**ОГЛАВЛЕНИЕ**

[ВВЕДЕНИЕ 3](#_Toc165050057)

[ГЛАВА 1. ОБЗОР ПРЕДМЕТНОЙ ОБЛАСТИ И ОБЩАЯ ПОСТАНОВКА ЗАДАЧ 5](#_Toc165050058)

[1.1 История и развитие гостиничного бизнеса 5](#_Toc165050059)

[1.2 Организационная структура гостиницы 8](#_Toc165050060)

[1.3 Функции, выполняемые администратором гостиницы 12](#_Toc165050061)

[1.4 Постановка задачи 15](#_Toc165050062)

[ГЛАВА 2. ПРАКТИЧЕСКАЯ РЕАЛИЗАЦИЯ ИНФОРМАЦИОННОЙ СИСТЕМЫ 17](#_Toc165050063)

[2.1 Требования к программе 17](#_Toc165050064)

[2.2 Руководство пользователя 18](#_Toc165050065)

[2.3 Руководство программиста 27](#_Toc165050066)

[ЗАКЛЮЧЕНИЕ 31](#_Toc165050067)

[СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 32](#_Toc165050068)

[ПРИЛОЖЕНИЕ А 33](#_Toc165050069)

# ВВЕДЕНИЕ

В современных условиях цифровизации и автоматизации процессов информационные системы играют ключевую роль в управлении различными видами деятельности. Организация выставок собак — одно из направлений, где возникает необходимость в систематизации большого объёма данных, связанных с регистрацией участников, распределением экспертов по рингам, оценкой результатов и формированием итогов мероприятия. Проведение таких мероприятий вручную связано с многочисленными трудностями: высокой вероятностью ошибок, затратами времени на обработку информации, сложностью хранения и поиска данных. Всё это требует внедрения специализированного программного обеспечения, способного эффективно решать подобные задачи.

Тема курсовой работы — разработка программной системы, предназначенной для организаторов выставки собак. Данная система должна обеспечивать автоматизированный учет данных о собаках, экспертах, рингах, а также управление информацией о клубах и результатах выставки. Программный продукт должен быть простым в использовании, обеспечивать высокую скорость работы с данными, возможность быстрого поиска, сортировки, фильтрации и внесения изменений. Это позволит организаторам сосредоточиться на проведении мероприятия, доверив рутинные задачи информационной системе.

Актуальность работы заключается в том, что в условиях проведения крупных и массовых выставок ручное ведение записей становится крайне неэффективным. Возникают сложности с координацией участников, отслеживанием данных об экспертах, контролем за соблюдением правил и своевременным подведением итогов. Разработка программной системы, способной автоматизировать перечисленные процессы, повысит уровень организации, снизит нагрузку на персонал, обеспечит прозрачность и точность обработки информации.

Целью настоящей курсовой работы является создание программной системы, которая обеспечит:

-хранение сведений о собаках-участниках и их владельцах;

-учет информации об экспертах, их специализации и клубной принадлежности;

-распределение пород по рингам и экспертов по рингам;

-возможность управления списками участников и экспертов;

-определение медалистов и подведение итогов выставки.

Для достижения поставленной цели необходимо решить следующие задачи:

1)определить основные сущности предметной области и связи между ними;

2)разработать структуру хранения данных (модели: собака, эксперт, ринг);

3)реализовать интерфейс пользователя для ввода, редактирования и просмотра данных;

4)реализовать функции поиска и фильтрации информации по заданным критериям;

5)обеспечить возможность изменения состава участников и экспертов;

6)сформировать отчеты по итогам выставки, включая медалистов и показатели клубов.

Объектом исследования является процесс организации и проведения выставки собак, а предметом — информационные технологии, применяемые для автоматизации данного процесса.

Методы, используемые в работе, включают анализ предметной области, моделирование данных, проектирование архитектуры программной системы и реализацию на основе выбранного языка программирования и среды разработки.

Практическая значимость разработки заключается в возможности её применения в реальных условиях при проведении выставок собак различного уровня. Система может быть использована как отдельными организаторами, так и специализированными клубами и федерациями, благодаря своей гибкости и расширяемости. Таким образом, разработка программной системы для организаторов выставки собак является важной задачей, решающей широкий спектр проблем, связанных с организацией мероприятий, и способствующей повышению качества и прозрачности судейства, а также удобству взаимодействия между участниками, экспертами и организаторами.

# ГЛАВА 1. ОБЗОР ПРЕДМЕТНОЙ ОБЛАСТИ И ОБЩАЯ ПОСТАНОВКА ЗАДАЧ

## 1.1 История и развитие бизнеса выставки собак

Выставки собак являются одним из самых популярных и престижных мероприятий в мире кинологии. Их история насчитывает более полутора столетий и отражает развитие собаководства как науки, искусства и бизнеса. Поначалу выставки носили скорее показательный характер, а сегодня превратились в сложные организационные процессы с активным участием различных клубов, заводчиков, экспертов и коммерческих структур.

Истоки выставок собак относятся к середине XIX века. Первая официально задокументированная выставка прошла в 1859 году в Англии, в городе Ньюкасл-апон-Тайн. В то время мероприятие было узкоспециализированным и посвящалось охотничьим породам — сеттерам и пойнтерам. Основная цель заключалась в демонстрации рабочих качеств и экстерьера собак для повышения племенных стандартов. Эта выставка положила начало развитию кинологических клубов, появлению официальных стандартов и системы оценки собак.

В 1873 году в Великобритании был основан Британский кеннел-клуб (The Kennel Club), который стал первой организацией, занимающейся регулированием выставок и стандартизацией пород. Именно этот клуб начал создавать реестр племенных собак, разрабатывать стандарты пород и утверждать систему судейства. Подобные организации вскоре появились и в других странах Европы — Германии, Франции, а также в России. В Санкт-Петербурге первая крупная выставка собак прошла в 1874 году, и с тех пор кинология в России стала стремительно развиваться.

В XX веке собаководство перестало быть исключительно хобби для аристократии и превратилось в массовое явление. Появились новые породы, расширился круг участников выставок. В Советском Союзе кинологические мероприятия стали важной частью государственной системы, особенно в период послевоенного восстановления, когда активно развивалось служебное собаководство. Выставки служили не только для демонстрации пород, но и для оценки рабочих качеств, служили инструментом улучшения племенного фонда.

С конца XX века и по настоящее время выставки собак развиваются в двух взаимосвязанных направлениях — культурно-спортивном и коммерческом. Международные организации, такие как Fédération Cynologique Internationale (FCI), American Kennel Club (AKC), United Kennel Club (UKC), устанавливают единые правила проведения мероприятий, стандарты оценки собак и требования к экспертам. Крупнейшие выставки — Crufts в Великобритании, Westminster Kennel Club Dog Show в США, Евразия в России — собирают тысячи участников и зрителей, становясь не только спортивными соревнованиями, но и масштабными культурными событиями.

Современный бизнес, связанный с выставками собак, включает в себя широкий спектр услуг и направлений. Важным элементом является разведение и продажа племенных собак, что требует тщательного отбора и подготовки животных, их регистрации, оформления родословных и медицинских документов. Владельцы инвестируют значительные средства в груминг, тренировку, участие в мероприятиях, чтобы повысить престиж питомника и стоимость щенков.

Организаторы выставок работают с большим объемом информации — учет участников, распределение собак по классам и рингам, назначение экспертов, контроль результатов, публикация рейтингов и итогов. В условиях растущего числа участников и усложняющейся регламентации ручное ведение таких процессов становится неэффективным, что стимулирует развитие специализированных программных решений для автоматизации.

Эксперты — важнейший элемент выставок. Они проходят тщательный отбор, обладают высокой квалификацией и специализируются по породам. Их задача — объективно оценить экстерьер, поведение и соответствие собак установленным стандартам. Экспертные комиссии работают в тесной связке с организаторами, что требует точного расписания, распределения ролей и учета пожеланий участников.

Собаководческие клубы играют роль не только в подготовке и поддержке участников, но и в развитии бизнеса вокруг выставок. Клубы предоставляют площадки, поддерживают связь с владельцами и экспертами, участвуют в формировании правил и стандартов. Их деятельность способствует развитию кинологической культуры, повышению качества племенного материала и популяризации пород.

Коммерческая составляющая бизнеса на выставках собак включает оплату участия, спонсорство, продажу оборудования и товаров для животных, услуги ветеринаров, фотосъемку, рекламу и маркетинг. Современные технологии позволяют организовывать онлайн-трансляции, электронную регистрацию, создание электронных каталогов и баз данных, что значительно расширяет аудиторию и повышает качество обслуживания.

Помимо коммерческой выгоды, выставки собак имеют социальное значение. Они способствуют формированию у общества культуры ответственного отношения к животным, популяризируют идеи зоозащиты и гуманного содержания, способствуют развитию науки о поведении и здоровье собак.

В настоящее время рынок кинологических выставок характеризуется высокой конкуренцией, что требует постоянного совершенствования организации и внедрения инновационных решений. Информационные системы, позволяющие автоматизировать учет, управление и анализ данных, становятся необходимым инструментом для успешной деятельности организаторов. Это повышает прозрачность и объективность судейства, облегчает взаимодействие всех участников и способствует развитию отрасли в целом.

Таким образом, история и развитие бизнеса выставок собак отражают эволюцию кинологии от простого демонстрационного мероприятия до сложного и многофункционального процесса, включающего спортивные, культурные и коммерческие аспекты. Современные требования к организации выставок диктуют необходимость внедрения современных технологий, что делает актуальной задачу создания программных систем для поддержки деятельности организаторов выставок.

## 1.2 Организационная структура выставки собак

Организация выставки собак представляет собой сложный и многоуровневый процесс, который требует слаженной работы большого количества специалистов и участников. Для успешного проведения мероприятия необходимо четко выстроить организационную структуру, определить роли и обязанности каждого звена, а также обеспечить эффективное взаимодействие между всеми участниками выставочного процесса. Это позволяет создать условия для прозрачного судейства, комфортного пребывания участников и посетителей, а также гарантирует соблюдение всех стандартов и правил, принятых в кинологической сфере.

Ключевыми участниками выставки собак являются:

1. **Организаторы** — лица или группа лиц, ответственные за планирование, координацию и контроль всего процесса проведения выставки. Они разрабатывают регламент мероприятия, определяют место и время проведения, обеспечивают финансовую и юридическую поддержку, привлекают спонсоров и партнеров. Организаторы также занимаются вопросами регистрации участников, распределением собак по классам и рингам, а также обеспечением логистики и технической поддержки.
2. **Клубы и кинологические объединения** — организации, играющие важную роль в проведении выставок. Клубы выступают как соорганизаторы и предоставляют площадки для проведения мероприятий. Кроме того, клубы обеспечивают связь между заводчиками и экспертами, способствуют популяризации пород, поддерживают профессиональное сообщество и участвуют в формировании правил и стандартов.
3. **Эксперты (судьи)** — квалифицированные специалисты, обладающие глубокими знаниями стандартов пород и опытом оценки собак. Эксперты назначаются на определённые ринги, специализируясь на конкретных породах. Их задача — объективно оценить экстерьер, поведение и соответствие собак стандартам, выставить оценки и определить победителей. На одном ринге может работать несколько экспертов, что повышает точность судейства.
4. **Участники выставки** — заводчики и владельцы собак, регистрирующие питомцев для участия в соревнованиях. Они отвечают за подготовку собак, соответствие требованиям выставки и соблюдение правил поведения.
5. **Административный и технический персонал** — специалисты, обеспечивающие организационно-техническую поддержку выставки. Секретариат, регистраторы, волонтеры, ветеринары, специалисты по грумингу и обслуживанию площадок поддерживают порядок и комфорт на мероприятии.

Для эффективного управления выставкой собак создаётся иерархическая структура с распределением ролей и обязанностей:

1. **Главный организатор (руководитель выставки)** — отвечает за стратегическое планирование, ключевые решения и координацию всех этапов подготовки и проведения мероприятия. Взаимодействует с внешними организациями, спонсорами и кинологическими объединениями.
2. **Заместители и координаторы** — курируют отдельные направления деятельности: работу с участниками, взаимодействие с экспертами, техническое обеспечение, PR и маркетинг, финансовый учет. Обеспечивают оперативное управление и контроль.
3. **Комитет экспертов** — состоит из опытных и авторитетных судей. Отвечает за отбор судей, утверждение критериев оценивания, контроль качества судейства и решение спорных вопросов.
4. **Секретариат** — административное подразделение, занимающееся приемом заявок, обработкой документов, ведением реестров участников и экспертов, подготовкой итоговой документации.
5. **Технические группы** — отвечают за подготовку и оснащение площадок, монтаж оборудования, обеспечение безопасности и обслуживание инфраструктуры.

Эффективное взаимодействие между всеми участниками и службами — ключевой фактор успеха выставки:

* Организаторы обеспечивают прозрачный обмен информацией, своевременную коммуникацию и оперативное решение проблем.
* Эксперты сотрудничают с секретариатом и техническими службами для получения информации, материалов и оборудования.
* Административный персонал поддерживает порядок, помогает с регистрацией, предоставляет информацию и организует расписание.
* Медицинский и ветеринарный персонал контролируют здоровье собак и санитарные нормы, обеспечивая безопасность на выставке.

Ринги — специализированные площадки для оценки собак, каждый из которых имеет специализацию по породам. В одном мероприятии может быть несколько рингов, на каждом последовательно проходят показы различных пород. Это требует четкого расписания и координации.

Распределение экспертов по рингам учитывает их специализацию и опыт. Для бесперебойной работы выставки предусматриваются резервные эксперты, готовые заменить коллег при необходимости.

С развитием технологий все больше процессов автоматизируется:

-Используются программные системы для регистрации участников, формирования расписаний, распределения собак и экспертов, ведения протоколов и статистики.

-Автоматизация снижает человеческий фактор, уменьшает вероятность ошибок и повышает качество обслуживания.

-Электронные базы данных и цифровые протоколы обеспечивают прозрачность судейства и удобство анализа результатов.

-Современные информационные технологии расширяют возможности коммуникации, позволяя оперативно информировать участников, публиковать результаты онлайн, проводить трансляции и рекламные кампании.

Организационная структура выставки собак — сложный механизм, требующий четкой координации и взаимодействия множества специалистов. Хорошо выстроенная структура обеспечивает высокое качество проведения мероприятий, удовлетворяет потребности участников и зрителей, способствует развитию кинологической отрасли.

В современных условиях роль информационных технологий возрастает, делая автоматизацию организационных процессов необходимым условием успешной деятельности организаторов, что повышает уровень сервиса и объективность оценки.

## 1.3 Функции, выполняемые администратором выставки собак

Администратор выставки собак играет ключевую роль в организации и успешном проведении мероприятия. Его деятельность охватывает широкий спектр задач, начиная с этапа подготовки и заканчивая подведением итогов, обеспечивая эффективное взаимодействие всех участников и служб. Без слаженной работы администратора проведение выставки было бы невозможно, так как именно он отвечает за координацию процессов и контроль выполнения регламента.

История появления должности администратора связана с усложнением структуры и масштабов выставок собак. В начале организационные задачи выполнялись непосредственно владельцами клубов или главными организаторами, однако с ростом числа участников и усложнением регламентов возникла необходимость в выделении отдельной должности — администратора, который специализируется на управлении всеми организационными процессами выставки. Со временем роль администратора расширялась, и теперь она включает в себя не только организационные функции, но и взаимодействие с техническими службами, спонсорами, а также контроль за соблюдением правил и нормативов.

Сегодня обязанности администратора охватывают все этапы выставки — от планирования до подведения итогов. Ниже перечислены основные функции, которые выполняет администратор на современном этапе организации выставок собак.

1. Разрабатывает и реализует план проведения выставки, определяя основные этапы подготовки и согласовывая график работы с руководителями клубов, экспертами и техническими службами. Это позволяет обеспечить четкое и слаженное выполнение всех этапов мероприятия, минимизировать риски и оперативно реагировать на изменения.

2. Организует процесс регистрации участников — собак и экспертов, ведет реестры и проверяет документы. В его обязанности входит контроль правильности заполнения заявок, своевременной оплаты регистрационных взносов, а также внесение всей необходимой информации в электронные базы данных. Такая систематизация обеспечивает прозрачность и удобство управления информацией.

3. Координирует работу экспертов и распределяет их по рингам согласно специализации. Администратор следит за тем, чтобы каждый ринг был укомплектован квалифицированными экспертами, обладающими опытом работы с конкретными породами собак, что обеспечивает объективность и качество судейства.

4. Контролирует подготовку площадок и техническое обеспечение выставки, включая монтаж оборудования, организацию звукового сопровождения, освещения и работу электронных табло. Он взаимодействует с техническими специалистами и подрядчиками, гарантируя, что все системы работают без сбоев и соответствуют требованиям безопасности.

5. Ведет документацию по итогам мероприятия — протоколы судейства, сертификаты, дипломы и отчеты. Этот пункт является важной частью административной работы, так как обеспечивает официальное оформление результатов выставки и позволяет сохранить всю информацию для последующего анализа и отчетности.

6. Оперативно решает возникающие организационные вопросы, включая замену экспертов и отстранение участников, а также другие внештатные ситуации. Благодаря этому обеспечивается бесперебойное проведение мероприятия и соблюдение его регламента.

В современной практике различают несколько разновидностей администраторов, каждый из которых отвечает за отдельный блок задач и сферу ответственности. Такое разделение обязанностей позволяет повысить эффективность работы всей команды и обеспечить качественное проведение выставки.

1. Администратор регистрации — специализируется на приеме заявок участников, проверке документов, ведении базы данных и сопровождении процесса регистрации. Он обеспечивает, чтобы все данные были корректно оформлены и вовремя внесены в систему.

2. Технический администратор — отвечает за подготовку площадок, организацию технического оснащения выставки. Он контролирует монтаж оборудования, настройку звуковых и световых систем, работу электронных табло и других технических средств.

3. Администратор рингов — координирует работу судей и экспертов на местах проведения выставок, следит за соблюдением регламента и расписания. Его задача — обеспечить комфортные условия для проведения состязаний и своевременное выполнение всех этапов.

4. Администратор по работе с участниками — является связующим звеном между организаторами и участниками. Он решает возникающие вопросы, консультирует участников, оказывает помощь при необходимости, организует информирование и обратную связь.

5. Администратор по безопасности — контролирует порядок и соблюдение правил безопасности на территории выставки. Он отвечает за санитарные нормы, охрану, а также за профилактику и оперативное реагирование на возможные чрезвычайные ситуации.

6. Финансовый администратор — управляет финансовыми аспектами выставки, включая учет оплаты регистрационных взносов, работу со спонсорами и бухгалтерией, контроль бюджета мероприятия. Его задача — обеспечить прозрачность и своевременность финансовых операций.

Таким образом, роль администратора выставки собак комплексна и многоаспектна. Его деятельность направлена на обеспечение высокого качества организации и проведения выставок, создание комфортных условий для участников и гостей, а также поддержание репутации клуба и уровня мероприятия. Хорошо слаженная работа администрации является залогом успеха любой выставки.

## 1.4 Постановка задачи

Общая задача данной курсовой работы заключается в разработке информационной системы для администратора выставки собак, которая будет обеспечивать следующие основные функции:

1. Учёт собак-участников выставки с такими данными, как: кличка, порода, возраст, клуб, родословная, дата последней прививки, ФИО и паспортные данные владельца.
2. Учёт экспертов: ФИО, специализация по породе, номер ринга, клуб, в котором состоит эксперт.
3. Учёт рингов: название, адрес, специализация по породам, закреплённые эксперты и собаки.
4. Управление выставочными мероприятиями: регистрация новых участников и экспертов, замена судей, отстранение собак от участия.
5. Поиск, сортировка, фильтрация информации, а также удаление данных из системы при необходимости.

Задачи, решаемые в рамках работы, включают:

1. Проведение анализа потребностей организаторов выставок собак и формализация требований к программной системе с учётом специфики предметной области.
2. Выбор оптимальных технологий разработки, включая язык программирования, среду разработки и средства хранения данных.
3. Реализация программного кода, обеспечивающего основные функции системы: регистрацию собак, учёт экспертов, управление рингами, проведение операций по отстранению/замене и другие административные действия.
4. Проведение тестирования системы на корректность работы, проверка устойчивости при вводе некорректных данных и обеспечение удобного пользовательского интерфейса.
5. Оценка эффективности разработанной системы на основе сценариев использования, сбор отзывов, формирование предложений по дальнейшему расширению функционала.

Результатом выполнения курсовой работы станет полноценная информационная система для администрирования выставки собак. Система позволит автоматизировать процесс управления участниками и экспертами, упростить организацию рингов и повысить оперативность в работе организаторов, способствуя более качественному и прозрачному проведению выставочных мероприятий.

# ГЛАВА 2. ПРАКТИЧЕСКАЯ РЕАЛИЗАЦИЯ ИНФОРМАЦИОННОЙ СИСТЕМЫ

## Требования к программе

Для корректного функционирования программы необходимо как минимум 120 Мб свободного места на жестком диске. Операционная система должна быть не младше Microsoft Windows 7, включая Windows 8 и Windows 10, поддерживающая 64-битные версии. Программа должна стабильно работать при длительной эксплуатации компьютеров. В случае сбоев в работе аппаратной части необходимо обеспечить возможность восстановления функциональности программы без потери данных, в том числе и их автоматическое резервное копирование.

Разработка программного продукта осуществлялась с использованием следующей конфигурации аппаратного и программного обеспечения:

* Процессор: AMD 3020е с тактовой частотой до 2.6 ГГц.
* Оперативная память: 8 ГБ LPDDR4.
* Видеокарта: Интегрированная AMD Radeon Graphics Vega 3.
* Операционная система: Windows 10 Pro 64-битная версия.
* Накопитель: SSD на 256 ГБ.
* Дисплей: 14-дюймовый Full HD (1920 x 1080).

Эта конфигурация обеспечила разработчикам необходимую производительность для эффективной разработки, тестирования и дебаггинга программного обеспечения.

Минимально необходимая конфигурация для использования программы:

* Операционная система: Windows 7, 8, 10 (64-битные версии);
* Процессор: Intel Core 2 Duo или аналогичный с частотой от 1.8 ГГц;
* ОЗУ: не менее 4 ГБ;
* Свободное место на жестком диске: не менее 1 ГБ;
* Минимальное разрешение экрана: 1024 x 768 пикселей.

Такие требования к аппаратной и программной части обеспечивают комфортное и бесперебойное использование программы, а также позволяют максимально использовать её функциональные возможности.

## 2.2 Руководство пользователя

Руководство пользователю по использованию консольного приложения информационной системы по управлению гостиницей предназначено для ознакомления пользователя с функциональными возможностями приложения, поможет научится использовать его.

Руководство пользователя обеспечивает полную информативность по структуре интерфейса приложения, описывает все его доступные функции.

Запуск приложения

Чтобы запустить приложение, нужно дважды кликнуть левой кнопкой мыши (ЛКМ) по исполняемому файлу — файл с расширением .exe. Данное действие показано на рисунке 1.

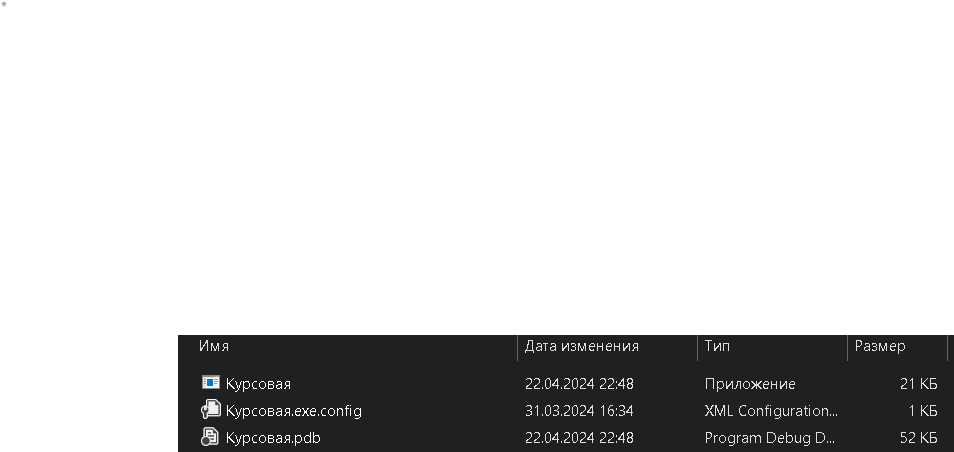


Рис. 1 Запуск приложения

Первое меню

После запуска приложения откроется первое меню, его можно увидеть на рисунке 2, на нем располагается функционал для переключения, реализованный при помощи клавиш. Нажав на клавишу на клавиатуре возможен переход на действие, которое необходимо организатору выставки собак.

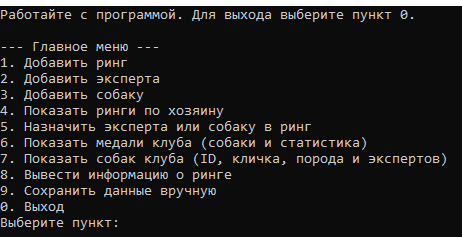


Рис. 2 Первое меню программы

Добавить ринг

Если организатору выставки собак необходимо добавить новый ринг в базу данных ему необходимо перейти на первый пункт и ввести все необходимые данные, данное действие показано на рисунке 3.

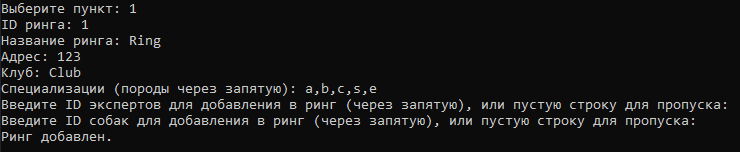


Рис. 3 Успешное добавление ринга

Добавление эксперта

Если организатору необходимо добавить нового эксперта в базу данных ему нужно перейти на второй пункт и ввести все необходимые данные, данное действие показано на рисунке 4.

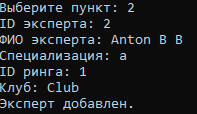


Рис. 4 Добавление эксперта

Добавление собаки и хозяина

После перехода на этот пункт организатор может добавить новую собаку в файл. Данное действие показано на рисунке 5.

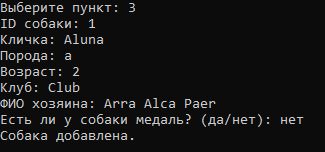


Рис. 5 Добавление собаки и хозяина

Показ ринга по хозяину

После перехода на этот пункт организатор может посмотреть по ФИО хозяина, ринг в котором он добавлен. Данное действие показано на рисунке 6.

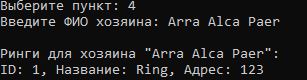


Рис. 6 Показ ринга по хозяину

Назначения эксперта или собаки в ринг

После перехода на этот пункт организатор может назначить эксперта или собаку в ринг. Данное действие показано на рисунке 7.

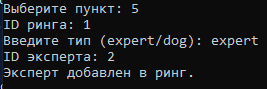


Рис. 7 Назначения эксперта или собаки в ринг

Показать медали клуба

После перехода на этот пункт организатор может посмотреть медали клуба (если есть). Данное действие показано на рисунке 8.

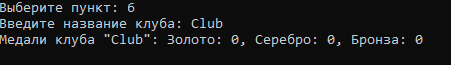


Рис. 8 Показывает медали клуба

Показывает собак клуба и эксперта

После перехода на этот пункт организатор может вывести информацию о клубе. Данное действие показано на рисунке 9.

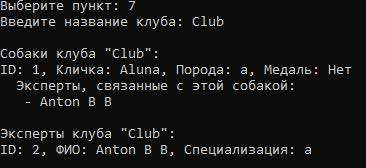


Рис. 9 Показывает собак клуба и эксперта

Вывод информации о ринге

После перехода на этот пункт организатор может вывести информацию о ринге. Данное действие показано на рисунке 10.

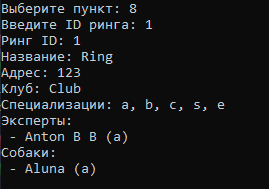


Рис. 10 Выводит информацию о ринге

Сохранение данных вручную

После перехода на этот пункт организатор может сохранить данные вручную. Данное действие показано на рисунке 11.



Рис. 11 Сохранение данных вручную

Выход из программы

После перехода на этот пункт организатор может выйти из программы. Данное действие показано на рисунке 12.

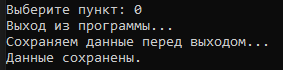


Рис. 12 Выход из программы

## Руководство программиста

Представленный программный продукт представляет собой консольное приложение. В разработке программного продукта реализовано множество алгоритмов.

Структура кода включает в себя:

1. Класс Dog — описывает собаку, участвующую в выставке. Содержит следующие свойства:

• Name (тип string) — имя собаки;

• Breed (тип string) — порода;

• Age (тип int) — возраст;

• OwnerName (тип string) — имя владельца;

• Score (тип double) — полученный балл на выставке.

1. Класс ShowOrganizer — основной класс, отвечающий за организацию выставки:

• Поле List<Dog> dogs — коллекция всех собак, зарегистрированных на выставке.

• Методы:

– public void LoadData() — метод считывает информацию из XML-файла и восстанавливает список собак.

– public void SaveData() — сохраняет текущий список собак в XML-файл.

– public void AddDog(Dog dog) — добавляет объект класса Dog в коллекцию.

– public void RemoveDog(string name) — удаляет собаку по имени.

– public void PrintDogs() — выводит список всех зарегистрированных собак.

– public Dog FindWinner() — возвращает собаку с наивысшим баллом.

1. Класс Program — точка входа приложения:

• В методе Main() создаётся объект ShowOrganizer, загружаются данные с помощью LoadData(), обрабатываются действия пользователя, данные сохраняются при завершении с помощью SaveData().

Также обработчик события AppDomain.CurrentDomain.ProcessExit отвечает за корректное сохранение данных при завершении программы.

# ЗАКЛЮЧЕНИЕ

В результате разработки данного программного продукта создано функциональное консольное приложение, обеспечивающее удобное управление информацией о выставке собак. В ходе работы реализованы основные алгоритмы для добавления, удаления, сохранения и отображения данных, что позволяет эффективно работать с коллекцией объектов.

Архитектура программы построена с применением принципов объектно-ориентированного программирования, что обеспечивает расширяемость и удобство сопровождения кода. Использование сериализации и десериализации через XML-файлы гарантирует сохранность данных между сессиями работы приложения.

В дальнейшем возможно расширение функционала за счёт реализации дополнительных модулей, например, статистики по участникам, улучшения пользовательского интерфейса, а также интеграции с базами данных для более масштабного хранения информации.

Таким образом, программный продукт удовлетворяет поставленным требованиям и служит основой для дальнейшего развития в направлении автоматизации процессов, связанных с организацией выставок собак.

# СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

· Троелсен Э., Джепикс Ф. — **C# 8.0 и .NET Core 3.0. Современное программирование на платформе .NET**. — СПб.: Питер, 2020.

· · Абрамян М.Э. — **Платформа .NET. Технология LINQ. Учебные задания**. — М.: ДМК Пресс, 2014.

· · Шилдт Г. — **Полный справочник по C#**. — М.: Вильямс, 2021.

· · Microsoft Docs — **Документация по C# и .NET**.  
URL: [https://learn.microsoft.com/ru-ru/dotnet/csharp/](https://learn.microsoft.com/ru-ru/dotnet/csharp/" \t "_new)

· · Stack Overflow — **Форум программистов по вопросам разработки на C#**.  
URL: [https://stackoverflow.com/questions/tagged/c%23](https://stackoverflow.com/questions/tagged/c%23" \t "_new)

· · Методы сериализации XML в C# (официальная справка) —  
URL: [https://learn.microsoft.com/ru-ru/dotnet/standard/serialization/xml-serialization](https://learn.microsoft.com/ru-ru/dotnet/standard/serialization/xml-serialization" \t "_new)

# ПРИЛОЖЕНИЕ А

using System;

using System.Collections.Generic;

using System.Linq;

namespace ConsoleApp1

{

public class Dog

{

public int Id { get; set; }

public string Name { get; set; }

public string Breed { get; set; }

public int Age { get; set; }

public string Club { get; set; }

public string OwnerFullName { get; set; }

public string PedigreeDocNumber { get; set; }

public string FatherName { get; set; }

public string MotherName { get; set; }

public DateTime LastVaccinationDate { get; set; }

public string Medal { get; set; } // Золото / Серебро / Бронза / null

}

// Класс "Эксперт"

public class Expert

{

public int Id { get; set; }

public string FullName { get; set; }

public string Specialization { get; set; }

public int RingId { get; set; }

public string Club { get; set; }

}

// Класс "Ринг"

public class Ring

{

public int Id { get; set; }

public string Title { get; set; }

public string Address { get; set; }

public List<string> Specializations { get; set; } = new List<string>();

public string Club { get; set; }

public List<int> ExpertIds { get; set; } = new List<int>();

public List<int> DogIds { get; set; } = new List<int>();

}

// Основной класс управления

public class ShowOrganizer

{

public List<Dog> Dogs { get; set; } = new List<Dog>();

public List<Expert> Experts { get; set; } = new List<Expert>();

public List<Ring> Rings { get; set; } = new List<Ring>();

// Добавить новую собаку

public void AddDog(Dog dog)

{

Dogs.Add(dog);

}

// Добавить нового эксперта

public void AddExpert(Expert expert)

{

Experts.Add(expert);

}

// Удалить эксперта и заменить

public void ReplaceExpert(int oldExpertId, Expert newExpert)

{

var expert = Experts.FirstOrDefault(e => e.Id == oldExpertId);

if (expert != null)

{

Experts.Remove(expert);

AddExpert(newExpert);

}

}

// Отстранить собаку

public void RemoveDog(int dogId)

{

var dog = Dogs.FirstOrDefault(d => d.Id == dogId);

if (dog != null)

{

Dogs.Remove(dog);

foreach (var ring in Rings)

{

ring.DogIds.Remove(dogId);

}

}

}

// 🔎 Запросы:

// На каком ринге выступает заданный хозяин

public List<Ring> GetRingsByOwner(string ownerFullName)

{

var dogIds = Dogs.Where(d => d.OwnerFullName == ownerFullName).Select(d => d.Id).ToList();

return Rings.Where(r => r.DogIds.Intersect(dogIds).Any()).ToList();

}

// Породы, представленные клубом

public List<string> GetBreedsByClub(string club)

{

return Dogs.Where(d => d.Club == club).Select(d => d.Breed).Distinct().ToList();

}

// Сколько медалей у клуба

public Dictionary<string, int> GetMedalsByClub(string club)

{

return Dogs

.Where(d => d.Club == club && !string.IsNullOrEmpty(d.Medal))

.GroupBy(d => d.Medal)

.ToDictionary(g => g.Key, g => g.Count());

}

// Эксперты по породе

public List<Expert> GetExpertsByBreed(string breed)

{

return Experts.Where(e => e.Specialization == breed).ToList();

}

// Специализации рингов

public Dictionary<string, List<string>> GetRingSpecializations()

{

return Rings.ToDictionary(r => r.Title, r => r.Specializations);

}

// Рекордсмены по породам

public Dictionary<string, Dog> GetTopDogsByBreed()

{

return Dogs

.Where(d => !string.IsNullOrEmpty(d.Medal))

.GroupBy(d => d.Breed)

.ToDictionary(

g => g.Key,

g => g.OrderByDescending(d => d.Medal == "Золото" ? 3 : d.Medal == "Серебро" ? 2 : 1).First()

);

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Xml.Serialization;

using System.Text;

using System.IO;

namespace DogShow

{

public class Program

{

static ShowOrganizer organizer;

static void Main(string[] args)

{

organizer = new ShowOrganizer();

organizer.LoadData();

organizer.SaveData();

AppDomain.CurrentDomain.ProcessExit += OnProcessExit;

Console.WriteLine("Работайте с программой. Для выхода выберите пункт 0.");

while (true)

{

Console.WriteLine("\n--- Главное меню ---");

Console.WriteLine("1. Добавить ринг");

Console.WriteLine("2. Добавить эксперта");

Console.WriteLine("3. Добавить собаку");

Console.WriteLine("4. Показать ринги по хозяину");

Console.WriteLine("5. Назначить эксперта или собаку в ринг");

Console.WriteLine("6. Показать медали клуба (собаки и статистика)");

Console.WriteLine("7. Показать собак клуба (ID, кличка, порода и экспертов)");

Console.WriteLine("8. Вывести информацию о ринге");

Console.WriteLine("9. Сохранить данные вручную");

Console.WriteLine("0. Выход");

Console.Write("Выберите пункт: ");

ConsoleKeyInfo key = Console.ReadKey();

Console.WriteLine(); // для переноса строки после ввода

string input = key.KeyChar.ToString();

switch (input)

{

case "1": AddRingMenu(organizer); break;

case "2": AddExpertMenu(organizer); break;

case "3": AddDogMenu(organizer); break;

case "4": ShowRingsByOwnerMenu(organizer); break;

case "5": AssignToRingMenu(organizer); break;

case "6": ShowClubMedalsMenu(organizer); break;

case "7": ShowClubDogsMenu(organizer); break;

case "8": ShowRingInfoMenu(organizer); break;

case "9":

organizer.SaveData();

Console.WriteLine("Данные сохранены вручную (по пункту 9).");

break;

case "0":

Console.WriteLine("Выход из программы...");

organizer.SaveData();

return;

}

}

}

private static void OnProcessExit(object sender, EventArgs e)

{

Console.WriteLine("Сохраняем данные перед выходом...");

organizer.SaveData();

Console.WriteLine("Данные сохранены.");

}

static void AddRingMenu(ShowOrganizer organizer)

{

Console.Write("ID ринга: ");

int id = int.Parse(Console.ReadLine());

Console.Write("Название ринга: ");

string title = Console.ReadLine();

Console.Write("Адрес: ");

string address = Console.ReadLine();

Console.Write("Клуб: ");

string club = Console.ReadLine();

Console.Write("Специализации (породы через запятую): ");

List<string> specializations = Console.ReadLine().Split(',').Select(p => p.Trim()).ToList();

var ring = new Ring

{

Id = id,

Title = title,

Address = address,

Club = club,

Specializations = specializations,

ExpertIds = new List<int>(),

DogIds = new List<int>()

};

Console.Write("Введите ID экспертов для добавления в ринг (через запятую), или пустую строку для пропуска: ");

string expertsInput = Console.ReadLine();

if (!string.IsNullOrWhiteSpace(expertsInput))

{

var expertIds = expertsInput.Split(',').Select(s => s.Trim()).Where(s => int.TryParse(s, out \_)).Select(int.Parse).ToList();

foreach (var expertId in expertIds)

{

var expert = organizer.Experts.FirstOrDefault(e => e.Id == expertId);

if (expert != null)

{

ring.ExpertIds.Add(expertId);

}

else

{

Console.WriteLine($"Эксперт с ID {expertId} не найден и не добавлен.");

}

}

}

Console.Write("Введите ID собак для добавления в ринг (через запятую), или пустую строку для пропуска: ");

string dogsInput = Console.ReadLine();

if (!string.IsNullOrWhiteSpace(dogsInput))

{

var dogIds = dogsInput.Split(',').Select(s => s.Trim()).Where(s => int.TryParse(s, out \_)).Select(int.Parse).ToList();

foreach (var dogId in dogIds)

{

var dog = organizer.Dogs.FirstOrDefault(d => d.Id == dogId);

if (dog != null)

{

if (ring.Specializations.Contains(dog.Breed))

{

ring.DogIds.Add(dogId);

}

else

{

Console.WriteLine($"Собака с ID {dogId} имеет породу {dog.Breed}, которая не подходит рингу.");

}

}

else

{

Console.WriteLine($"Собака с ID {dogId} не найдена и не добавлена.");

}

}

}

organizer.Rings.Add(ring);

Console.WriteLine("Ринг добавлен.");

}

static void AddExpertMenu(ShowOrganizer organizer)

{

Console.Write("ID эксперта: ");

int id = int.Parse(Console.ReadLine());

Console.Write("ФИО эксперта: ");

string name = Console.ReadLine();

Console.Write("Специализация: ");

string spec = Console.ReadLine();

Console.Write("ID ринга: ");

int ringId = int.Parse(Console.ReadLine());

Console.Write("Клуб: ");

string club = Console.ReadLine();

organizer.Experts.Add(new Expert { Id = id, FullName = name, Specialization = spec, RingId = ringId, Club = club });

Console.WriteLine("Эксперт добавлен.");

}

static void AddDogMenu(ShowOrganizer organizer)

{

Console.Write("ID собаки: ");

int id = int.Parse(Console.ReadLine());

Console.Write("Кличка: ");

string name = Console.ReadLine();

Console.Write("Порода: ");

string breed = Console.ReadLine();

Console.Write("Возраст: ");

int age = int.Parse(Console.ReadLine());

Console.Write("Клуб: ");

string club = Console.ReadLine();

Console.Write("ФИО хозяина: ");

string owner = Console.ReadLine();

Console.Write("Есть ли у собаки медаль? (да/нет): ");

string hasMedal = Console.ReadLine().Trim().ToLower();

string medal = "";

if (hasMedal == "да")

{

Console.Write("Введите тип медали (Золото/Серебро/Бронза): ");

medal = Console.ReadLine().Trim();

}

var dog = new Dog

{

Id = id,

Name = name,

Breed = breed,

Age = age,

Club = club,

OwnerFullName = owner,

Medal = medal

};

organizer.Dogs.Add(dog);

if (string.IsNullOrWhiteSpace(organizer.Club))

{

organizer.Club = dog.Club;

}

Console.WriteLine("Собака добавлена.");

}

static void AssignToRingMenu(ShowOrganizer organizer)

{

Console.Write("ID ринга: ");

int ringId = int.Parse(Console.ReadLine());

var ring = organizer.Rings.FirstOrDefault(r => r.Id == ringId);

if (ring == null)

{

Console.WriteLine("Ринг не найден.");

return;

}

Console.Write("Введите тип (expert/dog): ");

string type = Console.ReadLine();

if (type == "expert")

{

Console.Write("ID эксперта: ");

int expertId = int.Parse(Console.ReadLine());

if (!ring.ExpertIds.Contains(expertId))

{

ring.ExpertIds.Add(expertId);

Console.WriteLine("Эксперт добавлен в ринг.");

}

else

{

Console.WriteLine("Эксперт уже добавлен в ринг.");

}

}

else if (type == "dog")

{

Console.Write("ID собаки: ");

int dogId = int.Parse(Console.ReadLine());

var dog = organizer.Dogs.FirstOrDefault(d => d.Id == dogId);

if (dog == null || !ring.Specializations.Contains(dog.Breed))

{

Console.WriteLine("Собака не найдена или её порода не подходит рингу.");

return;

}

if (!ring.DogIds.Contains(dogId))

{

ring.DogIds.Add(dogId);

Console.WriteLine("Собака добавлена в ринг.");

}

else

{

Console.WriteLine("Собака уже добавлена в ринг.");

}

}

else

{

Console.WriteLine("Неверный тип.");

}

}

static void ShowClubDogsMenu(ShowOrganizer organizer)

{

Console.Write("Введите название клуба: ");

var club = Console.ReadLine();

var dogs = organizer.Dogs.Where(d => d.Club.Equals(club, StringComparison.OrdinalIgnoreCase)).ToList();

var experts = organizer.Experts.Where(e => e.Club.Equals(club, StringComparison.OrdinalIgnoreCase)).ToList();

Console.WriteLine($"\nСобаки клуба \"{club}\":");

foreach (var dog in dogs)

{

Console.WriteLine($"ID: {dog.Id}, Кличка: {dog.Name}, Порода: {dog.Breed}, Медаль: {(string.IsNullOrEmpty(dog.Medal) ? "Нет" : dog.Medal)}");

var dogExperts = organizer.Experts.Where(e => organizer.Rings.Any(r => r.ExpertIds.Contains(e.Id) && r.DogIds.Contains(dog.Id))).ToList();

if (dogExperts.Any())

{

Console.WriteLine(" Эксперты, связанные с этой собакой:");

foreach (var expert in dogExperts)

Console.WriteLine($" - {expert.FullName}");

}

}

Console.WriteLine($"\nЭксперты клуба \"{club}\":");

foreach (var expert in experts)

{

Console.WriteLine($"ID: {expert.Id}, ФИО: {expert.FullName}, Специализация: {expert.Specialization}");

}

}

static void ShowClubMedalsMenu(ShowOrganizer organizer)

{

Console.Write("Введите название клуба: ");

var club = Console.ReadLine();

var dogs = organizer.Dogs.Where(d => d.Club.Equals(club, StringComparison.OrdinalIgnoreCase) && !string.IsNullOrEmpty(d.Medal)).ToList();

var goldCount = dogs.Count(d => d.Medal.Equals("Золото", StringComparison.OrdinalIgnoreCase));

var silverCount = dogs.Count(d => d.Medal.Equals("Серебро", StringComparison.OrdinalIgnoreCase));

var bronzeCount = dogs.Count(d => d.Medal.Equals("Бронза", StringComparison.OrdinalIgnoreCase));

Console.WriteLine($"Медали клуба \"{club}\": Золото: {goldCount}, Серебро: {silverCount}, Бронза: {bronzeCount}");

}

static void ShowRingsByOwnerMenu(ShowOrganizer organizer)

{

Console.Write("Введите ФИО хозяина: ");

var owner = Console.ReadLine();

var rings = organizer.Rings.Where(r => r.DogIds.Any(dogId =>

{

var dog = organizer.Dogs.FirstOrDefault(d => d.Id == dogId);

return dog != null && dog.OwnerFullName.Equals(owner, StringComparison.OrdinalIgnoreCase);

})).ToList();

Console.WriteLine($"\nРинги для хозяина \"{owner}\":");

foreach (var ring in rings)

{

Console.WriteLine($"ID: {ring.Id}, Название: {ring.Title}, Адрес: {ring.Address}");

}

}

static void ShowRingInfoMenu(ShowOrganizer organizer)

{

Console.Write("Введите ID ринга: ");

int id = int.Parse(Console.ReadLine());

var ring = organizer.Rings.FirstOrDefault(r => r.Id == id);

if (ring == null)

{

Console.WriteLine("Ринг не найден.");

return;

}

Console.WriteLine($"Ринг ID: {ring.Id}");

Console.WriteLine($"Название: {ring.Title}");

Console.WriteLine($"Адрес: {ring.Address}");

Console.WriteLine($"Клуб: {ring.Club}");

Console.WriteLine("Специализации: " + string.Join(", ", ring.Specializations));

Console.WriteLine("Эксперты:");

foreach (var expertId in ring.ExpertIds)

{

var expert = organizer.Experts.FirstOrDefault(e => e.Id == expertId);

if (expert != null)

{

Console.WriteLine($" - {expert.FullName} ({expert.Specialization})");

}

}

Console.WriteLine("Собаки:");

foreach (var dogId in ring.DogIds)

{

var dog = organizer.Dogs.FirstOrDefault(d => d.Id == dogId);

if (dog != null)

{

Console.WriteLine($" - {dog.Name} ({dog.Breed})");

}

}

}

public class ShowOrganizer

{

public string Club { get; set; } = "";

public List<Ring> Rings { get; set; } = new List<Ring>();

public List<Expert> Experts { get; set; } = new List<Expert>();

public List<Dog> Dogs { get; set; } = new List<Dog>();

private const string ClubFile = @"D:\DI\club.txt";

private const string RingsFile = @"D:\DI\rings.txt";

private const string ExpertsFile = @"D:\DI\experts.txt";

private const string DogsFile = @"D:\DI\dogs.txt";

public void AddDog(Dog dog)

{

Dogs.Add(dog);

// Если в ShowOrganizer.Club ещё пусто, установим клуб из собаки

if (string.IsNullOrWhiteSpace(this.Club) && !string.IsNullOrWhiteSpace(dog.Club))

{

this.Club = dog.Club;

}

SaveData();

}

public void SaveData()

{

SaveClub();

SaveRings();

SaveExperts();

SaveDogs();

}

public void LoadData()

{

LoadClub();

LoadRings();

LoadExperts();

LoadDogs();

}

private void SaveClub()

{

StringBuilder sb = new StringBuilder();

sb.AppendLine($"Клуб: {Club}");

sb.AppendLine();

var dogsInClub = Dogs.Where(d => d.Club.Equals(Club, StringComparison.OrdinalIgnoreCase)).ToList();

sb.AppendLine("Собаки клуба:");

if (dogsInClub.Count == 0)

{

sb.AppendLine("Собак нет.");

}

else

{

foreach (var dog in dogsInClub)

{

sb.AppendLine($"ID: {dog.Id}, Кличка: {dog.Name}, Порода: {dog.Breed}, Возраст: {dog.Age}, Владелец: {dog.OwnerFullName}, Медаль: {dog.Medal}");

}

}

sb.AppendLine();

var expertsInClub = Experts.Where(e => e.Club.Equals(Club, StringComparison.OrdinalIgnoreCase)).ToList();

sb.AppendLine("Эксперты клуба:");

if (expertsInClub.Count == 0)

{

sb.AppendLine("Экспертов нет.");

}

else

{

foreach (var expert in expertsInClub)

{

sb.AppendLine($"ID: {expert.Id}, ФИО: {expert.FullName}, Специализация: {expert.Specialization}");

}

}

File.WriteAllText(ClubFile, sb.ToString(), Encoding.UTF8);

}

private void LoadClub()

{

if (!File.Exists(ClubFile))

{

Club = "";

return;

}

var lines = File.ReadAllLines(ClubFile, Encoding.UTF8);

foreach (var line in lines)

{

if (line.StartsWith("Клуб: "))

{

Club = line.Substring("Клуб: ".Length).Trim();

return;

}

}

// Если не нашли строку с названием клуба

Club = "";

}

// --- Далее без изменений, как и раньше ---

private void SaveRings()

{

StringBuilder sb = new StringBuilder();

foreach (var ring in Rings)

{

sb.AppendLine($"{ring.Id}|{ring.Title}|{ring.Address}|{ring.Club}|{string.Join(",", ring.Specializations)}|{string.Join(",", ring.ExpertIds)}|{string.Join(",", ring.DogIds)}");

}

File.WriteAllText(RingsFile, sb.ToString(), Encoding.UTF8);

}

private void LoadRings()

{

Rings.Clear();

if (!File.Exists(RingsFile)) return;

var lines = File.ReadAllLines(RingsFile, Encoding.UTF8);

foreach (var line in lines)

{

if (string.IsNullOrWhiteSpace(line)) continue;

var parts = line.Split('|');

var ring = new Ring

{

Id = int.Parse(parts[0]),

Title = parts[1],

Address = parts[2],

Club = parts[3],

Specializations = parts[4].Split(new[] { ',' }, StringSplitOptions.RemoveEmptyEntries).Select(s => s.Trim()).ToList(),

ExpertIds = parts[5].Split(new[] { ',' }, StringSplitOptions.RemoveEmptyEntries).Select(int.Parse).ToList(),

DogIds = parts[6].Split(new[] { ',' }, StringSplitOptions.RemoveEmptyEntries).Select(int.Parse).ToList()

};

Rings.Add(ring);

}

}

private void SaveExperts()

{

StringBuilder sb = new StringBuilder();

foreach (var expert in Experts)

{

sb.AppendLine($"{expert.Id}|{expert.FullName}|{expert.Specialization}|{expert.RingId}|{expert.Club}");

}

File.WriteAllText(ExpertsFile, sb.ToString(), Encoding.UTF8);

}

private void LoadExperts()

{

Experts.Clear();

if (!File.Exists(ExpertsFile)) return;

var lines = File.ReadAllLines(ExpertsFile, Encoding.UTF8);

foreach (var line in lines)

{

if (string.IsNullOrWhiteSpace(line)) continue;

var parts = line.Split('|');

var expert = new Expert

{

Id = int.Parse(parts[0]),

FullName = parts[1],

Specialization = parts[2],

RingId = int.Parse(parts[3]),

Club = parts[4]

};

Experts.Add(expert);

}

}

private void SaveDogs()

{

StringBuilder sb = new StringBuilder();

foreach (var dog in Dogs)

{

sb.AppendLine($"{dog.Id}|{dog.Name}|{dog.Breed}|{dog.Age}|{dog.Club}|{dog.OwnerFullName}|{dog.Medal}");

}

File.WriteAllText(DogsFile, sb.ToString(), Encoding.UTF8);

}

private void LoadDogs()

{

Dogs.Clear();

if (!File.Exists(DogsFile)) return;

var lines = File.ReadAllLines(DogsFile, Encoding.UTF8);

foreach (var line in lines)

{

if (string.IsNullOrWhiteSpace(line)) continue;

var parts = line.Split('|');

var dog = new Dog

{

Id = int.Parse(parts[0]),

Name = parts[1],

Breed = parts[2],

Age = int.Parse(parts[3]),

Club = parts[4],

OwnerFullName = parts[5],

Medal = parts[6]

};

Dogs.Add(dog);

}

}

}

public class Ring

{

public int Id { get; set; }

public string Title { get; set; }

public string Address { get; set; }

public string Club { get; set; }

public List<string> Specializations { get; set; }

public List<int> ExpertIds { get; set; }

public List<int> DogIds { get; set; }

}

public class Expert

{

public int Id { get; set; }

public string FullName { get; set; }

public string Specialization { get; set; }

public int RingId { get; set; }

public string Club { get; set; }

}

public class Dog

{

public int Id { get; set; }

public string Name { get; set; }

public string Breed { get; set; }

public int Age { get; set; }

public string Club { get; set; }

public string OwnerFullName { get; set; }

public string Medal { get; set; }

}

}

}