



МИНОБРНАУКИ РОССИИ

*Федеральное государственное бюджетное образовательное учреждение
высшего образования*

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт Информационных технологий (ИТ)

Кафедра Математического обеспечения и стандартизации информационных
технологий (МОСИТ)

ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ № 1

по дисциплине

«Тестирование и верификация программного обеспечения»

**Тема: «ТЕСТИРОВАНИЕ ПРОГРАММНОГО ПРОДУКТА
МЕТОДОМ «ЧЕРНОГО ЯЩИКА»**

Выполнили студенты группы ИКБО-43-23

Соловьева Мария, Шпагина
Юлиана, Карасев Егор.

Принял

Практическая работа выполнена

«__»_____2025 г.

(подпись студента)

«Зачтено»

«__»_____2025 г.

(подпись руководителя)

Москва 2025

Название команды «VHLOP»

1 Разработка технического задания и программного продукта

1 Введение

Программный продукт «MiniShop» представляет собой консольное приложение для оформления простых заказов.

Назначение — обучение тестированию методом «черного ящика»: проверка поведения системы на основе входов/выходов.

2 Основания для разработки

Разработка выполняется в рамках учебной ****Практической работы №1**** по тестированию ПО методом «черного ящика».

Нормативные ссылки: ГОСТ 19.201-78, ГОСТ 34.602-2020, ISO/IEC/IEEE 29148:2018.

1 Назначение разработки

Система позволяет создать заказ из нескольких позиций (название, цена, количество, промокод) и получить итог к оплате, сохранив сведения в CSV.

Учебная цель — предоставить тестовый стенд с намеренно посеянными дефектами.

2 Требования к программе

2.1 Функциональные требования

- Ввод позиций заказа (название, цена, количество, промокод).
- Расчет итога с учетом НДС и скидочного кода.
- Просмотр текущего «чека» и суммы.
- Сохранение заказа в `orders.csv`.
- Консольное меню управления.

2.2 Требования к надёжности

- Приложение должно сохранять заказ на диск без зависимостей от СУБД.
- При ошибочном вводе должен выводиться текст ошибки и запрашиваться повторный ввод (фактическое поведение может отличаться — объект тестирования).

2.3 Требования эксплуатации

- ОС: Windows/Linux/macOS; интерпретатор Python 3.8+.
- Консоль/терминал, раскладка RU/EN.

2.4 Требования к совместимости

- Не требует внешних библиотек.
- Файл `orders.csv` совместим с табличными редакторами (Excel/LibreOffice).

3 Требования к интерфейсу

- Простой текстовый интерфейс меню (1–4).
- Все суммы отображаются в рублях с указанием валюты.

1 Критерии приёмки

- Не менее 95% тест-кейсов должны завершаться ожидаемым результатом для исправленной версии приложения.
- Сохранение должно формировать корректные строки, пригодные для импорта в табличные редакторы.
- Итоговая сумма должна корректно рассчитываться для валидных входов согласно разделу 4.1 (см. формулы в TestPlan).

2 Требования к документации

- Руководство пользователя (User_Manual.md)
- Тест-план и набор тест-кейсов (TestPlan.md)
- Отчет о найденных дефектах другой команды (будет добавлен после обмена)
- Итоговый отчет по работе (Report_Practice_1.md)

3 Порядок контроля и приёмки

- Функциональное тестирование методом «черного ящика» (эквивалентные классы, граничные значения, негативные сценарии).
- Проверка корректности документации (ревью, чек-листы).
- Приемочные испытания: запуск, ввод типовых сценариев, сверка результатов с ожидаемыми значениями.

4 Этапы и сроки разработки

- Инициация/планирование — 0.5 дн.
- Реализация прототипа — 1 дн.
- Подготовка документации — 1 дн.
- Внутреннее тестирование — 0.5 дн.
- Передача на внешнее тестирование — 0.5 дн

9 Описание внесённых ошибок в собственное ПО

BUG-1 (Валидация): допускаются `price <= 0` и `qty <= 0` в `calc_total` — отсутствует проверка.

BUG-2 (Логика налогообложения/скидки): налог начисляется до скидки, что завышает итог.

BUG-3 (Локализация): `parse_price` не поддерживает запятую как разделитель.

BUG-4 (CSV): заголовок файла пишется при каждом сохранении.

BUG-5 (Интерфейс): пункт меню 4 — Выход фактически сохраняет и выходит без подтверждения.

BUG-6 (Границы): нет верхнего ограничения на количество (например, 100+).

BUG-7 (Округление): округление до 1 знака вместо 2-х.

BUG-8 (UI/текст): неконсистентное указание валюты и орфографическая ошибка «Ошибка».

10 Техническое задание (ТЗ) и документацию программного продукта другой команды

1. 10.1 Введение

Настоящий документ описывает требования к небольшому программному продукту «Учёт экспедиции». Программа предназначена для хранения списка участников экспедиции с их ролями, просмотра списка, фильтрации и подсчёта численности. Реализация ориентирована на тестирование методом «чёрного ящика».

2. Основания для разработки

Работа выполняется в рамках учебной практики по дисциплине «Тестирование программного обеспечения» для подгруппы «Экспедиция 74-23».

3. Назначение разработки

Программный продукт обеспечивает базовый учёт состава экспедиции:

- добавление участника с указанием роли;
- вывод полного списка участников;
- вывод участников по выбранной роли;
- удаление участника;
- подсчёт общего количества участников.

4. Требования к программе

4.1 Функциональные требования

- **F1. Добавление участника.** Команда: add <Имя> <Роль>.
- **F2. Вывод всех участников.** Команда: list — сортировка по алфавиту по полю «Имя».
- **F3. Фильтрация по роли.** Команда: list --role <Роль> — список участников выбранной роли.
- **F4. Подсчёт численности.** Команда: count — вывод общего количества участников.
- **F5. Удаление участника.** Команда: remove <Имя> — удаление записи по имени.

4.2 Ограничения и валидация ввода

- **Имя:** от 2 до 40 символов; допустимы буквы кириллицы/латиницы, пробел и дефис.
- **Роль:** выбирается из фиксированного списка: штурман, водитель, грузчик, механик. Сравнение — без учёта регистра.
- **Уникальность:** имена уникальны (сравнение — без учёта регистра).
- Недопустимые значения сопровождаются понятным сообщением об ошибке, выполнение команды не приводит к изменению данных.

4.3 Нефункциональные требования

- **Тип приложения:** консольная утилита (CLI).
- **Язык интерфейса:** русский.
- **Производительность:** операции выполняются интерактивно (менее 0,5 сек для списков до 10 000 записей).
- **Надёжность:** при ошибке ввода программа не завершается аварийно.
- **Безопасность:** работа с локальным файлом данных без сетевого доступа.
- **Портируемость:** Windows 10/11, macOS 12+, Linux (совместимость командного интерфейса).

4.4 Совместимость и зависимости

- Интерпретатор/движок выбранного языка программирования и стандартная библиотека. Внешние сетевые или БД-зависимости отсутствуют.

4.5 Хранение данных

- Персистентность обеспечивается локальным файлом members.json в рабочей директории приложения.
- Формат данных — JSON (см. Приложение А).

5. Требования к интерфейсу

Интерфейс командной строки. Формат сообщений:

- **Успех:** ОК: <сообщение>

- **Ошибка валидации:** ERR: <описание проблемы>
- **Неизвестная команда:** ERR: unknown command

Примеры ожидаемых ответов приведены в «Руководстве пользователя».

6. Критерии приёмки

- Реализованы функции F1–F5 и вся валидация из п. 4.2.
- Команды возвращают корректные сообщения об успехе/ошибках.
- Данные сохраняются и корректно восстанавливаются между запусками.
- Тестовое покрытие сценариями «чёрного ящика»: не менее 15 тест-кейсов, включая не менее 5 негативных; успешно выполняется $\geq 95\%$ тест-кейсов при корректной конфигурации.

7. Состав документации

- Настоящее ТЗ.
- Руководство пользователя.
- Инструкция по установке и запуску.
- Приложение: описание формата данных.

8. Порядок контроля и приёмки

- Предварительная проверка корректности ввода и сообщений об ошибках.
- Функциональное тестирование по сценариям из Руководства пользователя.
- Приёмочные испытания: демонстрация выполнения F1–F5, проверки персистентности, сортировки и фильтрации по роли.

9. Этапы и сроки выполнения

1. Подготовка ТЗ и структуры проекта.
2. Реализация CLI-интерфейса и операций F1–F5.
3. Реализация персистентности (JSON файл).
4. Оформление документации и примерных сценариев.
5. Внутренняя проверка и фиксация версии 1.0.

Руководство пользователя:

1. Назначение

Программа «Учёт экспедиции» поддерживает добавление, просмотр, фильтрацию, удаление участников и подсчёт численности экспедиции.

2. Поддерживаемые роли

штурман, водитель, грузчик, механик (регистр не важен).

3. Команды и примеры

3.1 Добавление участника

Команда:

add <Имя> <Роль>

Примеры и ответы:

add Анна штурман

ОК: добавлен "Анна" (штурман)

add АННА ШТУРМАН

ERR: такое имя уже существует

3.2 Просмотр всех участников

Команда:

list

Ответ: нумерованный список, отсортированный по алфавиту.

Пример:

1) Анна — штурман

2) Борис — механик

3) Иван — водитель

3.3 Фильтрация по роли

Команда:

list --role <Роль>

Пример:

list --role механик

1) Борис — механик

При отсутствии записей выводится:

нет записей

3.4 Подсчёт участников

Команда:

count

Пример ответа:

Всего участников: 3

3.5 Удаление участника

Команда:

remove <Имя>

Примеры и ответы:

remove Иван

ОК: удалён "Иван"

remove Пётр

ERR: запись с именем "Пётр" не найдена

3.6 Сообщения об ошибках (типовые)

- ERR: имя должно быть 2–40 символов (буквы, пробел, дефис)
- ERR: роль должна быть одной из: штурман, водитель, грузчик, механик
- ERR: такое имя уже существует
- ERR: запись с именем "<Имя>" не найдена
- ERR: unknown command

Инструкция по установке и запуску

1. Системные требования

- Windows 10/11, macOS 12+ или Linux.
- Установленный интерпретатор/рантайм выбранного языка программирования.

2. Запуск

- Запуск выполняется из командной строки/терминала в каталоге с программой.
- Файл данных `members.json` создаётся автоматически при первом успешном добавлении участника.
- Завершение работы — стандартным способом ОС (например, `Ctrl+C` в терминале).

Приложение А. Формат файла данных (JSON)

А.1 Структура

Файл `members.json` содержит массив объектов следующего вида:

```
[  
  {  
    "name": "Анна",  
    "role": "штурман"  
  },  
  {  
    "name": "Борис",  
    "role": "механик"  
  }  
]
```

А.2 Требования к данным

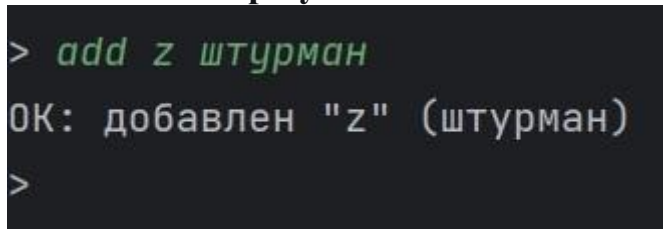
- Поле `name` — строка 2–40 символов; уникальна без учёта регистра.
- Поле `role` — одно из: штурман, водитель, грузчик, механик (хранится в нижнем регистре либо в том виде, в котором введено — по реализации; поведение интерфейса при сравнении регистр-независимое).
- При повреждении файла программа должна корректно сообщить об ошибке чтения без аварийного завершения.

13 Тестирование программного продукта другой группы

Ошибка №1: Проверка валидации длины имени при добавлении участника

Ожидаемый результат Система возвращает ошибку: "имя должно быть (2–40 символов)"

Фактический результат



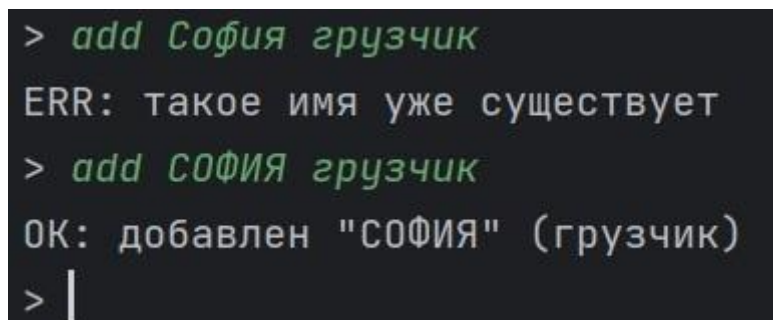
```
> add z штурман
OK: добавлен "z" (штурман)
>
```

Рисунок 1 – Система принимает короткое имя: "OK: добавлен 'z' (штурман)"

Ошибка №2: Проверка регистра о независимости при проверке дубликатов имен

Ожидаемый результат: Система возвращает ошибку: "такое имя уже существует"

Фактический результат



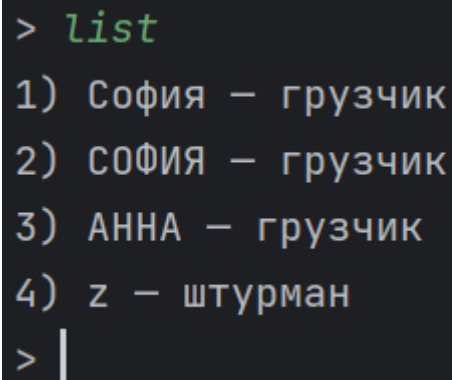
```
> add София грузчик
ERR: такое имя уже существует
> add СОФИЯ грузчик
OK: добавлен "СОФИЯ" (грузчик)
> |
```

Рисунок 2 – Система принимает имя как уникальное: "OK: добавлен СОФИЯ (грузчик)"

Ошибка №3: Проверка сортировки списка участников по возрастанию

Ожидаемый результат: Список отображается в порядке по алфавиту

Фактический результат



```
> list
1) София – грузчик
2) СОФИЯ – грузчик
3) АННА – грузчик
4) z – штурман
> |
```

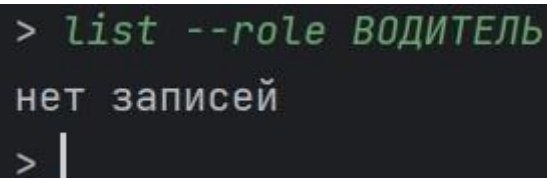
Рисунок 3 – Список отображается в обратном порядке

Ошибка №4: Проверка регистра о независимости фильтрации по роли

Ожидаемый результат: Система отображает участника с ролью

"водитель"

Фактический результат



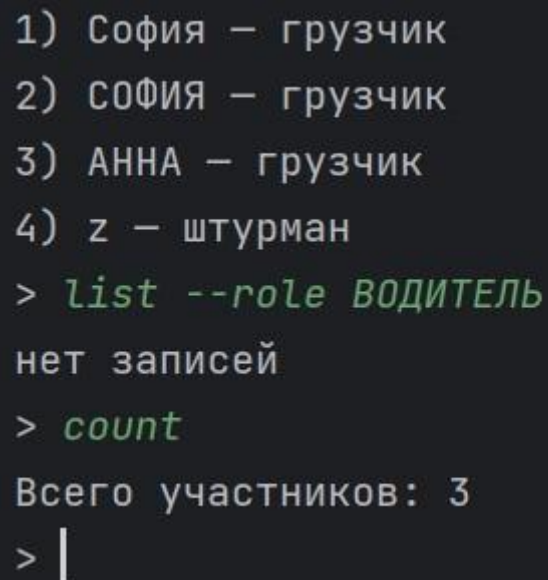
```
> list --role ВОДИТЕЛЬ
нет записей
> |
```

Рисунок 4 – Система возвращает: "нет записей"

Ошибка №5: Проверка корректности подсчета участников

Ожидаемый результат Система возвращает: "Всего участников: 4"

Фактический результат



```
1) София – грузчик
2) СОФИЯ – грузчик
3) АННА – грузчик
4) z – штурман
> list --role ВОДИТЕЛЬ
нет записей
> count
Всего участников: 3
> |
```

Рисунок 5 – Система возвращает: "Всего участников: 3"

Заключение

Тестирование программного продукта "Учёт экспедиции" выявило ряд критических недочётов в реализации основных функций системы. Найденные ошибки (№1–№5) демонстрируют системные проблемы с валидацией входных данных и соблюдением заявленных в ТЗ требований. В частности:

- Ошибка №1 показывает отсутствие контроля минимальной длины имени, что нарушает базовые требования к целостности данных и позволяет создавать некорректные записи.
- Ошибка №2 свидетельствует о неправильной реализации проверки уникальности имен - система не обеспечивает регистронезависимое сравнение, что приводит к созданию дублирующихся записей.
- Ошибка №3 выявляет нарушение алгоритма сортировки, при котором список отображается в обратном алфавитном порядке, что противоречит явному требованию ТЗ о сортировке по возрастанию.
- Ошибка №4 демонстрирует некорректную работу фильтрации по роли, которая также не обеспечивает заявленную в ТЗ регистронезависимость, делая функцию практически неработоспособной.
- Ошибка №5 указывает на серьезную ошибку в логике подсчета участников, когда система возвращает заниженное количество записей, что искажает базовую статистику.

Для исправления ситуации необходимо:

- Переработать механизмы валидации входных данных с реализацией проверки граничных значений.
- Исправить алгоритмы сравнения строк для обеспечения регистронезависимости при проверке уникальности и фильтрации.
- Восстановить корректную логику сортировки и подсчета записей в соответствии с техническим заданием.
- Провести комплексное модульное тестирование всех исправленных функций для подтверждения их работоспособности.
- Выполнить регрессионное тестирование для обеспечения отсутствия побочных эффектов после внесения исправлений.

Только после устранения указанных системных ошибок и подтверждения соответствия всем требованиям ТЗ можно говорить о готовности продукта к эксплуатации

Анализ технической документации

В ходе анализа было изучено предоставленное техническое задание (ТЗ) команды "Экспедиция". Была проведена оценка полноты, логичности и непротиворечивости требований.

Оценка полноты технического задания

Техническое задание структурировано и охватывает основные аспекты продукта, однако при детальном рассмотрении выявляется ряд существенных пробелов:

1) Отсутствие требований к обработке исключительных ситуаций. В документе не описано поведение системы в следующих случаях:

- Повреждение или отсутствие файла `members.json
- Отсутствие прав на запись в рабочую директорию
- Некорректный формат JSON в файле данных
- Одновременный запуск нескольких экземпляров программы

2) Неполная спецификация валидации входных данных. Хотя указаны базовые ограничения, отсутствуют требования к:

- Обработке пробелов в начале и конце строк
- Вводу специальных символов кроме букв, пробела и дефиса
- Проверке максимальной длины роли
- Обработке пустых параметров команд

3) Отсутствие требований к производительности при больших объемах данных. Указано время отклика для 10 000 записей, но не описано:

- Поведение при достижении лимита оперативной памяти
- Время запуска при большом файле данных
- Производительность операций поиска и фильтрации

Серьезные недостатки:

4) Недостаточная спецификация формата данных JSON. Требования к структуре файла описаны поверхностно:

- Не указана кодировка файла (UTF-8 обязательно для кириллицы)
- Отсутствуют требования к порядку полей в объектах
- Не описана обработка дополнительных полей в JSON
- Нет спецификации для пустого массива

5) Отсутствие требований к миграции данных. Не предусмотрены сценарии:

- Обновление формата данных между версиями
- Восстановление из резервной копии
- Импорт данных из внешних источников

Оценка логичности и согласованности

Требования в целом непротиворечивы, однако некоторые формулировки допускают двойственное толкование:

Неоднозначные требования:

1) Регистронезависимость. В разделе 4.2 указано "сравнение — без учёта регистра", но в Приложении А допускается хранение роли "в том виде, в котором введено". Это создает противоречие между логикой сравнения и хранения данных.

- Формат сообщений об ошибках. Указаны общие шаблоны, но не детализированы:
 - Должны ли сообщения быть на русском языке всегда
 - Форматирование сложных сообщений с переменными
 - Единообразие пунктуации и стиля

2) Сортировка списка. Требуется "сортировка по алфавиту", но не уточняется:

- Учет регистра при сортировке
- Локализация сортировки (русский алфавит)
- Поведение с именами в разных языках

Неконкретные требования:

1) Процедура удаления данных. Указана команда remove, но не описано:

- Подтверждение удаления для предотвращения ошибок
- Восстановление случайно удаленных данных
- Влияние удаления на целостность данных

2) Работа с файловой системой. Не определены требования к:

- Блокировке файла при работе
- Атомарности операций записи
- Резервному копированию перед изменением

Выявленные противоречия с реализацией

Анализ выявленных ошибок показывает системные расхождения между ТЗ и реализацией:

Критические несоответствия:

Ошибка №1: Требование валидации длины имени (2-40 символов) не выполняется

Ошибка №2: Регистронезависимость проверки дубликатов не реализована

Ошибка №3: Сортировка по алфавиту работает в обратном порядке

Ошибка №4: Регистронезависимость фильтрации по роли не работает

Ошибка №5: Функция подсчета возвращает неверные результаты