

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

**«Дальневосточный федеральный университет»**

(ДВФУ)

**ИНСТИТУТ МАТЕМАТИКИ И КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ**

**(ШКОЛА)**

**Департамент программной инженерии и искусственного интеллекта**

ТКАЧЕВ АНДРЕЙ ВЛАДИМИРОВИЧ

РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ ДЛЯ РАБОТЫ СО СПРАВОЧНИКАМИ ПРЕДМЕТНОЙ ОБЛАСТИ «ВЕТЕРИНАРНАЯ КЛИНИКА»

**КУРСОВОЙ ПРОЕКТ**

по дисциплине «Фундаментальные структуры данных и алгоритмы»

по образовательной программе подготовки бакалавров по направлению

09.03.04 «Программная инженерия», профиль «Программная инженерия»

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | |  |  | Студент группы Б9123-09.03.04 | | | | | | | |
|  |  |  | | | | Ткачев А.В. | | | |
|  | | | | | | |  |  | (подпись) | | | |  | | | |
|  | | | | | | | | | | | | | | | | |
|  | | | | | | |  |  | Руководитель | | | | | | | |
|  | | | | | | |  |  | ассистент ДПИиИИ | | | | | | | |
|  | | | | | | |  |  | ученая степень, должность | | | | |  | Решетнев Н.Я. | |
|  | | | | | | |  |  | (подпись) | | | | |  | (ФИО) | |
|  | | | | | | | | | | | | | | | | |
| Регистрационный № | | | | | | |  |  | Защищен с оценкой | | | | | | | |
|  | | | | | | |  |  |  | | | | | | | |
|  | | | |  |  | |  |  | « |  | » |  |  | | | 2025 г. |
| (подпись) | | | |  | (ФИО) | |  |  |  | | | | | | | |
| « |  | » |  | | | 2025 г. |  |  |  | | | | | | | |

г. Владивосток

2025

# Оглавление

[Оглавление 2](#_Toc203327293)

[Введение 3](#_Toc203327294)

[1 Анализ предметной области 4](#_Toc203327295)

[1.1 Объект предметной области 4](#_Toc203327296)

[1.2 Законы ПО 7](#_Toc203327297)

[1.3 Постановка задачи «Формирование отчета «Посещение питомца»» 8](#_Toc203327298)

[2 Теоретическая часть 9](#_Toc203327299)

[2.1 Хеш-таблица 9](#_Toc203327300)

[2.1.1 Хеш-функция 9](#_Toc203327301)

[2.1.2 Разрешение коллизий методом открытая адресация 10](#_Toc203327302)

[2.1.3 Указываете, какая цепочка используется в ХТ (если цепочка только в дереве, то этот пункт должен быть ниже после деревьев) **Ошибка! Закладка не определена.**](#_Toc203327303)

[2.1.4 Пример хеш-таблицы на данных из ПО 12](#_Toc203327304)

[2.2 AVL-дерево 12](#_Toc203327305)

[2.2.1 Динамический двусвязный упорядоченный по возрастанию список 13](#_Toc203327306)

[2.2.2 Пример дерева на данных из ПО 13](#_Toc203327307)

[3 Требования к информационной системе 15](#_Toc203327308)

[3.1 Требования к данным 15](#_Toc203327309)

[3.1.1 Требования к входным данным 15](#_Toc203327310)

[3.1.2 Требования к выходным данным 18](#_Toc203327311)

[3.2 Функциональные требования 19](#_Toc203327312)

[3.2.1 Общие требования 19](#_Toc203327313)

[3.2.2 Требования для работы со справочником «Питомцы» 19](#_Toc203327314)

[3.2.3 Требования для работы со справочником «Приемы» 20](#_Toc203327315)

[4 Реализация 21](#_Toc203327316)

[4.1 Спецификация структур данных 21](#_Toc203327317)

[4.2 Описание среды разработки 22](#_Toc203327318)

[4.3 Руководство пользователя 22](#_Toc203327319)

[4.3.1 Работа со справочником «Питомцы» 23](#_Toc203327320)

[4.3.2 Работа со справочником «Приемы» 25](#_Toc203327321)

[4.3.3 Формирование отчета «Приемы питомцев» 25](#_Toc203327322)

[4.3.4 Отладка 27](#_Toc203327323)

[4.3.5 О программе 28](#_Toc203327324)

[4.4 Тестирование 28](#_Toc203327325)

[Заключение 30](#_Toc203327326)

[Список литературы 31](#_Toc203327327)

# Введение

ё

Ветеринарные клиники — это специализированные учреждения, занимающиеся приемом и лечением животных, проведением вакцинаций, хирургических операций, ведением медицинских карт пациентов, а также решением задач, таких как запись на прием и учет лекарственных препаратов.

Значительную часть работы клиник составляет документооборот. Ручная обработка данных требует много времени и усилий, поэтому оптимизация этих процессов — важный шаг к повышению общей эффективности работы учреждения.

В связи с этим возникает необходимость в создании программной системы, автоматизирующей управление данными в ветеринарных клиниках. Такой подход предполагает формализацию и структурирование информации, что значительно упрощает и ускоряет работу персонала. В рамках проекта будет разработан удобный интерфейс для взаимодействия с системой, а также реализованы функции хранения данных о пациентах и приемах.

Целью курсового проекта является: разработка информационной системы для работы со справочниками предметной области «Ветеринарная клиника».

Для достижения поставленной цели необходимо выполнить следующие задачи:

1. Провести анализ предметной области «Ветеринарная клиника».

2. Изучить теоретические основы методов построения справочников.

3. Определить требования к информационной системе.

4. Реализовать информационную систему и провести тестирование.

# 1 Анализ предметной области

Требуется разработать информационную систему для предметной области (ПО) «Ветеринарная клиника».

Система должна решать следующие задачи:

1. хранить, позволять просматривать, добавлять и удалять информацию о питомцах и приёмах в клинику;
2. позволять искать информацию о питомцах и приёмах в клинику по кличке и имени владельца;
3. формировать и позволять просматривать список приёмов питомцев в клинике;
4. проверять целостность информации, представленной в справочниках.

1.1 Объект предметной области

Исходя из задач, которые должна решать информационная система, выделены объекты ПО, представленный в таблице 1.

|  |  |
| --- | --- |
| Тип и название объекта | Описание объекта |
| Справочник «Питомцы» | Хранит информацию по каждому питомцу |
| Справочник «Приемы» | Хранит информацию по каждому приему в ветеринарной клинике |
| Отчет «Посещение питомца» | Хранит информацию по каждому питомцу и посещенному им приему |

Таблица 1 – Объекты ПО

Каждый питомец характеризируется следующими параметрами: кличка, вид, владелец.

**Кличка** ̶ слово, состоящее из букв русского алфавита, первая буква ̶ заглавная, остальные ̶ строчные.

**Вид** – слово, полностью состоящее из строчных букв.

**Владелец**  ̶ строка, состоящая из нескольких слов, которые состоят из букв русского алфавита, первые буквы каждого слова ̶ заглавные, остальные ̶ строчные.

Пример справочника «Питомцы» представлен в таблице 2.

Таблица 2 – Справочник «Питомцы»›.

|  |  |  |
| --- | --- | --- |
| **Кличка** | **Вид** | **Владелец** |
| Барсик | кот | Иванов Иван Иванович |
| Мурзик | кот | Петров Петр Петрович |
| Мурка | кошка | Сидорова Анна Васильевна |
| Васька | кот | Смирнов Алексей Игоревич |
| Рыжик | кот | Иванов Иван Иванович |
| Пушок | кот | Козлова Елена Дмитриевна |
| Дымок | кот | Петров Петр Петрович |
| Снежок | кот | Кузнецов Олег Юрьевич |
| Люсик | кошка | Иванова Мария Петровна |
| Тимоша | кот | Васильев Андрей Иванович |

Каждый прием характеризируется следующими параметрами: кличка, владелец, диагноз, врача, дата приема.

**Кличка** ̶ слово, состоящее из букв русского алфавита, первая буква ̶ заглавная, остальные ̶ строчные.

**Владелец**  ̶ строка, состоящая из нескольких слов, которые состоят из букв русского алфавита, первые буквы каждого слова ̶ заглавные, остальные ̶ строчные.

**Диагноз**  ̶ строка, состоящая из одного или несколько слов. Каждое слово состоит из заглавных или прописных букв.

**Врач**  ̶ строка, состоящая из нескольких слов, которые состоят из букв русского алфавита, первые буквы каждого слова ̶ заглавные, остальные ̶ строчные.

**Дата приема** – строка, формата «DD месяц YYYY», где DD – день (целое число со значением от 1 до 31), месяц - x { “январь”, “февраль”, “март”, “апрель”, “мая”, “июнь”, “июль”, “август”, “сентябрь”, “октябрь”, “ноябрь”, “декабрь” }, YYYY – число от 2000 до 2025.

Пример справочника «Приемы» представлен в таблице 3.

Таблица 3 – Справочник «Приемы».

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Кличка** | **Владелец** | **Диагноз** | **Врач** | **Дата приема** |
| Барсик | Иванов Иван Иванович | Простуда | Соколов Артем Викторович | 15 январь 2024 |
| Барсик | Иванов Иван Иванович | Повторный осмотр | Соколов Артем Викторович | 22 январь 2024 |
| Мурзик | Петров Петр Петрович | Плановый осмотр | Михайлова Елена Сергеевна | 18 февраль 2024 |
| Мурка | Сидорова Анна Васильевна | Вакцинация | Михайлова Елена Сергеевна | 5 март 2024 |

*Продолжение таблицы 3.*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Кличка** | **Владелец** | **Диагноз** | **Врач** | **Дата приема** |
| Васька | Смирнов Алексей Игоревич | Чистка ушей | Соколов Артем Викторович | 10 март 2024 |
| Рыжик | Иванов Иван Иванович | Царапина на ухе | Михайлова Елена Сергеевна | 12 март 2024 |
| Рыжик | Иванов Иван Иванович | Снятие швов | Михайлова Елена Сергеевна | 20 март 2024 |
| Пушок | Козлова Елена Дмитриевна | Аллергия | Соколов Артем Викторович | 11 апрель 2024 |
| Дымок | Петров Петр Петрович | Проверка зубов | Соколов Артем Викторович | 30 апрель 2024 |
| Снежок | Кузнецов Олег Юрьевич | Плановый осмотр | Михайлова Елена Сергеевна | 7 май 2024 |

Каждая запись отчета характеризуется следующими полями: кличка, вид, владелец, диагноз, врача, дата приема.

**Кличка** ̶ слово, состоящее из букв русского алфавита, первая буква ̶ заглавная, остальные ̶ строчные.

**Вид** – слово, полностью состоящее из строчных букв.

**Владелец**  ̶ строка, состоящая из нескольких слов, которые состоят из букв русского алфавита, первые буквы каждого слова ̶ заглавные, остальные ̶ строчные.

**Диагноз** – строка, состоящая из одного или несколько слов. Каждое слово состоит из заглавных или прописных букв.

**Врач**  ̶ строка, состоящая из нескольких слов, которые состоят из букв русского алфавита, первые буквы каждого слова ̶ заглавные, остальные ̶ строчные.

**Дата приема** – строка, формата «DD месяц YYYY», где DD – день (целое число со значением от 1 до 31), месяц - x { “январь”, “февраль”, “март”, “апрель”, “мая”, “июнь”, “июль”, “август”, “сентябрь”, “октябрь”, “ноябрь”, “декабрь” }, YYYY – число от 2000 до 2025.

Пример отчета «Посещение питомца» представлен в таблице 4.

Таблица 4 – Отчет «Приемы питомцев».

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Кличка** | **Вид** | **Владелец** | **Диагноз** | **Врач** | **Дата приема** |
| Барсик | кот | Иванов Иван Иванович | Простуда | Соколов Артем Викторович | 15 январь 2024 |
| Барсик | кот | Иванов Иван Иванович | Повторный осмотр | Соколов Артем Викторович | 22 январь 2024 |
| Мурзик | кот | Петров Петр Петрович | Плановый осмотр | Михайлова Елена Сергеевна | 18 февраль 2024 |
| Мурка | кошка | Сидорова Анна Васильевна | Вакцинация | Михайлова Елена Сергеевна | 5 март 2024 |
| Васька | кот | Смирнов Алексей Игоревич | Чистка ушей | Соколов Артем Викторович | 10 март 2024 |
| Рыжик | кот | Иванов Иван Иванович | Царапина на ухе | Михайлова Елена Сергеевна | 12 март 2024 |
| Рыжик | кот | Иванов Иван Иванович | Снятие швов | Михайлова Елена Сергеевна | 20 март 2024 |
| Пушок | кот | Козлова Елена Дмитриевна | Аллергия | Соколов Артем Викторович | 11 апрель 2024 |
| Дымок | кот | Петров Петр Петрович | Проверка зубов | Соколов Артем Викторович | 30 апрель 2024 |
| Снежок | кот | Кузнецов Олег Юрьевич | Плановый осмотр | Михайлова Елена Сергеевна | 7 май 2024 |

1.2 Законы ПО

1. Данные о приёме не могут существовать без соответствующей записи о питомце.
2. Питомец однозначно идентифицируется парой (Кличка, Владелец).
3. Отчёт о посещениях должен содержать полную историю приёмов для выбранного питомца.
4. Дата приёма не может быть будущей относительно текущей даты системы и должна соответствовать формату
5. Каждый прием одного и того же питомца может иметь как одинаковый, так и разный диагноз, или диагноз может вообще отсутствовать.
6. Даты приёмов для одного питомца должны идти в хронологическом порядке

1.3 Постановка задачи «Формирование отчета «Посещение питомца»»

Входные данные: Справочник «Питомцы», Справочник «Приёмы», дата начала периода, дата конца периода, Владелец и диагноз.

Выходные данные: Отчет «Приёмы питомцев».

Связь:

Отчет «Приёмы питомцев».Кличка = Справочник «Питомцы».Кличка

Отчет «Приёмы питомцев».Владелец = Справочник «Питомцы».Владелец

Отчет «Приёмы питомцев».Вид = Справочник «Питомцы».Вид

Отчет «Приёмы питомцев».Диагноз = Справочник «Приёмы».Диагноз

Отчет «Приёмы питомцев».Врач = Справочник «Приёмы».Врач

Отчет «Приёмы питомцев».Дата = Справочник «Приёмы».Дата

Если существует запись в справочнике «Приёмы»:

Справочник «Приёмы».Дата приёма >= дата конца периода &&

Справочник «Приёмы».Дата приёма <= дата конца периода&&

Справочник «Приёмы».Владелец = заданный владелец &&

Справочник «Приёмы».Диагноз = Диагноз &&

Причем должна существовать запись в справочнике «Питомцы»:

Справочник «Питомцы».Кличка = Справочник «Приёмы».Кличка

Справочник «Питомцы».Владелец = Справочник «Приёмы».Владелец

# 2 Теоретическая часть

В рамках курсового проекта необходимо осуществлять поиск информации о приёмах питомцах на основании клички питомца и его владельца.

Для хранения данных о враче в Справочнике «Питомцы» будет использоваться структура данных «Хеш-таблица», исходя из того, что пара «Кличка – владелец» должны быть уникальны в справочнике, и необходимо обеспечить быстрое выполнение операций поиска, добавления, удаления.

Для хранения данных о приёме в Справочнике «Приёмы» будет использоваться структура данных «АВЛ-дерево», так как питомец может иметь несколько приёмов, и необходимо обеспечить быстрое выполнение операций поиска, добавления, удаления.

2.1 Хеш-таблица

Хеш-таблица представляет собой эффективную структуру данных для реализации словарей. Хеш-таблица обобщает обычный массив. Возможность прямой индексации элементов массива обеспечивает доступ к произвольному элементу массива за время O(1). Позволяет выполнять операции поиска, вставки и удаления[Кормен.А ].

2.1.1 Хеш-функция

Теоретический материал со ссылками на литературу, что такое ХФ [номер из списка литературы].

Теоретический материал про метод хеширования, который будете использовать.

Пример на тех записях, которые приводили в анализе ПО (подобрать такой, где значения ХФ одинаковые) -> подводите к коллизии + указываете, какие методы разрешения коллизий существуют и какой вы будете использовать.

2.1.2 Разрешение коллизий методом открытая адресация

Открытая адресация — это метод разрешения коллизий, при котором все элементы хранятся непосредственно в самой хеш-таблице. Каждая ячейка в этой хеш-таблице может находиться в одном из трех состояний:

Пустая (статус 0)

Занятая (статус 1)

Удаленная (статус 2)

При поиске элемента последовательно исследуются (пробируются) ячейки таблицы до тех пор, пока не будет найден искомый элемент или пустая ячейка.

Для вставки при открытой адресации последовательно исследуются ячейки хеш-таблицы до тех пор, пока не будет найдена пустая ячейка для размещения вставляемого ключа. Вместо фиксированного порядка исследования ячеек, последовательность исследуемых ячеек зависит от вставляемого ключа.

Особенность операции удаления в открытой адресации заключается в том, что простое опустошение слота после удаления ключа может нарушить корректность последующих операций поиска. Если ключ, который был удален, находился на пути пробирования для другого ключа (то есть, поиск другого ключа должен был пройти через этот слот), то после его удаления поиск мог бы ошибочно остановиться, посчитав, что искомый ключ отсутствует.

Для предотвращения таких ошибок, удаленные слоты специально помечаются как "удаленные". Это позволяет операции вставки использовать такой слот для размещения нового элемента. Однако, при поиске, алгоритм не останавливается на "удаленном" слоте; он продолжает пробирование так, как если бы слот был занят, пока не найдет искомый ключ или действительно пустой слот.

2.1.2 Разрешение коллизий методом открытой адресации

В хеш-таблицах с открытой адресацией пары «ключ-значение» хранятся непосредственно в основном массиве H. Когда происходит вставка нового элемента, алгоритм последовательно проверяет ячейки массива до тех пор, пока не найдёт первую свободную. Именно в неё и записывается новый элемент. Последовательность проверки ячеек вычисляется динамически, что позволяет экономить память, так как не требуются указатели, как в методе цепочек [1].

Эта последовательность, в которой просматриваются ячейки, называется последовательностью проб. Она уникальна для каждого ключа и представляет собой набор индексов h0(x), h1(x), h2(x), ..., где x — это ключ. Обычно первый элемент этой последовательности h₀(x) является результатом основной хеш-функции, а последующие вычисляются по определённым правилам. Важнейшее требование к последовательности проб — она должна гарантировать, что в конечном итоге каждая ячейка таблицы будет проверена ровно один раз [1].

Алгоритм поиска работает, следуя той же самой последовательности проб, что и при вставке. Поиск останавливается в двух случаях: либо когда найден элемент с нужным ключом, либо когда встречается пустая ячейка. Пустая ячейка означает, что искомого элемента в таблице нет [1].

Удаление элементов в такой структуре представляет определённую сложность. Простое освобождение ячейки может нарушить цепочку проб и сделать недоступными другие элементы. Для решения этой проблемы для каждой ячейки вводят специальный флаг (метку), который указывает, был ли элемент из неё удалён. Тогда процесс удаления сводится к установке этого флага. При этом логика поиска считает ячейки с флагом удаления занятыми и продолжает просмотр, а вставка считает такие ячейки свободными, может записывать в них новые данные и сбрасывать флаг [1].

2.1.4 Пример хеш-таблицы на данных из ПО

В рамках курсового проекта в хеш-таблице хранится указываете какой ключ и значение. На рисунке номер представлен пример для справочника «Название».

Рисуете пример построенной ХТ для справочников п. 1.1, как выполняется поиск, что происходит после добавления и удаления.

2.2 AVL-дерево

АВЛ-дерево, названное в честь его изобретателей, советских математиков Г. М. Адельсона-Вельского и Е. М. Ландиса, представляет собой один из первых примеров двоичного дерева поиска (БДП)., характеризующееся строгим свойством баланса: для любого узла разность высот его левого и правого поддеревьев (фактор баланса) ограничена интервалом [-1, 1]. Отклонение фактора баланса за пределы данного интервала вызывает автоматическое выполнение операций ротации для восстановления сбалансированного состояния.

В отличие от стандартных БДП, подверженных деградации производительности до O(n) в худшем случае при последовательной вставке отсортированных элементов, АВЛ-дерево поддерживает логарифмическую высоту O(log n) за счет механизма балансировки. Это гарантирует временную сложность O(log n) для всех основных операций, включая поиск, вставку и удаление, независимо от порядка поступления данных. АВЛ-дерево сохраняет при этом инварианты БДП: ключи левого поддерева меньше ключа узла, правого — больше.

Основные операции на АВЛ-дереве:

1. **Поиск:** Итеративное или рекурсивное сравнение искомого ключа с ключом текущего узла с последующим переходом в соответствующее поддерево. Временная сложность O(log n) благодаря сбалансированности.
2. **Вставка:** Новый узел добавляется как лист, аналогично стандартной вставке в БДП. После вставки осуществляется восходящий обход от вставленного узла к корню (retracing). Для несбалансированных узлов (фактор баланса > 1 или < -1) выполняются соответствующие операции ротации (левая, правая, лево-правая, право-левая) для восстановления AVL-свойства.
3. **Удаление:** Реализуется как стандартное удаление в БДП. Узел с двумя потомками заменяется своим непосредственным предшественником или преемником, который затем удаляется. После удаления производится проверка и, при необходимости, балансировка восходящим обходом от поддерева изменения к корню. В отличие от вставки, удаление может требовать каскадных ротаций на предках.

2.2.1 Динамический двусвязный упорядоченный по возрастанию список

Для списка определены операции добавления, удаления и поиска. Добавление записи осуществляется при помощи поиска места вставки с сохранением порядка возрастания. Список может содержать повторяющиеся элементы. Операции удаления и поиска выполняются путем последовательной проверки всех элементов списка для нахождения необходимого элемента по заданному ключу. Каждый узел списка, помимо данных, содержит указатели на предыдущий и следующий узел. Указатель на следующий узел хвостового элемента имеет значение NULL.

2.2.2 Пример дерева на данных из ПО

В рамках курсового проекта в дереве хранится указываете какой ключ и значение. На рисунке номер представлен пример для справочника «Название».

Рисуете пример построенной дерева для справочников п. 1.1, как выполняется поиск, что происходит после добавления и удаления.

И аналогично для дерева, которое используется при фильтрации для формирования отчета.

# 3 Требования к информационной системе

В данной главе описываются требования к информационной системе «Ветеринарная клиника», а именно: требования к входным данным системы, требования к ее выходным данным и требования к ее функционалу.

3.1 Требования к данным

3.1.1 Требования к входным данным

Основываясь на анализе ПО, входными данными является:

* текстовый файл «Pets.txt», каждая строка которого содержит информацию о питомце из Справочника «Питомцы»: Кличка, Владелец, Вид, разделенные символом «;».Разделителем внутри поля Владелец служит символ пробела;

Пример текстового файла:

Барсик;Иванов Иван Иванович;кот

Мурзик;Петров Петр Петрович;кот

Мурка;Сидорова Анна Васильевна;кошка

Васька;Смирнов Алексей Игоревич;кот

Рыжик;Иванов Иван Иванович;кот

Пушок;Козлова Елена Дмитриевна;кот

Дымок;Петров Петр Петрович;кот

Снежок;Кузнецов Олег Юрьевич;кот

Люсик;Иванова Мария Петровна;кошка

Тимоша;Васильев Андрей Иванович;кот

Зорька;Козлова Елена Дмитриевна;кошка

Алмаз;Соколов Роман Юрьевич;кот

Черныш;Волков Дмитрий Андреевич;кот

Феня;Кузнецова Светлана Николаевна;кошка

Милка;Николаева Ольга Сергеевна;кошка

Бусинка;Козлова Елена Дмитриевна;кошка

Симба;Сидорова Анна Васильевна;кот

Маркиз;Волков Дмитрий Андреевич;кот

Персик;Николаева Ольга Сергеевна;кот

Феликс;Смирнов Алексей Игоревич;кот

* текстовый файл «Visits.txt», каждая строка которого содержит информацию о приёме из Справочника «Приемы»: Кличка, Владелец, Диагноз, Врач, Дата разделенные символом «;»;

Пример текстового файла:

Барсик;Иванов Иван Иванович;Простуда;Соколов Артем Викторович;15 январь 2024

Барсик;Иванов Иван Иванович;Повторный осмотр;Соколов Артем Викторович;22 январь 2024

Мурзик;Петров Петр Петрович;Плановый осмотр;Михайлова Елена Сергеевна;18 февраль 2024

Мурка;Сидорова Анна Васильевна;Вакцинация;Михайлова Елена Сергеевна;5 март 2024

Васька;Смирнов Алексей Игоревич;Чистка ушей;Соколов Артем Викторович;10 март 2024

Рыжик;Иванов Иван Иванович;Царапина на ухе;Михайлова Елена Сергеевна;12 март 2024

Рыжик;Иванов Иван Иванович;Снятие швов;Михайлова Елена Сергеевна;20 март 2024

Пушок;Козлова Елена Дмитриевна;Аллергия;Соколов Артем Викторович;11 апрель 2024

Дымок;Петров Петр Петрович;Проверка зубов;Соколов Артем Викторович;30 апрель 2024

Снежок;Кузнецов Олег Юрьевич;Плановый осмотр;Михайлова Елена Сергеевна;7 май 2024

Люсик;Иванова Мария Петровна;Обследование;Соколов Артем Викторович;2 июнь 2024

Тимоша;Васильев Андрей Иванович;Стрижка когтей;Михайлова Елена Сергеевна;7 июнь 2024

Зорька;Козлова Елена Дмитриевна;Плановый осмотр;Соколов Артем Викторович;12 июнь 2024

Граф;Иванов Иван Иванович;Удаление клеща;Михайлова Елена Сергеевна;19 июнь 2024

Алмаз;Соколов Роман Юрьевич;Повторный визит;Соколов Артем Викторович;26 июнь 2024

Черныш;Волков Дмитрий Андреевич;Проверка зрения;Михайлова Елена Сергеевна;1 июль 2024

Феня;Кузнецова Светлана Николаевна;Отравление;Соколов Артем Викторович;5 июль 2024

Милка;Николаева Ольга Сергеевна;Плановая вакцинация;Михайлова Елена Сергеевна;10 июль 2024

Бусинка;Козлова Елена Дмитриевна;Обследование кожи;Соколов Артем Викторович;15 июль 2024

Симба;Сидорова Анна Васильевна;Повторный осмотр;Михайлова Елена Сергеевна;20 июль 2024

Маркиз;Волков Дмитрий Андреевич;Стрижка когтей;Соколов Артем Викторович;25 июль 2024

Персик;Николаева Ольга Сергеевна;Чистка зубов;Михайлова Елена Сергеевна;30 июль 2024

Феликс;Смирнов Алексей Игоревич;Аллергия;Соколов Артем Викторович;3 август 2024

Мурка;Сидорова Анна Васильевна;Повторный осмотр;Михайлова Елена Сергеевна;5 август 2024

Люсик;Иванова Мария Петровна;Плановый осмотр;Соколов Артем Викторович;10 август 2024

3.1.2 Требования к выходным данным

Выходными данными являются:

* текстовый файл «changePets.txt», каждая строка которого содержит информацию о питомце из Справочника «Питомцы»: Кличка, Владелец, Вид, разделенные символом «;». Разделителем внутри поля ФИО служит символ пробела;
* текстовый файл «changeVisits.txt», каждая строка которого содержит информацию о приёме из Справочника «Приемы»: Кличка, Владелец, Диагноз, Врач, Дата разделенные символом «;». Разделителем внутри поля ФИО служит символ пробела;
* окно графического интерфейса, содержащее информацию из отчёта «Приемы питомцев»;
* сообщения об ошибках должны выводиться в отдельной форме и содержать информацию об ошибке. Всевозможные сообщения об ошибках приведены ниже:
* «Не удалось удалить питомца. Проверьте введенные данные.»;
* «Произошла ошибка при удалении записи.»;
* «Неверный формат входных данных.»;
* «Питомец с такими данным не найден»;
* «Не удалось удалить данные о приеме. Проверьте введенные данные»;
* «Необходимо указать хотя бы один фильтр.»;
* «Пожалуйста, выберите питомца для удаления.»;
* «Пожалуйста, выберите прием для удаления.»;
* «Не удалось загрузить данные из файла. Пожалуйста, проверьте, что файл существует и имеет корректный формат.»;

3.2 Функциональные требования

3.2.1 Общие требования

Информационная система «Ветеринарная клиника» должна позволять:

1. считывать данные для справочника «Питомцы» из текстового файла «Pets.txt» в массив, который выводиться в виде таблицы в графическом интерфейсе;
2. считывать данные для справочника «Приёмы» из текстового файла «Visits.txt» в массив, который выводиться в виде таблицы в графическом интерфейсе
3. сохранять данные справочника «Питомцы» в текстовый файл «changePets.txt»;
4. сохранять данные справочника «Приемы» в текстовый файл «changeVisits.txt»
5. работать со справочниками «Питомцы» и «Приемы» при помощи графического интерфейса;
6. проверять корректность вводимых пользователем данных для справочников «Питомцы» и «Приемы» (см. п. 1.1);
7. формировать список для отчёта «Посещение питомца» и выводить в окно графического интерфейса;
8. проверять целостность данных, хранимых в справочниках.

3.2.2 Требования для работы со справочником «Питомцы»

Информационная система «Юридическая фирма» должна позволять:

* 1. хранить информацию о питомцах: Кличка, Владелец, Вид;
  2. просматривать информацию о питомцах, используя графический интерфейс информационной системы;
  3. осуществлять поиск информации о питомце на основе его клички и владельце. Поиск должен осуществляться через графический интерфейс с помощью полей для ввода данных. Если в справочнике отсутствует искомая запись, программа возвращает соответствующее сообщение. Иначе выводится сообщение о нахождении нужного клиента;
  4. добавлять информацию о питомце в справочник «Питомцы». Добавление должно осуществляться через графический интерфейс с помощью полей для ввода данных. Каждая запись уникальна, при наличии добавляемой записи в справочнике выводится сообщение об ошибке;
  5. удалять информацию о питомце из справочника «Питомцы». Удаление должно осуществляться через графический интерфейс с помощью полей для ввода данных. После удаления должен быть проверен справочник «Приемы» на наличие записей с удалённым питомцем, в случае нахождения записи должны быть удалены. Если в справочнике «Питомцы» изначально не был найден удаляемый клиент, система возвращает соответствующее сообщение;

3.2.3 Требования для работы со справочником «Приемы»

Информационная система «Юридическая фирма» должна позволять:

1. хранить информацию о посещениях: Кличка, Владелец, Диагноз, Врач, Дата;
2. просматривать информацию о посещениях, используя графический интерфейс информационной системы;
3. осуществлять поиск информации о посещениях на основе клички питомца и владельце. Поиск должен осуществляться через графический интерфейс с помощью полей для ввода данных. Если в справочнике отсутствует искомая запись, программа возвращает соответствующее сообщение. Иначе выводится сообщение о нахождении нужного клиента;
4. добавлять информацию о посещение в справочник «Приемы». Добавление должно осуществляться через графический интерфейс с помощью полей для ввода данных. Каждая запись уникальна, при наличии добавляемой записи в справочнике выводится сообщение об ошибке. Если в справочнике «Питомцы» не существует клиента, указанного в добавляемой консультации, возвращается сообщение об ошибке;
5. удалять информацию о посещение из справочника «Приёмы». Удаление должно осуществляться через графический интерфейс с помощью полей для ввода данных. Если в справочнике «Приёмы» изначально не был найден удаляемый клиент, система возвращает соответствующее сообщение;

# 4 Реализация

В данной главе содержится спецификация реализованных структур данных, описание графического интерфейса, а также приведены результаты тестирования структур данных.

4.1 Спецификация структур данных

Структура **Person** – структура для хранения ФИО и обработки вывода формата

Поля:

* string surname – фамилия;
* string name – имя;
* string secondname – отчество.

Методы:

* string toString() const – преобразует структуру в строку формата " ФамилияИмяОтчество";

Входные данные: экземпляр структуры Person;

Формальные параметры: отсутствуют;

Выходные данные: строка string в формате "ФамилияИмяОтчество".

* string toStringView() const – преобразует структуру в строку формата "Фамилия Имя Отчество";

Входные данные: экземпляр структуры Person;

Формальные параметры: отсутствуют;

Выходные данные: строка string в формате "Фамилия Имя Отчество".

* operator<, operator==, operator> – для сравнения двух объектов Person.

Структура **Date** – структура для хранения и обработки даты

Поля:

* int day – день;
* int month – месяц;
* int year – год.

Методы:

* string toString() const – преобразует структуру в строку формата "День Месяц Год";

Входные данные: экземпляр структуры Date;

Формальные параметры: отсутствуют;

Выходные данные: строка string в формате " День Месяц Год ".

* int monthToInt() – преобразует строку в соответствующий ей число, если существует такой месяц по счёту;

Входные данные: экземпляр структуры Date;

Формальные параметры: string month;

Выходные данные: число int от 0 до 11, соответствующая месяцу.

* string intToMonth(int number) – преобразует число в соответствующий ей месяц, если такой месяц существует.

Входные данные: экземпляр структуры Date;

Формальные параметры: int number;

Выходные данные: строку, соответствующую месяцу по номеру.

* operator<, operator==, operator> – для сравнения двух объектов Date.

Структура **Owner** – структура для хранения ФИО и обработки вывода его формата, наследуется от **Person** и имеет те же поля и методы.

Структура **Doctor** – структура для хранения ФИО и обработки вывода его формата, наследуется от **Person** и имеет те же поля и методы.

Структура **Key** – структура для хранения ключа, который используется в СД

Поля:

* string petName – кличка;
* Owner owner – владелец.

Методы:

* string toString() const – преобразует структуру в строку формата "КличкаФамилияИмяОтчество";

Входные данные: экземпляр структуры Key;

Формальные параметры: отсутствуют;

Выходные данные: строка string в формате "КличкаФамилияИмяОтчество".

* string toStringView() const – преобразует структуру в строку формата "Кличка Фамилия Имя Отчество";

Входные данные: экземпляр структуры Key;

Формальные параметры: отсутствуют;

Выходные данные: строка string в формате "Кличка Фамилия Имя Отчество".

* operator<, operator==, operator> – для сравнения двух объектов Key.

Структура **recordPet** – структура для хранения одной записи о питомце справочника «Питомцы».

Поля:

* Key key – пара владельца и клички питомца;
* string type – вид питомца.

Методы:

* string toString() const – преобразует структуру в строку формата "Кличка Фамилия Имя отчество Вид";

Входные данные: экземпляр структуры recordPet;

Формальные параметры: отсутствуют;

Выходные данные: строка string в формате " Кличка Фамилия Имя Отчество Вид ".

* operator<, operator==, operator> – для сравнения двух объектов recordPet.

Структура **recordVisit** – структура для хранения одной записи о приеме справочника «Приёмы»

Поля:

* Key key – пара владельца и клички питомца;
* string diagnos – диагноз при посещение приема;
* Doctor doctor – доктор, проводивший прием;
* Date date – дата приема.

Методы:

* string toString() const – преобразует структуру в строку формата "Кличка Фамилия Имя Отчество Диагноз Фамилия Имя Отчество День Месяц Год";

Входные данные: экземпляр структуры recordVisit;

Формальные параметры: отсутствуют;

Выходные данные: строка string в формате "Кличка Фамилия Имя Отчество Диагноз Фамилия Имя Отчество День Месяц Год".

* operator<, operator==, operator> – для сравнения двух объектов recordVisit.

Структура **ReportEntry** – структура для хранения одной записи сформированного отчета «Приёмы питомцев»

Поля:

* string petName – кличка;
* string owner – владелец;
* string petType – вид питомца;
* string diagnos – диагноз при посещение приема;
* string doctor – доктор, проводивший прием;
* string date – дата приема.

Методы: Отсутствуют.

Структура **ListNode**<dataType> – структура для хранения узла связного списка

Поля:

* ListNode\* prev – указатель на предыдущую структуру списка;
* dataType data – значение узла списка;
* ListNode\* next – указатель на следующую структуру списка.

Методы: Отсутствуют.

Класс **DoubleLinkedList**<dataType> – класс, описывающий двухсвязный список, хранящий произвольный тип данных

Поля:

* Внутренние указатели на голову списка состоящий из структур ListNode.

Методы:

* operator= - оператор копирования одного объекта DoubleLinkedList<datatype>;
* void copy(DoubleLinkedList& other) – процедура для копирование списка в другой;

Входные данные: экземпляр структуры DoubleLinkedList<dataType>;

Формальные параметры: DoubleLinkedList<dataType> other ;

Выходные данные: DoubleLinkedList<dataType>.

* void add(dataType value) – процедура для добавление значения с список;

Входные данные: экземпляр структуры DoubleLinkedList<dataType>;

Формальные параметры: dataType value;

Выходные данные: DoubleLinkedList<dataType>;

* bool isEmpty() – функция проверки является ли список пустым (true – пустой, false – содержит элементы);

Входные данные: экземпляр структуры DoubleLinkedList<dataType>;

Формальные параметры: отсутствуют;

Выходные данные: bool.

* void clearList() – процедура освобождения выделенной памяти на спиок;

Входные данные: экземпляр структуры DoubleLinkedList<dataType>;

Формальные параметры: отсутствуют;

Выходные данные: отсутствуют.

* optional<dataType> deleteNodeByValue( datatype n) – функция удаления элемента по значению и возвращает значение если оно есть;

Входные данные: экземпляр структуры DoubleLinkedList<dataType>;

Формальные параметры: dataType n;

Выходные данные: dataType n / null.

* ListNode\* getHead( ) – функция, которая возвращает указатель на голову списка;

Входные данные: экземпляр структуры DoubleLinkedList<dataType>;

Формальные параметры: отсутствует;

Выходные данные: ListNode\* head.

* void clearList() – процедура, выводящая значения всех узлов списка;

Входные данные: экземпляр структуры DoubleLinkedList<dataType>;

Формальные параметры: отсутствуют;

Выходные данные: отсутствуют.

* string toString() – функция, возвращающая строковое значения всех узлов списка.

Входные данные: экземпляр структуры DoubleLinkedList<dataType>;

Формальные параметры: отсутствуют;

Выходные данные: string.

4.2 Описание среды разработки

Для реализации информационной системы «Юридическая фирма» была выбрана среда разработки Visual Studio Сode. Язык реализации C++, а также использовался фреймворк Qt6 для создания графического интерфейса

4.3 Руководство пользователя

Изображение выглядит как снимок экрана, Мультимедийное программное обеспечение, текст, программное обеспечение

Содержимое, созданное искусственным интеллектом, может быть неверным.При запуске исполняемого файла приложения my\_vet\_app.exe открывается главное окно с заголовком «Ветеринарная клиника», представляющее собой основной интерфейс информационной системы. Стартовое окно содержит две таблицы: «Питомцы» (слева) и «Приемы» (справа), предназначенные для просмотра и управления данными (см. Рисунок n).

Рисунок n – Главное окно информационной системы

4.3.1 Работа со справочником «Питомцы»

В левой части окна, расположена таблица, отображающая содержимое справочника «Питомцы» (Кличка, Владелец, Тип). При добавлении записей или загрузке данных из файла, вся информация о питомцах, хранящаяся в памяти, выводится в эту таблицу.

При нажатии на пункт меню «Правка» → «Добавить» → «Добавить питомца» открывается диалоговое окно для добавления новой записи в справочник «Питомцы». На Рисунке n+1 представлено окно добавления записи.

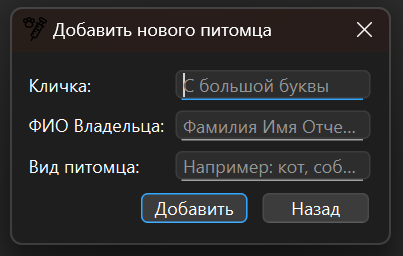


Рисунок n+1 – Окно «Добавить клиента» справочника «Клиенты»

В текстовые поля «Кличка», «ФИО Владелец» и «Вид питомца» пользователю необходимо ввести соответствующие данные и затем нажать кнопку «Добавить». При некорректности формата введенных данных или попытке добавить питомца с существующими данными, всплывающее окно уведомит пользователя об этом. В случае успешного добавления таблица обновится с новыми данными.

При нажатии на пункт меню «Правка» и последующем выборе в выпадающем меню пункта выбрать «Удалить» → «Удалить питомца» → «Удалить по данным» открывается диалоговое окно для удаления записи из справочника «Питомцы». На Рисунке n+2 представлено окно удаления записи.

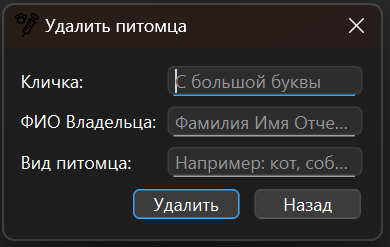


Рисунок n+2 – Окно «Удаление записи» справочника «Клиенты»

В текстовые поля «Кличка», «ФИО Владельцы» и «Вид» необходимо ввести данные питомца, которого пользователь хочет удалить, и нажать кнопку «Удалить». Система запросит дополнительное подтверждение удаления, так как при удалении клиента будут также удалены все связанные с ним консультации. О некорректности данных, отсутствии записи в справочнике или несовпадении данных пользователя уведомят соответствующие всплывающие окна.

Также доступно удаление выделенного питомца, необходимо выбрать запись из справочника «Питомцы» путем клика по записи и выбрать в меню «Удалить» → «Удалить питомца» → «Удалить выделенного».

При нажатии на пункт меню «Правка» и последующем выборе в выпадающем меню пункта «Найти питомца» открывается диалоговое окно для ввода клички питомца и ФИО владельца. На Рисунке n+3 представлено окно поиска питомца.

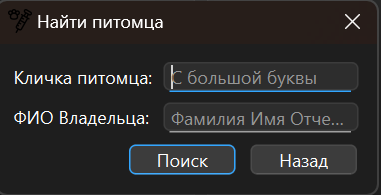


Рисунок n+3 – Окно «Найти питомца»

В текстовое поле необходимо ввести кличку питомца и ФИО владельца. Если кличка питомца или ФИО владельца введены в неверном формате, система уведомит об этом. В случае нахождения питомца будет выведено информационное сообщение с его данными. Если питомец не найден, появится сообщение «Питомец с такими данными не найден».

Для работы с файлами, в меню «Файл» доступны пункты «Загрузить» → «Питомцев из файла» и «Сохранить» → «Питомцев в файл» При выборе этих пунктов открываются стандартные диалоговые окна для выбора файла загрузки или указания имени файла для сохранения. В случае успешной операции будет выведено соответствующее информационное сообщение. При ошибках загрузки или сохранения (например, файл не найден, неверный формат данных, отсутствие прав на запись) пользователь будет уведомлен.

4.3.2 Работа со справочником «Приемы»

4.3.3 Формирование отчета «Приемы питомцев»

При нажатии на пункт меню «Отчет» и последующем выборе в выпадающем меню пункта «Сформировать отчет» открывается диалоговое окно для настройки параметров отчета «Приемы питомцев». На Рисунке K представлено окно определения фильтром отчета.

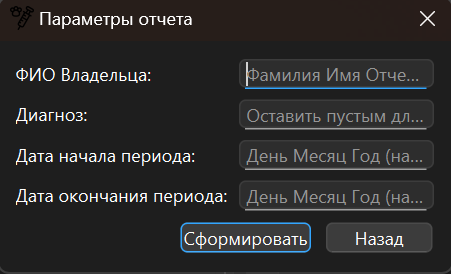


Рисунок n – Окно определения параметров зачета.

В полях «ФИО Владельца», «Диагноз», «Дата начала периода» и «Дата начала периода» пользователь может ввести критерии для фильтрации. Поля могут быть оставлены пустыми, если фильтрация по ним не требуется. При нажатии кнопки «Сформировать», открывается новое окно с представлением отчета в табличном виде. На рисунке K+1 представлено окно со сформированным отчетом.

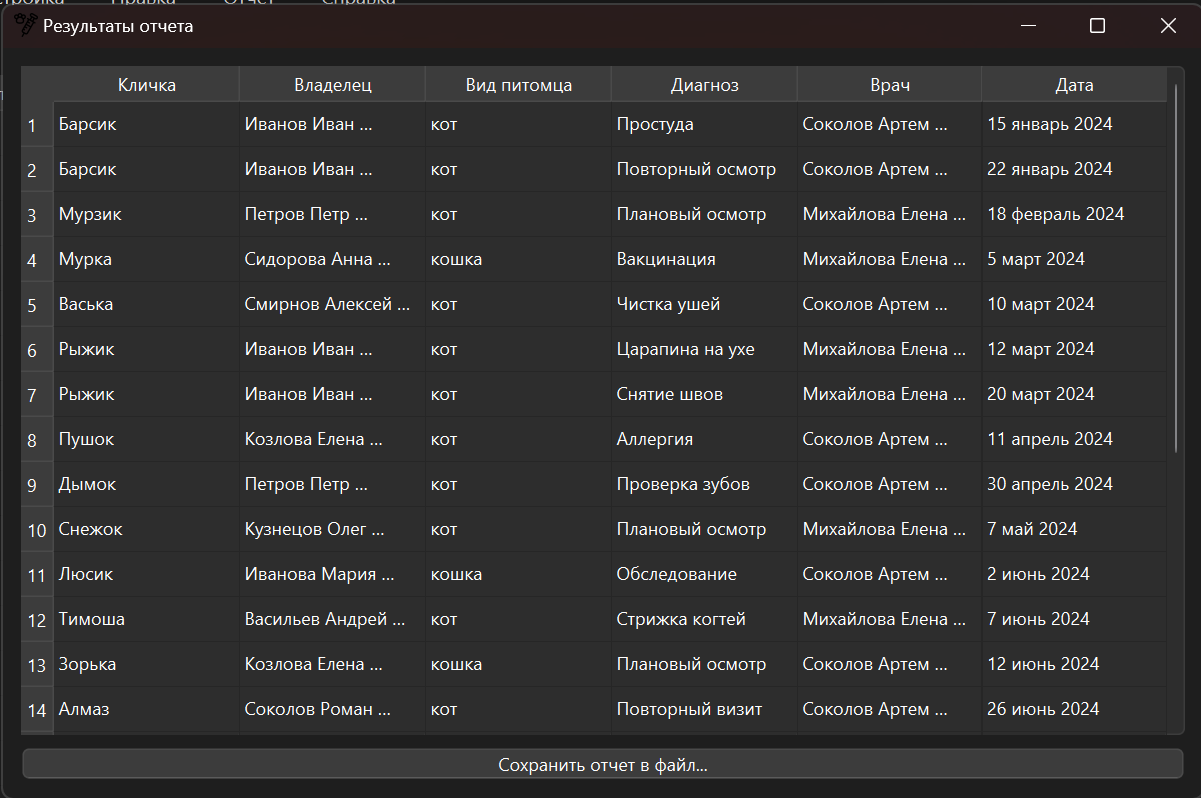


Рисунок n – Окно с представлением отчета .

Для сохранения отчета необходимо нажать кнопку «Сохранить отчет в файл…». Далее открывается стандартные диалоговые окна для выбора пути для сохранения файла. В случае успешной операции будет выведено соответствующее информационное сообщение. При ошибках загрузки или сохранения (например, файл не найден, неверный формат данных, отсутствие прав на запись) пользователь будет уведомлен.

4.3.4 Отладка

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Содержимое, созданное искусственным интеллектом, может быть неверным.При нажатии на пункт меню «Справка» и последующем выборе в выпадающем меню пункта «Отладка» открывается новое диалоговое окно «Окно отладки» с двумя вкладками: «Хэш-таблицы (Питомцы)», «АВЛ-дерево (Приемы)». На рисунке n представлено окно отладки c вкладкой «Хэш-таблицы (Питомцы)».

Рисунок n – Окно «Отладка» вкладка «Хэш-таблицы (Питомцы)».

На рисунке n представлено окно отладки c вкладкой «АВЛ-дерево (Приемы)».

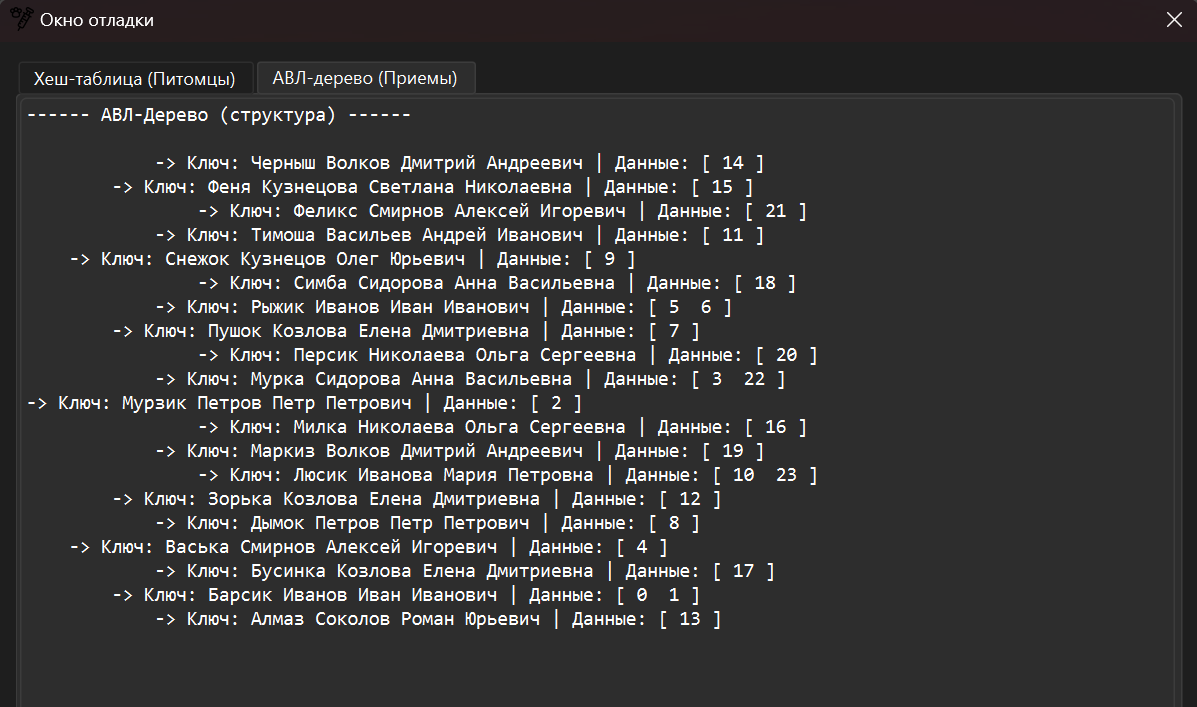


Рисунок n + 1 – Окно «Отладка» вкладка «АВЛ-дерево (Приемы)».

4.3.5 О программе

При нажатии на пункт меню «Справка» и последующем выборе в выпадающем меню пункта «О программе» открывается стандартное информационное окно с краткими сведениями о приложении.

4.4 Тестирование

Тестируете отдельно работу СД и отдельно работу со справочником.

Тестирование проводилось методом черного ящика, результаты которого представлены в Таблицах \_\_\_\_\_.

Таблица \_\_\_\_ – Тестирование работы со справочником «Название»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Описание тестовой ситуации | | Входные данные | | Выходные данные | |
| Ваш справочник | Ваши поля | Ваш справочник | Сообщение |
| Добавление | | | | | |
| 1 | Название тестовой ситуации |  |  |  |  |
| 2 | и т.д. |  |  |  |  |
| Удаление | | | | | |
| 3 | Название тестовой ситуации |  |  |  |  |
| 4 | и т.д. |  |  |  |  |
| Поиск | | | | | |
| 5 | Название тестовой ситуации |  |  |  |  |
| 6 | и т.д. |  |  |  |  |

Таблица \_\_\_\_ – Тестирование структуры данных \_\_\_\_\_\_

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Описание тестовой ситуации | | Входные данные | | Выходные данные | |
| Ваша структура | Ваши поля | Ваша структура | Ваши поля |
| Добавление | | | | | |
| 1 | Название тестовой ситуации |  |  |  |  |
| 2 | и т.д. |  |  |  |  |
| Удаление | | | | | |
| 3 | Название тестовой ситуации |  |  |  |  |
| 4 | и т.д. |  |  |  |  |
| Поиск | | | | | |
| 5 | Название тестовой ситуации |  |  |  |  |
| 6 | и т.д. |  |  |  |  |

Аналогично для второго справочника и для отчета.

# Заключение

Целью курсового проекта было: разработать информационную систему для работы со справочниками предметной области «Ветеринарная клиника».

Цель достигнута. Для достижения поставленной цели были выполнены следующие задачи:

1. Проведен анализ предметной области «Ветеринарная клиника»;
2. Изучен теоретические основы методов построения справочников;
3. Определены требования к информационной системе;
4. Реализована информационная система и проведено тестирование;
   1. Был изучен язык C++ версии 17;
   2. Изучена система для построения клиентских приложений Qt Creator и Qt6;
   3. Во время разработки была использована система контроля версий Git

# Список литературы

должны быть ссылки на учебники (в том числе интернет-источники), из которых брали информацию теоретической части