Практическая работа №1

Тема ПП: «Учёт экспедиции»
Подгруппа: «Экспедиция 74-23»
Дисциплина: Тестирование программного обеспечения
Версия документа: 1.0

Техническое задание

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:** OK: <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 ≥ 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
OK: 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
OK: 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"
  }
]
```

A.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.

Конец Части 1.

Практическая работа №1 — Часть 2

Отчёт по тестированию чужого программного продукта (метод «чёрного ящика»)
Подгруппа тестирующих: «Экспедиция 74-23»
Версия документа: 1.0

---

1. Аннотация

Проведено функциональное тестирование чужого консольного приложения (CLI) для учёта участников экспедиции. Применялись техники «чёрного ящика»: классы эквивалентности, анализ граничных значений, негативные сценарии и предположение ошибок. Сформированы тест-кейсы, выполнён прогон, заведены дефекты, подготовлены выводы и рекомендации.

2. База тестирования (test basis)

- Техническое задание (ТЗ) на тестируемый ПП, версия 1.0.
- Руководство пользователя, версия 1.0.
- Описание интерфейса CLI-команд.

3. Объект тестирования (SUT)

Класс приложения: консольная утилита учёта участников.
Ключевые функции (согласно ТЗ тестируемой стороны):

F1 — добавление участника add <Имя> <Роль>;

F2 — вывод всех участников list (сортировка по алфавиту по имени);

F3 — фильтрация по роли list --role <Роль> (регистр роли незначим);

F4 — подсчёт количества count;

F5 — удаление по имени remove <Имя>;

Валидация: имя 2–40 символов; роль из фиксированного перечня; имена уникальны без учёта регистра; сообщения об ошибках — информативные.

4. Границы и объём тестирования

В scope: функциональность F1–F5, валидация ввода, сортировка/фильтрация, сообщения об ошибках, персистентность локального файла.
Out of scope: производительность >10k записей, устойчивость к отказам ОС/ФС, безопасность и сетевые сценарии.

5. Среда и инструменты

- ОС: macOS 14.5 / Windows 11 (терминал/PowerShell).
- Данные: локальный файл members.json в каталоге запуска.
- Фиксация результатов: таблицы тест-кейсов и баг-репортов (Markdown/Docx).

6. Подход к тестированию

- **Техники:** классы эквивалентности (валидные/невалидные имена и роли), граничные значения (1/2/40/41 символ), негативные сценарии (дубликаты, неизвестные команды), «ошибкогадание» (case-insensitive сравнения, сортировка).
- **Критерии входа:** доступна сборка 1.0, предоставлены ТЗ и руководство.
- **Критерии выхода:** выполнены ≥15 тест-кейсов (≥5 негативных), дефекты заведены и классифицированы, сформированы выводы.

7. Трассируемость требований

Требование	Описание	Тест-кейсы
F1	Добавление участника	TC-01, TC-02, TC-03, TC-04, TC-05, TC-06
F2	Вывод всех участников (сортировка)	TC-07, TC-08
F3	Фильтрация по роли (регистр незначим)	TC-09, TC-10
F4	Подсчёт количества	TC-11
F5	Удаление по имени	TC-12, TC-13
Валидация	Сообщения об ошибках	TC-02, TC-03, TC-04, TC-05, TC-14, TC-15
Персистентность	Сохранение/восстановление	TC-16, TC-17
Неизвестные команды	Обработка ошибок	TC-18

---

8. Тест-кейсы и результаты первого прогона

Формат записи: **ID — Название — Предусловия — Шаги — Ожидаемый результат — Фактический результат — Статус**.

TC-01 — Добавление валидного участника

Пустая база. → add Анна штурман → OK, запись создана. → OK, запись создана. → **Passed**

TC-02 — Имя слишком короткое (1 символ)

Пустая база. → add А штурман → Ошибка валидации длины имени. → Ошибка валидации выведена. → **Passed**

TC-03 — Имя слишком длинное (41 символ)

Пустая база. → add ОченьОченьОченьОченьОченьДлинноеИмя штурман → Ошибка валидации. → Ошибка валидации выведена. → **Passed**

TC-04 — Роль вне перечня

Пустая база. → add Борис инженер → Ошибка «роль должна быть одной из…». → Выведено «ERR: unknown command». → **Failed**

TC-05 — Дубликат имени без учёта регистра

В базе: «Анна, штурман». → add АННА водитель → Ошибка «такое имя уже существует». → Запись создана как новая. → **Failed**

TC-06 — Добавление нескольких валидных участников

Пустая база. → add Анна штурман, add Борис механик, add Иван водитель → Три записи созданы. → Три записи созданы. → **Passed**

TC-07 — Сортировка списка (возрастание по имени)

В базе: Анна, Борис, Иван. → list → Порядок: «Анна, Борис, Иван». → Порядок: «Иван, Борис, Анна». → **Failed**

TC-08 — Пустой список — корректный вывод

Пустая база. → list → «нет записей». → «нет записей». → **Passed**

TC-09 — Фильтрация по роли (регистр незначим)

В базе: «АННА, ШТУРМАН». → list --role штурман → Анна в выдаче. → Пусто. → **Failed**

TC-10 — Фильтрация по отсутствующей роли

В базе нет «грузчик». → list --role грузчик → «нет записей». → «нет записей». → **Passed**

TC-11 — Подсчёт количества

В базе: Анна, Борис, Иван. → count → «Всего участников: 3». → «Всего участников: 2». → **Failed**

TC-12 — Удаление существующей записи

В базе: Иван. → remove Иван → Успех, запись удалена. → Успех, запись удалена. → **Passed**

TC-13 — Удаление несуществующей записи

Пустая база. → remove Пётр → «запись не найдена». → «запись не найдена». → **Passed**

TC-14 — Неверная команда

Любая база. → foo → «ERR: unknown command». → «ERR: unknown command». → **Passed**

TC-15 — Неверный формат параметров

Любая база. → add ТолькоИмя → Сообщение о правильном использовании команды. → Сообщение неконкретное «unknown command». → **Failed**

TC-16 — Персистентность: сохранение и восстановление

Добавить записи → завершить → запустить → list. → Данные восстановлены. → Данные восстановлены. → **Passed**

TC-17 — Персистентность: пустая база после удаления всех

Добавить 1 запись → удалить → завершить → запустить → list. → «нет записей». → Файл отсутствует, при запуске ошибка чтения. → **Failed**

TC-18 — Сообщение об ошибке роли (качество текста)

add Антон архитектор. → Текст ошибки: «роль должна быть одной из: …». → Выдано «ERR: unknown command». → **Failed**

Итого прогон: 18 кейсов, **Passed: 11**, **Failed: 7**, показатель прохождения **61,1%**.

---

9. Заведённые дефекты

BUG-01 — Неверный текст ошибки при неизвестной роли

Серьёзность: **Major**. Ожидалось: конкретная подсказка по допустимым ролям. Факт: выводится «unknown command».

BUG-02 — Дубликаты имён не обнаруживаются без учёта регистра

Серьёзность: **Major**. Добавлены «Анна» и «АННА» как разные записи.

BUG-03 — Сортировка списка по убыванию вместо возрастания

Серьёзность: **Minor**. Порядок «Иван, Борис, Анна» при требуемом алфавитном возрастании.

BUG-04 — Фильтрация по роли чувствительна к регистру

Серьёзность: **Major**. list --role штурман не находит запись «ШТУРМАН».

BUG-05 — Счётчик count занижает результат на 1

Серьёзность: **Critical**. При трёх записях выводится «2».

BUG-06 — Некорректное сообщение при неверном формате параметров

Серьёзность: **Minor**. Вместо подсказки по синтаксису возвращается «unknown command».

BUG-07 — Персистентность при пустой базе

Серьёзность: **Major**. После удаления всех записей файл данных отсутствует/портится; при повторном запуске ошибка вместо «нет записей».

---

10. Метрики тестирования

- Всего тест-кейсов: 18
- Пройдено: 11 (**61,1%**)
- Провалено: 7
- Дефектов заведено: 7 (Critical: 1, Major: 4, Minor: 2)
- Плотность дефектов (на 10 кейсов): ~3,9

11. Риски и наблюдения

- Несоответствия текстов ошибок ТЗ усложняют пользовательское восстановление после ошибки.
- Чувствительность к регистру нарушает единые правила сравнения, приводит к скрытым дубликатам и пропускам в фильтрации.
- Ошибка в count критична: подрывает доверие ко всем отчётным показателям.
- Проблемы с персистентностью на пустой базе создают риск потери данных и препятствуют повторным запускам.

12. Рекомендации

1. Унифицировать сравнение строк (имя, роль) с нормализацией регистра и пробелов.
2. Исправить алгоритм сортировки списка на возрастание по полю «Имя».
3. Привести сообщения об ошибках к ТЗ: конкретные тексты и перечень допустимых значений.
4. Пересмотреть реализацию count (возвращать фактическое количество записей).
5. Пересобрать слой сохранения: корректная инициализация пустого файла и чтение отсутствующего/пустого массива.
6. Добавить юнит-тесты на валидацию, сортировку, фильтрацию и персистентность.

13. Заключение

Функциональность в целом соответствует задекларированным областям применения, однако обнаружены существенные отклонения от требований по валидации, сортировке, подсчёту и персистентности. До исправления критического дефекта в count и ошибок обработки ролей рекомендуем считать релиз **непригодным к приёмке**. После устранения дефектов и повторного прогона ожидается достижение показателя прохождения ≥95% согласно критериям качества.

Приложение A. Классы эквивалентности и граничные значения

- **Имя**:

Валидные — длина [2…40], символы: буквы, пробел, дефис.

Невалидные — длина <2, длина >40, посторонние символы.

- **Роль**:

Валидные — из списка; регистр незначим.

Невалидные — вне списка, пустые, с опечатками.

- **Команды**:

Валидные — add, list, list --role, count, remove.

Невалидные — неизвестные команды, неполные параметры.

Приложение B. Шаблон баг-репорта

- **ID:** BUG-NNN
- **Заголовок:** кратко (что/где)
- **Окружение:** ОС, версия приложения
- **Предусловия:** состояние базы/файлов
- **Шаги воспроизведения:** 1…2…3…
- **Ожидаемый результат:** …
- **Фактический результат:** …
- **Серьёзность / Приоритет:** …
- **Вложения:** скриншот/лог/видео

Приложение C. Журнал команд (фрагмент)

```
add Анна штурман
add АННА водитель
list
list --role штурман
count
remove Иван
add ОченьОченьОченьОченьОченьДлинноеИмя штурман
add Борис инженер
list
```

Конец Части 2.