП	1 день	Определение функционала
2	Реализация базовой логики	2 дня	Ввод/вывод, пересчёт валют
3	Добавление пользовательского меню	1 день	Навигация и интерфейс
4	Тестирование и отладка	1 день	Проверка корректности работы
5	Внесение преднамеренных ошибок	1 день	Для целей тестирования
6	Подготовка документации	1 день	Инструкция и описание
	Итого:	**7 дней**	—

Документация на программный продукт «Конвертер валют»

1. Инструкция по запуску

1. Установите Kotlin версии 1.8+ или Java (JVM 11+).
2. Скачайте исходный файл CurrencyConverter.kt.
3. Скомпилируйте программу:
4. kotlinc CurrencyConverter.kt -include-runtime -d CurrencyConverter.jar
5. Запустите программу:

6. java -jar CurrencyConverter.jar

---

2. Инструкция по использованию

После запуска программы в консоли отобразится меню:

1. Введите сумму, которую хотите конвертировать.
2. Выберите исходную валюту (RUB, USD, EUR, CNY).
3. Выберите целевую валюту.
4. Программа выведет результат конвертации в формате:
5. 100 RUB = 1.05 USD
6. Для повторного ввода просто введите новые данные.

Пункт 5. Выявление ошибок другой команды

Описание замечаний и ошибок, найденных в ходе изучения ТЗ и дополнительной документации другой команды

В ходе изучения технического задания, дополнительной документации и тестирования приложения-конвертера валют были выявлены следующие замечания и ошибки:

1. **Ошибка округления**
    - **Описание:** Результат конвертации всегда округляется до целого числа, что приводит к потере точности.
    - **Как обнаружить:** Конвертировать 1 USD в RUB и обратно. Результат не совпадёт с исходной суммой.
    - **Влияние:** Потеря точности при пересчётах, особенно при повторных операциях.
2. **Ошибка обработки ввода**
    - **Описание:** Если пользователь вводит не число (например, «abc»), программа завершается с ошибкой вместо отображения сообщения об ошибке.
    - **Как обнаружить:** Ввести «abc» вместо суммы.
    - **Влияние:** Программа становится нестабильной, пользователь не может продолжить работу.
3. **Неверный курс валюты**
    - **Описание:** Курс EUR → RUB указан неправильно (например, 1 EUR = 50 RUB вместо актуального ~100).
    - **Как обнаружить:** Сравнить результаты пересчёта с актуальными курсами валют.
    - **Влияние:** Некорректные результаты конвертации, недоверие пользователей.
4. **Ошибка направления пересчёта**
    - **Описание:** Конвертация RUB → USD работает корректно, а обратное преобразование (USD → RUB) выдаёт неверный результат.
    - **Как обнаружить:** Перевести 100 RUB в USD, затем обратно. Итоговая сумма ≠ 100 RUB.
    - **Влияние:** Ошибка расчетов при обратной конвертации, снижение точности приложения.

5. **Отсутствие сохранения выбора валют**
    - **Описание:** Программа каждый раз сбрасывает валюты по умолчанию (RUB → USD), даже если пользователь выбирал другие.
    - **Как обнаружить:** Выполнить несколько переводов подряд с разными валютами.
    - **Влияние:** Ухудшение удобства использования приложения.
6. **Ошибка интерфейса**
    - **Описание:** Результат выводится с избыточным количеством знаков после запятой (например, 1.0000000001 USD).
    - **Как обнаружить:** Перевести дробную сумму (например, 123.45 RUB в USD).
    - **Влияние:** Непривлекательный и трудночитаемый интерфейс, возможное недоверие пользователя.
7. **Ошибка завершения работы**
    - **Описание:** В программе отсутствует команда выхода; для остановки требуется принудительное завершение (Ctrl+C).
    - **Как обнаружить:** Попробовать завершить работу через меню.
    - **Влияние:** Неподдерживаемый пользовательский сценарий, неудобство при завершении работы.

Заключение

Изучение технического задания и сопроводительной документации позволило выявить ряд замечаний и потенциальных проблем, связанных с неполным описанием функционала, недостаточной проработкой механизмов защиты данных и возможными сценариями возникновения ошибок. Для минимизации рисков на этапе разработки и эксплуатации необходимо доработать ТЗ, детализировав функциональные требования, тестовые сценарии и меры по обеспечению безопасности приложения.