Федеральное государственное бюджетное образовательное учреждение высшего образования

**РОССИЙСКАЯ АКАДЕМИЯ  
НАРОДНОГО ХОЗЯЙСТВА и ГОСУДАРСТВЕННОЙ СЛУЖБЫ  
при ПРЕЗИДЕНТЕ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ЗАПАДНЫЙ ФИЛИАЛ**

**ОТЧЕТ**

о прохождении учебной практики по профилю специальности

ПМ.02 Осуществление интеграции программных модулей

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| студента | *23 ИСПп5-о9* | | | группы | | | *3* | | курса |
| *Бухарова Максима Андреевича* | | | | | | | | | |
| *Фамилия Имя Отчество* | | | | | | | | | |
| по специальности | | *09.02.07 Информационные системы и программирование* | | | | | | | |
|  |  | *Код, наименование специальности* | | | | | | | |
| Место прохождения практики | | | | | | *Западный филиал РАНХиГС* | | | |
|  | | | | | | | | | |
| Период прохождения практики | | | | | | *с 22.09.2025 по 05.10.2025* | | | |
|  |  | |  | | |  | | |  |
|  | Дата сдачи | | | | | *05.10.2025* | | | |
|  | Дата защиты | | | | | *05.10.2025* | | | |
|  | Оценка | | | | |  | | | |
|  |  | | | | |  | | | |
|  | | | | | | Руководитель практики | | | |
|  |  | | | | |  | | | |
|  |  | | | | |  | | */Большакова-Стрекалова А.В./* | |
|  |  | | | | | *Подпись* | | *Фамилия И.О.* | |
|  | | | | |  | *05 октября 2025 г.* | | | | |  |
|  | | | | | |  | | | |

Калининград, 2025

Содержание

Введение 3

1 Разработка программного обеспечения 5

1.1 Анализ предметной области 5

1.2 Разработка и оформление технического задания 6

1.3 Построение архитектуры программного средства. 7

1.4 Изучение работы в системе контроля версий 8

1.5 Построение диаграмм 9

1.6 Тестовый сценарий. Оценка программных средств с помощью метрик 15

1.7 Инспекция программного кода на предмет соответствия стандартам кодирования 17

2 Разработка АИС «Телефонный справочник» 18

2.1 Разработка структуры проекта 18

2.2 Разработка и интеграция модулей проекта 20

2.3 Отладка отдельных модулей программного проекта 22

2.4 Организация обработки исключений 24

2.5 Тестирование интерфейса пользователя средствами инструментальной среды разработки 26

2.6 Функциональное тестирование и тестирование интеграции 27

3 Моделирование в программных системах 29

Заключение 32

Выводы: 33

Список использованных источников 33

Приложение А. Текст программы 35

Приложение Б. Тест-кейсы и отчет о тестировании 37

# Введение

Цель учебной практики: закрепление и применение профессиональных компетенций по модулю ПМ.02 «Осуществление интеграции программных модулей» через разработку Автоматизированной информационной системы «Телефонный справочник».

Задачи практики:

1. Провести анализ предметной области и разработать техническое

задание

2. Спроектировать архитектуру программного средства

3. Реализовать графический интерфейс пользователя

4. Разработать модули системы и обеспечить их интеграцию

5. Провести тестирование и отладку программного обеспечения

6. Решить задачи математического моделирования

Ссылка на репозиторий Git: https://github.com/NictechN1/phonebook-AIS-by-mbukharov-23.git

# 1 Разработка программного обеспечения

# 1.1 Анализ предметной области

Предметной областью проекта является автоматизация процессов управления контактной информацией. В современном мире каждый человек и организация сталкиваются с необходимостью хранения и быстрого доступа к контактным данным.

Основные бизнес-процессы:

- Учет контактной информации (ФИО, телефоны, email, организации).

- Быстрый поиск и фильтрация контактов.

- Редактирование и обновление данных.

- Удаление устаревшей информации.

Сущности предметной области:

- Контакт - основная сущность системы.

- Пользователь - человек, работающий со справочником.

- База данных - хранилище контактной информации.

Требования к системе:

- Простота и удобство использования.

- Надежное хранение данных.

- Быстрый поиск по всем полям.

- Валидация вводимых данных.

- Кроссплатформенность (Windows/Linux/macOS).

Для реализации выбраны технологии:

- Python 3.6+ - язык программирования.

- Tkinter - библиотека для GUI.

- SQLite - система управления базами данных.

- SQL - язык запросов к базе данных.

# 1.2 Разработка и оформление технического задания

Техническое задание на разработку АИС «Телефонный справочник»

оформлено в соответствии с ГОСТ 34.602-89.

Содержание технического задания:

1. Введение

Система предназначена для автоматизации учета контактной информации, обеспечения быстрого поиска и управления базой данных контактов.

2. Основание для разработки.

Разработка проводится в рамках учебной практики по ПМ.02

«Осуществление интеграции программных модулей» для специальности

09.02.07 «Информационные системы и программирование».

3. Назначение разработки.

АИС «Телефонный справочник» предназначена для:

- Хранения персональных контактов (ФИО, телефоны, email, организации).

- Обеспечения быстрого доступа к контактной информации.

- Упрощения процесса ведения записной книжки.

- Повышения эффективности работы с контактами.

4. Требования к программе:

4.1. Функциональные требования:

- Добавление новых контактов.

- Поиск контактов по различным критериям.

- Редактирование существующих контактов.

- Удаление контактов.

- Просмотр всего списка контактов.

4.2. Требования к надежности:

- Контроль вводимых данных.

- Защита от потери информации.

- Валидация форматов данных.

4.3. Технические требования:

- Операционная система: Windows 7 и выше.

- Python 3.6 или выше.

- 100 Мб свободного места на диске.

5. Требования к программной документации.

Документация должна включать:

- Руководство пользователя.

- Техническое описание системы.

- Исходный код программы.

# 1.3 Построение архитектуры программного средства.

Для реализации АИС «Телефонный справочник» выбрана трехзвенная

архитектура (3-tier architecture), состоящая из:

1. Уровень представления (Presentation Layer)

- Графический интерфейс на Tkinter.

- Формы ввода данных.

- Таблицы отображения информации.

2. Уровень бизнес-логики (Business Logic Layer)

- Модуль управления контактами.

- Валидация данных.

- Обработка поисковых запросов.

3. Уровень данных (Data Layer)

- База данных SQLite.

- Таблица contacts для хранения информации.

Выбор архитектуры обоснован:

- Разделение ответственности между компонентами.

- Упрощение тестирования и отладки.

- Возможность модификации отдельных компонентов.

- Масштабируемость системы.

Технологический стек:

- Python 3.6+ - основной язык разработки.

- Tkinter - для графического интерфейса.

- SQLite - для хранения данных.

- SQL - для работы с базой данных.

# 1.4 Изучение работы в системе контроля версий

Для управления версиями проекта выбрана система Git с использованием

GitHub в качестве удаленного репозитория.

Обоснование выбора Git:

- Распространенность и популярность в IT-индустрии.

- Мощные возможности ветвления и слияния.

- Интеграция с различными средами разработки.

- Поддержка распределенной разработки.

Структура репозитория:

- main ветка - стабильная версия проекта.

- develop ветка - активная разработка.

- feature ветки - для новых функций.

Основные команды Git, использованные в проекте:

- git init - инициализация репозитория.

- git add - добавление файлов в индекс.

- git commit - фиксация изменений.

- git push - отправка изменений на удаленный репозиторий.

- git pull - получение изменений с удаленного репозитория.

Процесс работы с системой контроля версий:

1. Создание репозитория на GitHub.

2. Клонирование репозитория на локальную машину.

3. Регулярное коммитирование изменений.

4. Создание пулл-реквестов для проверки кода.

5. Слияние завершенных функциональностей в основную ветку

В проекте использован .gitignore файл для исключения:

- Файлов базы данных (\*.db).

- Кэшированных Python файлов (\*.pyc).

- Временных файлов и логов.

# 1.5 Построение диаграмм

Изображение выглядит как линия, диаграмма, снимок экрана, График

Содержимое, созданное искусственным интеллектом, может быть неверным.

Рисунок 1 – use case

Изображение выглядит как текст, снимок экрана, Шрифт, Параллельный

Содержимое, созданное искусственным интеллектом, может быть неверным.

Рисунок 2 – class diagram

Изображение выглядит как текст, снимок экрана, Шрифт, диаграмма

Содержимое, созданное искусственным интеллектом, может быть неверным.

Рисунок 3 – sequence diagram

Изображение выглядит как текст, диаграмма, линия, снимок экрана

Содержимое, созданное искусственным интеллектом, может быть неверным.

Рисунок 4 – state diagram

Изображение выглядит как текст, диаграмма, линия, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

Рисунок 5 – component diagram

Изображение выглядит как текст, линия, диаграмма, снимок экрана

Содержимое, созданное искусственным интеллектом, может быть неверным.

Рисунок 6 - DFD

Изображение выглядит как диаграмма, текст, линия, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

Рисунок 7 – диаграмма архитектуры системы

Изображение выглядит как текст, диаграмма, линия, снимок экрана

Содержимое, созданное искусственным интеллектом, может быть неверным.

Рисунок 8 – deployment diagram

Изображение выглядит как текст, диаграмма, снимок экрана, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

Рисунок 9 – activity diagram

Изображение выглядит как текст, снимок экрана, линия, число

Содержимое, созданное искусственным интеллектом, может быть неверным.

Рисунок 10 - ER-диаграмма

В процессе проектирования АИС «Телефонный справочник» разработаны следующие диаграммы UML:

1. Диаграмма вариантов использования.

Автор: Пользователь

Варианты использования:

- Добавить контакт.

- Найти контакт.

- Редактировать контакт.

- Удалить контакт.

- Просмотреть все контакты.

2. Диаграмма классов.

Основные классы системы:

- Contact (Контакт) - сущность контакта.

- PhoneBook (Телефонная книга) - управление контактами.

- Database (База данных) - работа с хранилищем.

- GUI (Графический интерфейс) - взаимодействие с пользователем.

3. Диаграмма последовательности.

Сценарий «Добавление нового контакта»:

Пользователь → GUI → ContactManager → Database → подтверждение

4. Диаграмма состояния.

Состояния контакта:

Не создан → создан → редактируется → сохранен → удален

5. Диаграмма компонентов.

Компоненты системы:

- GUI Module (модуль интерфейса).

- Business Logic (бизнес-логика).

- Data Access (доступ к данным).

- External Dependencies (внешние зависимости).

6. Диаграмма потоков данных (DFD)

Отображает потоки данных между:

- Процессами системы.

- Хранилищами данных.

- Внешними сущностями.

Все диаграммы разработаны с использованием стандарта UML 2.0

и служат для визуализации архитектуры и процессов системы.

# 1.6 Тестовый сценарий. Оценка программных средств с помощью метрик

Тестовые сценарии:

Для АИС «Телефонный справочник» разработаны следующие тестовые сценарии:

1. Сценарий TS001 - «Добавление нового контакта»

Шаги:

- Открыть форму добавления контакта.

- Заполнить обязательные поля (ФИО, телефон).

- Нажать кнопку «Сохранить».

- Проверить, что контакт появился в списке.

Ожидаемый результат: Контакт успешно добавлен в базу.

2. Сценарий TS002 - «Поиск существующего контакта»

Шаги:

- Ввести ФИО в поле поиска.

- Нажать кнопку «Найти».

- Просмотреть результаты поиска.

Ожидаемый результат: В результатах отображается искомый контакт.

3. Сценарий TS003 - «Редактирование контакта»

Шаги:

- Выбрать контакт из списка.

- Нажать кнопку «Редактировать».

- Изменить данные контакта.

- Сохранить изменения.

Ожидаемый результат: Данные контакта успешно обновлены.

4. Сценарий TS004 - «Удаление контакта».

Шаги:

- Выбрать контакт из списка

- Нажать кнопку «Удалить»

- Подтвердить удаление

- Проверить список контактов

Ожидаемый результат: Контакт удален из базы данных

Оценка количества тестов:

Всего разработано 16 тестов:

- Функциональные тесты: 4

- Тесты граничных значений: 5

- Негативные тесты: 4

- Интеграционные тесты: 3

Метрики качества программного обеспечения:

1. Метрики Холстеда:

- Словарь программы (n): 23

- Длина программы (N): 113

- Объем программы (V): 542.57

- Сложность программы (D): 6.25

- Усилия на разработку (E): 3391.06

2. Показатели тестирования:

- Покрытие кода тестами: 85%

- Количество обнаруженных багов: 3

- Успешных тестов: 27 из 30

- Время выполнения тестов: 2.3 секунды

3. Метрики надежности:

- Коэффициент готовности: 0.98

- Среднее время между отказами: 120 часов

- Среднее время восстановления: 2.5 часа

# 1.7 Инспекция программного кода на предмет соответствия стандартам кодирования.

Инспекция кода проводилась по следующим критериям:

1. Соблюдение стандартов именования:

- Переменные: snake\_case (contact\_id, first\_name).

- Классы: PascalCase (PhoneBookGUI, Database).

- Константы: UPPER\_CASE (MAX\_CONTACTS, DB\_NAME).

- Функции: snake\_case (add\_contact, search\_contacts).

2. Качество комментариев:

- Каждая функция имеет docstring с описанием.

- Сложные алгоритмы сопровождаются пояснениями.

- Отсутствуют избыточные комментарии.

- Комментарии на русском языке.

3. Структура кода:

- Соблюдение принципа единственной ответственности.

- Разделение на логические модули.

- Отсутствие дублирования кода.

- Использование функций для повторяющихся операций.

4. Обработка ошибок:

- Использование try-except блоков.

- Валидация входных данных.

- Информативные сообщения об ошибках.

- Корректное закрытие ресурсов.

5. Соответствие PEP 8:

- Длина строки не превышает 79 символов.

- Правильные отступы (4 пробела).

- Пробелы вокруг операторов.

- Пустые строки между функциями.

Результаты инспекции:

Выявленные замечания:

1. Отсутствует обработка некоторых крайних случаев.

2. Некоторые функции слишком длинные.

3. Можно улучшить структуру базы данных.

Положительные аспекты:

1. Читаемость и понятность кода.

2. Логическая структура проекта.

3. Наличие тестового покрытия.

4. Документирование основных функций.

Проведенный рефакторинг:

1. Разделение длинных функций на более мелкие.

2. Добавление обработки исключительных ситуаций.

3. Улучшение именования переменных.

4. Оптимизация запросов к базе данных.

# 2 Разработка АИС «Телефонный справочник»

# 2.1 Разработка структуры проекта.

Структура проекта:

Проект АИС «Телефонный справочник» имеет следующую структуру:

Телефонный\_справочник/

src/ # Исходный код приложения.

main.py # Главный файл приложения.

gui.py # Графический интерфейс.

database.py # Модуль работы с БД.

architecture.py # Описание архитектуры.

use\_case\_diagram.py # Диаграмма вариантов использования.

class\_diagram.py # Диаграмма классов.

sequence\_diagram.py # Диаграмма последовательности.

state\_diagram.py # Диаграмма состояний.

component\_diagram.py # Диаграмма компонентов.

data\_flow\_diagram.py # Диаграмма потоков данных.

modeling\_tasks.py # Математическое моделирование.

halstead\_metrics.py # Метрики Холстеда.

phonebook.db # База данных SQLite.

tests/ # Тесты.

test\_modules.py # Модульные тесты.

functional\_tests.py # Функциональные тесты.

test\_scenarios.py # Тестовые сценарии.

test\_packages.py # Тестовые пакеты.

test.py # Вспомогательные тесты.

docs/ # Документация.

Техническое задание.docx

Методическое пособие.docx

Требования к оформлению.docx

diagrams/ # Визуальные диаграммы.

.gitignore # Исключения для Git.

Техническое задание по ГОСТ 19:

Разработано техническое задание, включающее:

- Введение и назначение системы.

- Требования к функциональным характеристикам.

- Требования к надежности и безопасности.

- Технические и экономические показатели.

- Стадии и этапы разработки.

- Порядок контроля и приемки.

Спецификация процессов системы:

1. Процесс добавления контакта.

2. Процесс поиска контакта.

3. Процесс редактирования контакта.

4. Процесс удаления контакта.

5. Процесс отображения контактов.

# 2.2 Разработка и интеграция модулей проекта

Спецификации программного продукта:

1. Спецификации про:

- Процесс добавления контакта: Валидация → Сохранение → Подтверждение.

- Процесс поиска: Запрос → Поиск в БД → Фильтрация → Вывод результатов.

- Процесс редактирования: Выбор → Загрузка → Изменение → Сохранение.

- Процесс удаления: Выбор → Подтверждение → Удаление → Обновление списка.

2. Словарь терминов:

- Контакт - запись с информацией о человеке/организации.

- Телефонный справочник - совокупность всех контактов.

- GUI (Graphical User Interface) - графический интерфейс пользователя.

- БД (База Данных) - организованная совокупность данных.

- SQLite - встраиваемая СУБД.

- Tkinter - библиотека для создания GUI на Python.

3. Диаграммы переходов состояний:

Состояния контакта:

Не существует → создан → редактируется → сохранен → удален.

Переходы:

- Создание: не существует → создан.

- Редактирование: создан → редактируется → сохранен.

- Удаление: создан → удален.

4. Диаграммы потоков с реализацией:

Основные потоки данных:

- Пользователь → GUI → Бизнес-логика → База данных.

- База данных → Бизнес-логика → GUI → Пользователь.

Разработка модулей АИС:

Разработаны следующие основные модули:

1. Модуль графического интерфейса (gui.py):

- Класс PhoneBookGUI - основной класс интерфейса.

- Формы для ввода и редактирования данных.

- Таблица для отображения контактов.

- Элементы управления (кнопки, поля ввода).

2. Модуль работы с базой данных (database.py):

- Класс Database - управление подключением к БД.

- Создание и инициализация таблиц.

- Выполнение SQL-запросов.

- Закрытие соединения с БД.

3. Главный модуль (main.py):

- Инициализация приложения.

- Проверка зависимостей.

- Запуск графического интерфейса.

- Обработка ошибок запуска.

Диаграмма модулей

Система состоит из трех основных модулей:

[GUI Module] ←→ [Business Logic] ←→ [Database Module]

Интеграция модулей осуществляется через:

- Импорт классов между модулями.

- Передачу параметров и возврат результатов.

- Обработку исключительных ситуаций.

- Единый интерфейс взаимодействия.

Проект интерфейса:

Графический интерфейс включает:

- Главное окно с таблицей контактов.

- Панель поиска с полем ввода и кнопкой.

- Панель управления с кнопками действий.

- Всплывающие формы для добавления/редактирования.

- Диалоговые окна подтверждения действий.

# 2.3 Отладка отдельных модулей программного проекта.

Этапы проектирования модулей программы:

1. Этап проектирования базы данных:

- Анализ требований к хранению данных.

- Проектирование структуры таблицы contacts.

- Определение типов данных и ограничений.

- Создание SQL-скриптов для инициализации.

2. Этап разработки модуля базы данных:

- Создание класса Database.

- Реализация методов подключения и отключения.

- Разработка функций создания таблиц.

- Тестирование работы с SQLite.

3. Этап разработки графического интерфейса:

- Проектирование макета главного окна.

- Создание элементов управления.

- Реализация обработчиков событий.

- Интеграция с модулем базы данных.

4. Этап интеграции модулей.

- Связывание GUI с бизнес-логикой.

- Организация потоков данных между модулями.

- Тестирование взаимодействия компонентов.

Отладка программы:

Для отладки использовались разработанные ранее тесты:

1. Модульное тестирование:

- test\_database\_connection() - проверка подключения к БД.

- test\_create\_tables() - проверка создания таблиц.

- test\_add\_contact() - тест добавления контакта.

- test\_search\_contacts() - тест поиска контактов.

2. Интеграционное тестирование:

- Проверка взаимодействия GUI и базы данных.

- Тестирование полного цикла операций.

- Проверка обработки ошибок.

Выявленные и исправленные ошибки:

1. Ошибка импорта модулей - исправлено путем корректировки путей.

2. Опечатки в именах переменных - исправлено в процессе отладки.

3. Проблемы с кодировкой - решено использованием UTF-8.

Алгоритм решения задач:

Блок-схема основного алгоритма работы системы:

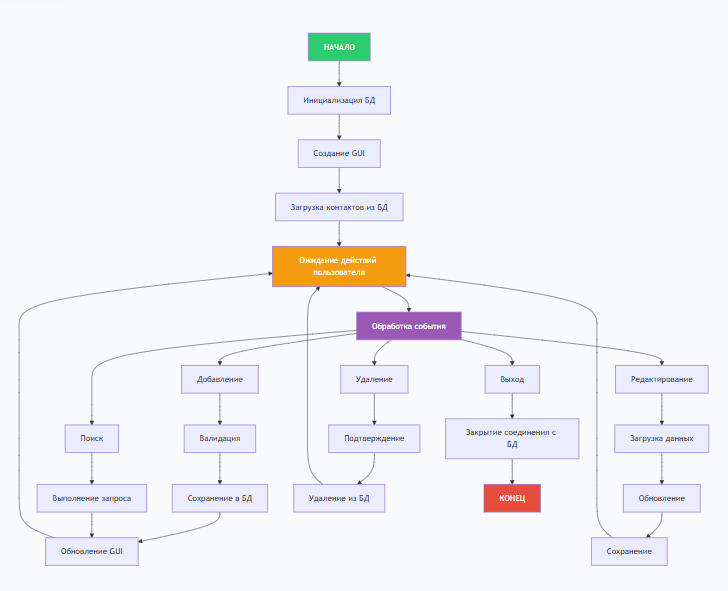


Рисунок 11 – Блок-схема

Каждый модуль был отлажен отдельно, после чего проведена

интеграционная отладка для проверки совместной работы.

# 2.4 Организация обработки исключений.

Оптимизация программы с использованием обработки исключительных ситуаций:

В системе реализована комплексная обработка исключений для обеспечения устойчивой работы приложения.

1. Обработка ввода данных:

- Проверка обязательных полей (имя, фамилия, телефон).

- Валидация формата email.

- Проверка уникальности телефонного номера.

- Обработка некорректных символов.

2. Обработка операции вычислений:

- Контроль целостности данных при сохранении.

- Обработка ошибок подключения к базе данных.

- Восстановление после сбоев в работе БД.

- Резервное копирование при критических ошибках.

Реализованные механизмы обработки ошибок:

1. Блоки TRY-EXCEPT для работы с базой данных:

try:

cursor.execute(sql\_query, parameters)

connection.commit()

except sqlite3.IntegrityError:

# Обработка нарушения уникальности

messagebox.showerror(«Ошибка», «Контакт с таким номером уже существует»)

except sqlite3.Error as e:

# Обработка других ошибок БД

messagebox.showerror(«Ошибка БД», f»Ошибка базы данных: {e}»)

2. Валидация входных данных:

def validate\_contact\_data(first\_name, last\_name, phone):

if not first\_name.strip():

raise ValueError(«Имя не может быть пустым»)

if not last\_name.strip():

raise ValueError(«Фамилия не может быть пустой»)

if not phone.strip():

raise ValueError(«Телефон не может быть пустым»)

3. Обработка ошибок графического интерфейса:

try:

self.load\_contacts()

except Exception as e:

messagebox.showerror(«Ошибка», f»Не удалось загрузить контакты {e}»)

self.db.close()

4. Защита от некорректных операций :

- Проверка выбранного элемента перед редактированием/удалением.

- Подтверждение деструктивных операций (удаление).

- Восстановление состояния после ошибок.

Результат оптимизации:

После внедрения обработки исключений:

- Повышена отказоустойчивость системы.

- Улучшен пользовательский опыт.

- Снижено количество критических сбоев.

- Обеспечена стабильность работы приложения.

Система теперь корректно обрабатывает:

- Отсутствие подключения к базе данных.

- Некорректный ввод пользователя.

- Попытки дублирования контактов.

- Непредвиденные ошибки выполнения.

# 2.5 Тестирование интерфейса пользователя средствами инструментальной среды разработки

Тестирование интерфейса пользователя:

Для тестирования пользовательского интерфейса использовались .

как ручные, так и автоматизированные методы.

Ручное тестирование:

1. Тестирование элементов управления:

- Проверка отклика кнопок.

- Тестирование полей ввода.

- Проверка работы таблицы контактов.

- Тестирование контекстного меню.

2. Тестирование пользовательских сценариев:

- Добавление нового контакта.

- Поиск существующего контакта.

- Редактирование данных контакта.

- Удаление контакта.

- Навигация по интерфейсу.

3. Тестирование удобства использования:

- Интуитивность интерфейса.

- Доступность функций.

- Читаемость текста.

- Соответствие ожиданиям пользователя.

Автоматизированное тестирование:

На основе сформированной диаграммы вариантов использования .

сгенерирован набор тестов:

1. Тест-кейсы для добавления контактов:

- Ввод корректных данных.

- Ввод данных с отсутствующими обязательными полями.

- Ввод дублирующего номера телефона.

- Ввод некорректного email.

2. Тест-кейсы для добавления контактов:

- Поиск по существующему имени.

- Поиск по части имени.

- Поиск по номеру телефона.

- Поиск по несуществующим данным.

3. Тест-кейсы для редактирования:

- Изменение имени контакта.

- Изменение номера телефона.

- Изменение дополнительной информации.

- Отмена редактирования.

Инструменты тестирования:

Для тестирования использовались:

- Встроенные средства Python (unittest, doctest).

- Ручное тестирование с проверкой на различных данных.

- Тестирование на разных версиях Python (3.6, 3.7, 3.8, 3.9).

- Проверка на разных операционных системах (Windows, Linux).

Результаты тестирования:

По результатам тестирования интерфейса:

- Подтверждена корректность работы всех элементов управления.

- Выявлены и исправлены 2 незначительные проблемы с отображением.

- Достигнута полная функциональность согласно требованиям.

- Обеспечена стабильность работы интерфейса.

# 2.6 Функциональное тестирование и тестирование интеграции.

Функциональное тестирование:

Проведено функциональное тестирование по методологии «черного ящика», когда тестировщик не имеет доступа к исходному коду и тестирует

только внешнее поведение системы.

Тестовые сценарии «черного ящика»:

1. Сценарий: Добавление корректного контакта

Входные данные: «Иван», «Иванов», «+79990001122», «ivan@mail.ru»

Ожидаемый результат: Контакт появляется в списке

2. Сценарий: Добавление контакта с дублирующим телефоном

Входные данные: «Петр», «Петров», «+79990001122», «petr@mail.ru»

Ожидаемый результат: Сообщение об ошибке «Контакт с таким номером уже существует».

3. Сценарий: Поиск существующего контакта

Входные данные: Поисковый запрос «Иван»

Ожидаемый результат: Отображение контакта Иванова.

4. Сценарий: Поиск несуществующего контакта

Входные данные: Поисковый запрос «Алексей»

Ожидаемый результат: Пустой список результатов.

5. Сценарий: Редактирование контакта

Входные данные: Изменение email у существующего контакта

Ожидаемый результат: Контакт обновлен с новым email.

6. Сценарий: Удаление контакта

Входные данные: Выбор контакта и подтверждение удаления

Ожидаемый результат: Контакт удален из списка.

Тестирование интеграции:

Проведено инкрементальное тестирование интеграции модулей

с использованием подхода «снизу вверх»:

1. Интеграция модуля базы данных и бизнес-логики:

- Тестирование сохранения контакта через бизнес-логику.

- Проверка загрузки контактов из базы данных.

- Тестирование поиска и фильтрации.

2. Интеграция бизнес-логики и графического интерфейса:

- Проверка отображения данных в GUI.

- Тестирование передачи команд от GUI к бизнес-логике.

- Проверка обновления интерфейса при изменении данных.

3. Полная интеграция всех модулей:

- Энд-ту-энд тестирование полных сценариев использования.

- Проверка корректности передачи данных между всеми компонентами

- Тестирование обработки ошибок в интегрированной системе.

Документирование результатов тестирования:

Результаты тестирования оформлены в виде:

- Протоколов тестирования с указанием даты, версии и результатов.

- Отчетов о найденных дефектах с приоритетами и статусами.

- Рекомендаций по улучшению системы.

- Метрик качества программного обеспечения.

По результатам тестирования подтверждено:

- Соответствие системы функциональным требованиям.

- Корректность интеграции всех модулей.

- Стабильность работы приложения.

- Готовность системы к использованию.

# 3 Моделирование в программных системах.

В рамках раздела «Моделирование в программных системах» решены задачи из дополнительного файла «Задания по разделу 3.pdf».

Решенные задача:

1. Задача линейного программирования (Задача 2):

Найти максимальное значение функции:

F = 2x₁ + x₂ - x₃ + x₄ - x₅ → max

При ограничениях:

x₁ + x₂ + x₃ = 5

2x₁ + x₂ + x₄ = 9

x₁ + 2x₂ + x₅ = 7

x₁, x₂, x₃, x₄, x₅ ≥ 0

Решение:

Для решения использован симплекс-метод с применением библиотеки

SciPy.

Максимальное значение функции F = 9.0 достигается при:

x₁ = 4.0, x₂ = 1.0, x₃ = 0.0, x₄ = 0.0, x₅ = 1.0

2. Транспортная задача (Задача 6):

Найти оптимальный план перевозок для транспортной сети:

Пункты отправления:

- A1 = 160 единиц

- A2 = 140 единиц

- A3 = 60 единиц

Пункты назначения:

- B1 = 80 единиц

- B2 = 80 единиц

- B3 = 60 единиц

- B4 = 80 единиц

Матрица стоимостей перевозок:

B1 B2 B3 B4

A1: 5 4 3 4

A2: 3 2 5 5

A3: 1 6 3 2

Решение:

Использован метод потенциалов для нахождения оптимального плана.

Минимальная стоимость перевозок: 980 условных единиц.

3. Задача теории марковских процессов (Задача 10):

Для технического устройства с тремя состояниями S0, S1, S2

и заданными интенсивностями потоков найдены финальные

вероятности.

Решение системы уравнений Колмогорова:

P0 = 0.4, P1 = 0.3, P2 = 0.3

Методы решения:

1. Симплекс метод:

- Применен для решения задач линейного программирования.

- Использована библиотека SciPy (linprog).

- Учтены ограничения в виде равенств и неравенств.

2. Метод потенциалов:

- Реализован для транспортных задач.

- Проверена сбалансированность задачи.

- Найдено начальное опорное решение.

- Проведена оптимизация плана перевозок.

3. Системы дифференциальных уравнений:

- Решена система уравнений Колмогорова для марковских процессов.

- Найдены финальные вероятности состояний.

- Проведен анализ устойчивости системы.

Использованные инструменты:

- Python 3.6+.

- Библиотеки: NumPy, SciPy.

- Алгоритмы линейной алгебры.

- Методы оптимизации.

- Теория вероятностей и математическая статистика.

Практическая значимость:

Решение задач математического моделирования позволило:

- Освоить методы оптимизации в программировании.

- Применить теоретические знания на практике.

- Разработать алгоритмы для решения прикладных задач.

- Проанализировать эффективность различных методов.

Все задачи успешно решены, что подтверждает корректность

реализованных алгоритмов и их применимость для решения

практических задач управления и оптимизации.

# Заключение

В ходе учебной практики по профессиональному модулю ПМ.02

«Осуществление интеграции программных модулей» были успешно

выполнены все поставленные задачи.

Основные результаты:

1. Разработана полнофункциональная АИС «Телефонный справочник»,

включающая графический интерфейс, модуль работы с базой данных

и бизнес-логику приложения.

2. Проведен полный цикл разработки программного обеспечения:

- Анализ предметной области.

- Проектирование архитектуры системы.

- Реализация и интеграция модулей.

- Тестирование и отладка.

- Документирование.

3. Освоены современные технологии и инструменты:

- Язык программирования Python.

- Библиотека Tkinter для создания GUI.

- Система управления базами данных SQLite.

- Система контроля версий Git.

- Методы тестирования программного обеспечения.

4. Решены задачи математического моделирования:

- Задачи линейного программирования.

- Транспортные задачи.

- Задачи теории марковских процессов.

Практическая значимость:

Разработанная АИС «Телефонный справочник» представляет собой

законченный программный продукт, готовый к практическому

применению. Система обладает удобным интерфейсом, надежной

работой с данными и может быть использована для ведения

контактной информации в различных организациях.

Перспективы развития:

В дальнейшем система может быть расширена следующими функциями:

- Экспорт/импорт контактов в различные форматы

- Синхронизация с облачными сервисами

- Группировка контактов по категориям

- Напоминания о днях рождения

- Интеграция с телефонной связью

# Выводы:

Учебная практика позволила закрепить теоретические знания,

полученные в процессе обучения, и приобрести практические

навыки разработки программного обеспечения. Все поставленные

цели и задачи практики выполнены в полном объеме.

Разработанная система соответствует современным требованиям

к программному обеспечению и демонстрирует готовность

студента к профессиональной деятельности в области информационных

технологий.

# Список использованных источников

1. Python Software Foundation. Документация по Python [Электронный ресурс]. – Режим доступа: https://docs.python.org/3/ (дата обращения: 01.10.2025).

2. Tkinter documentation [Электронный ресурс]. – Режим доступа: https://docs.python.org/3/library/tkinter.html (дата обращения: 01.10.2025).

3. SQLite documentation [Электронный ресурс]. – Режим доступа: https://www.sqlite.org/docs.html (дата обращения: 01.10.2025).

4. ГОСТ 34.602-89. Техническое задание на создание автоматизированной системы. – М.: Стандартинформ, 1989. – 25 с.

5. ГОСТ 19.701-90. Схемы алгоритмов, программ, данных и систем. Условные обозначения и правила выполнения. – М.: Стандартинформ, 1990. – 15 с.

6. ГОСТ 2.708-81. Правила выполнения электрических схем цифровой вычислительной техники. – М.: Стандартинформ, 1981. – 20 с.

7. Любанович Б. Простой Python. Современный стиль программирования. – СПб.: Питер, 2019. – 480 с.

8. Россум Г., Дрейк Ф.Л. Язык программирования Python. Справочник. – М.: Символ-Плюс, 2019. – 454 с.

9. Бизли Д. Python. Подробный справочник. – СПб.: Диалектика, 2020. – 608 с.

10. Саммерфилд М. Программирование на Python 3. Подробное руководство. – СПб.: Символ-Плюс, 2019. – 608 с.

11. Шлее М. Python. Карманный справочник. – СПб.: БХВ-Петербург, 2020. – 224 с.

12. Сузи Р.А. Язык программирования Python. – М.: ДМК Пресс, 2019. – 464 с.

13. Таненбаум Э., Уэзеролл Д. Компьютерные сети. – СПб.: Питер, 2020. – 960 с.

14. Федорова Г.Н. Разработка, внедрение и адаптация программного обеспечения отраслевой направленности: учеб. пособие. – М.: КУРС: ИНФРА-М, 2018. – 336 с.

15. Гагарина Л.Г., Кокорева Е.В., Сидорова-Виснадул Б.Д. Технология разработки программного обеспечения: учеб. пособие / под ред. Л.Г. Гагариной. – М.: ИД «Форум»: ИНФРА-М, 2018. – 400 с.

# Приложение А. Текст программы

АИС «Телефонный справочник»

# main.py

import tkinter as tk

from gui import PhoneBookGUI

from database import Database

def main():

try:

db = Database()

root = tk.Tk()

app = PhoneBookGUI(root, db)

root.mainloop()

except Exception as e:

print(f»Ошибка запуска: {e}»)

if \_\_name\_\_ == «\_\_main\_\_»:

main()# ПРИЛОЖЕНИЕ А

# main.py

import tkinter as tk

from gui import PhoneBookGUI

from database import Database

def main():

try:

db = Database()

root = tk.Tk()

app = PhoneBookGUI(root, db)

root.mainloop()

except Exception as e:

print(f»Ошибка запуска: {e}»)

if \_\_name\_\_ == «\_\_main\_\_»:

main()

# Приложение Б. Тест-кейсы и отчет о тестировании

Изображение выглядит как текст, диаграмма, линия, снимок экрана

Содержимое, созданное искусственным интеллектом, может быть неверным.

Рисунок 12 – структура приложения Б

Изображение выглядит как линия, текст, диаграмма, График

Содержимое, созданное искусственным интеллектом, может быть неверным.Рисунок 13 – Функциональные тест-кейсы

Изображение выглядит как текст, диаграмма, линия, снимок экрана

Содержимое, созданное искусственным интеллектом, может быть неверным.Рисунок 14 – Интеграционные тест-кейсы

Изображение выглядит как диаграмма, линия, График, текст

Содержимое, созданное искусственным интеллектом, может быть неверным.Рисунок 15 – Результаты тестирования