**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

Федеральное государственное бюджетное образовательное учреждение высшего образования

**«Сибирский государственный университет науки и технологий  
имени академика М.Ф. Решетнева»**

Институт информатики и телекоммуникаций

Кафедра информатики и вычислительной техники

**КУРСОВАЯ РАБОТА**

Записная книжка

тема работы

Руководитель \_\_\_\_\_\_\_\_\_\_\_ Вдовенко В.В.

подпись, дата инициалы, фамилия

Обучающийся БПИ22-01 221219019 \_\_\_\_\_\_\_\_\_\_\_ Лобачев Д.К.

номер группы, зачетной книжки подпись, дата инициалы, фамилия

Красноярск 2024

Институт информатики и телекоммуникаций

Кафедра информатики и вычислительной техники

|  |
| --- |
| **ЗАДАНИЕ**  на курсовую работу по дисциплине «Объектно-ориентированное программирование»  студенту |
| Группа БПИ22-01 Форма обучения очная |  |
| 1. Тема работы: Записная книжка |
| 2. Срок сдачи студентом работы: |
| 3. Перечень вопросов, подлежащих разработке при написании теоретической части:  1. Системный анализ предметной области  2. Объектно-ориентированное моделирование программы:  2.1. Разработка UML-диаграммы прецедентов  2.2. Разработка UML-диаграммы активности  3. Разработка архитектуры программы |
| 4. Перечень вопросов, подлежащих разработке при написании практической части (либо указать номер варианта задания)  1. Разработка кода программы  2. Тестирование программы  3. Разработка руководства программиста |
| 4. Разработка руководства пользователя  5. Дата выдачи задания: 08.02.24 |
| Руководитель: Вдовенко Валентина Васильевна |
| Подпись |
|  |
| Задание принял к исполнению (дата): 08.02.24  (подпись студента) |
|  |

**Содержание**

[Введение](#_Toc169667869) 5

[1 Анализ предметной области приложений для хранения заметок 6](#_Toc169667870)

[1.1 История средств для ведения записей 6](#_Toc169667871)

[1.2 Обзор существующих аналогов 8](#_Toc169667872)

[1.3 Выводы 13](#_Toc169667873)

[2 Разработка программы «Моя записная книжка» 15](#_Toc169667874)

[2.1 UML-диаграмма прецедентов 15](#_Toc169667875)

[2.2 UML-диаграмма активностей 15](#_Toc169667876)

[2.3 Обоснование выбора инструментария разработки 16](#_Toc169667877)

[2.4 Руководство программиста. Диаграмма классов 18](#_Toc169667878)

[2.5 Руководство пользователя 19](#_Toc169667879)

[2.6 Выводы 24](#_Toc169667880)

[Заключение 25](#_Toc169667881)

[Список использованных источников 26](#_Toc169667882)

[Приложение. Код программы 27](#_Toc169667883)

# Введение

В наше быстро меняющееся время ведение записей и организация информации играют важную роль в повседневной жизни. Приложения для ведения заметок стали неотъемлемой частью нашего цифрового образа жизни, предоставляя удобный и эффективный способ сохранения и организации идей, задач, списков покупок, напоминаний и другой важной информации. Они помогают пользователям структурировать свои мысли, улучшить планирование и управление временем, а также повысить производительность.

Приложение "Моя записная книжка" представляет собой актуальное и востребованное решение, обеспечивающее пользователей инструментами для удобного хранения, организации и поиска записей. В современном мире, где информационный поток постоянно растет, такая разработка позволяет пользователям эффективно управлять своими данными, сохранять важные заметки и иметь к ним быстрый доступ в любое время.

Цель данной работы заключается в разработке программы «Моя записная книжка».

Для ее достижения предстоит выполнить ряд задач:

* рассмотрение особенностей программ для ведения заметок;
* провести обзор существующих аналогов;
* спроектировать программу, описать с помощью UML-диаграмм;
* реализовать и протестировать программное средство для ведения записной книжки;

Пояснительная записка к курсовой работе изложена на 30 страницах, включает 9 рисунков, 1 таблицу, введение, два раздела, заключение, список использованных источников из шести наименований, а также приложение с кодом программы.

В первой анализируется предметная область, проводится обзор средств-аналогов для ведения записей.

Во второй главе приведены UML-диаграммы проектирования программы, выполнено описание программы (классов и модулей, форм и интерфейса), демонстрируется работа программы «Моя записная книжка».

# 1 Анализ предметной области приложений для хранения заметок

## 1.1 История средств для ведения записей

История ведения заметок началась задолго до появления современных электронных устройств. В те дни, когда бумажные носители были единственным средством записи информации, люди использовали органайзеры, ежедневники и записные книжки для фиксации своих мыслей, планов и важных событий. Бумажные носители обладали своим очарованием и привлекательностью: возможность физически написать заметку, выразить свои мысли на бумаге, создавала особое ощущение личной связи с информацией.

Для многих людей использование бумажных записей оставалось предпочтительным способом сохранения информации. Органайзеры и ежедневники стали неотъемлемой частью жизни людей, отражая планы, встречи, идеи и важные события. Различные стили, материалы и дизайны записных книжек позволяли каждому найти подходящий вариант, отражающий его личный стиль.

Однако с развитием технологий и появлением электронных устройств произошел переход к новому способу ведения заметок. Сначала появились простые электронные органайзеры, предоставляющие возможность создания списков дел и заметок на экранах мобильных устройств. Эти первые приложения открывали новые возможности для организации информации, хотя и не обладали всеми функциями бумажных записей.

Одной из самых популярных платформ для электронных органайзеров была Palm OS (1996 год), которая использовала программы, такие как Date Book (Календарь), Address Book (Адресная книга), Memo Pad (Блокнот) и To-Do List (Список дел). Пользователи могли создавать заметки, планировать встречи, устанавливать напоминания и хранить контактные данные. На рисунках 1.1 и 1.2 приводятся примеры интерфейса Palm OS.

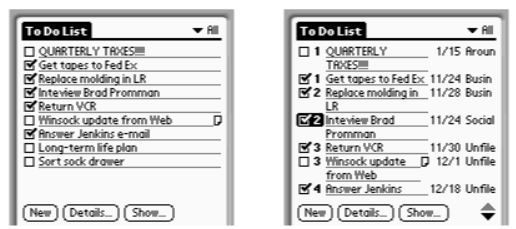
  
Рисунок 1.1 – Приложение To-Do List в Palm OS



Рисунок 1.2 – Средства ведения записей в Palm OS

Также Microsoft Outlook – одно из первых программных приложений для электронного органайзера, которое предоставляло возможность управления электронной почтой, календарем, контактами и задачами. Outlook позволял пользователям создавать заметки и списки дел, а также синхронизировать данные с компьютера на мобильные устройства (рисунок 1.3).

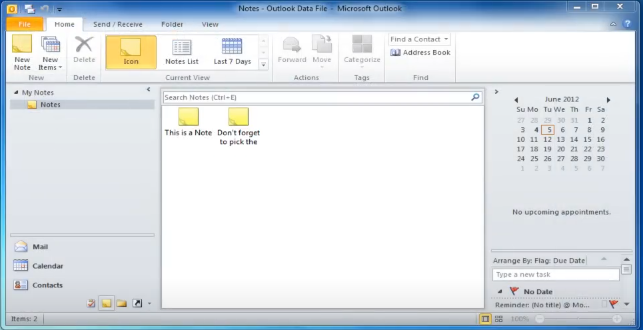


Рисунок 1.3 – Заметки в Outlook

Первые недостатки таких программ включали в себя ограниченные возможности синхронизации данных между различными устройствами. Также они часто требовали специального кабеля или инфракрасного соединения для передачи информации между устройствами. Интерфейсы и возможности редактирования текста были ограниченными по сравнению с бумажными записными книжками. Кроме того, ограниченные возможности хранения данных и отсутствие облачных сервисов делали сложным доступ к информации из любой точки, где бы вы ни находились.

С течением времени разработчики стали создавать более продвинутые приложения для ведения заметок, которые предлагали широкий спектр функций, включая создание различных типов заметок, добавление изображений, аудиофайлов и даже распознавание рукописного текста. Эти приложения обеспечивали возможность синхронизации данных между устройствами и облачное хранение, что делало доступ к информации более удобным и гибким.

Таким образом, эволюция средств для ведения заметок – от бумажных носителей к электронным записным книжкам и приложениям – отразила изменяющиеся потребности пользователей и привнесла новые возможности для организации информации в повседневной жизни.

## 1.2 Обзор существующих аналогов

Рассмотрим несколько известных программ-аналогов для ведения заметок и их основные характеристики.

Редактор *Together* предоставляет пользователю огромное количество функций. Здесь и довольно серьезные для заметок возможности редактирования текста и даже редактор изображений. Пользователь может добавить надпись или подрисовать что-то на изображении самостоятельно. Однако внешний вид главного окна приложения неопытного пользователя может удивить. Обилие элементов управления, меню и навигации усложняет работу с приложением (рисунок 1.4).

Так же данное приложение ориентировано в основном на работу с файлами мультимедиа и изображениями. Данный ресурс разработан для применения на мобильных устройствах, базируется на операционной системе МАС-ОС. В приложении реализовано несколько моделей структурирования данных: стандартное дерево каталогов по темам или дате, как пожелает пользователь, а так же Together сортирует сами данные во всех заметках, создавая библиотеки текста, видео, музыки и графики.



Рисунок 1.4 – Интерфейс программы Together

*Evernote* – это мощное приложение для создания заметок, которое позволяет пользователям сохранять и организовывать различные типы контента, включая текст, изображения, аудио и видео. Одной из ключевых особенностей Evernote является его способность синхронизировать заметки между различными устройствами, что позволяет пользователям получать доступ к своим данным в любое время и в любом месте. Пользователи могут также добавлять теги к своим заметкам для удобной организации и быстрого поиска. Кроме того, Evernote предоставляет возможность создания списков дел, что делает его полезным инструментом для управления задачами (рисунок 1.5).

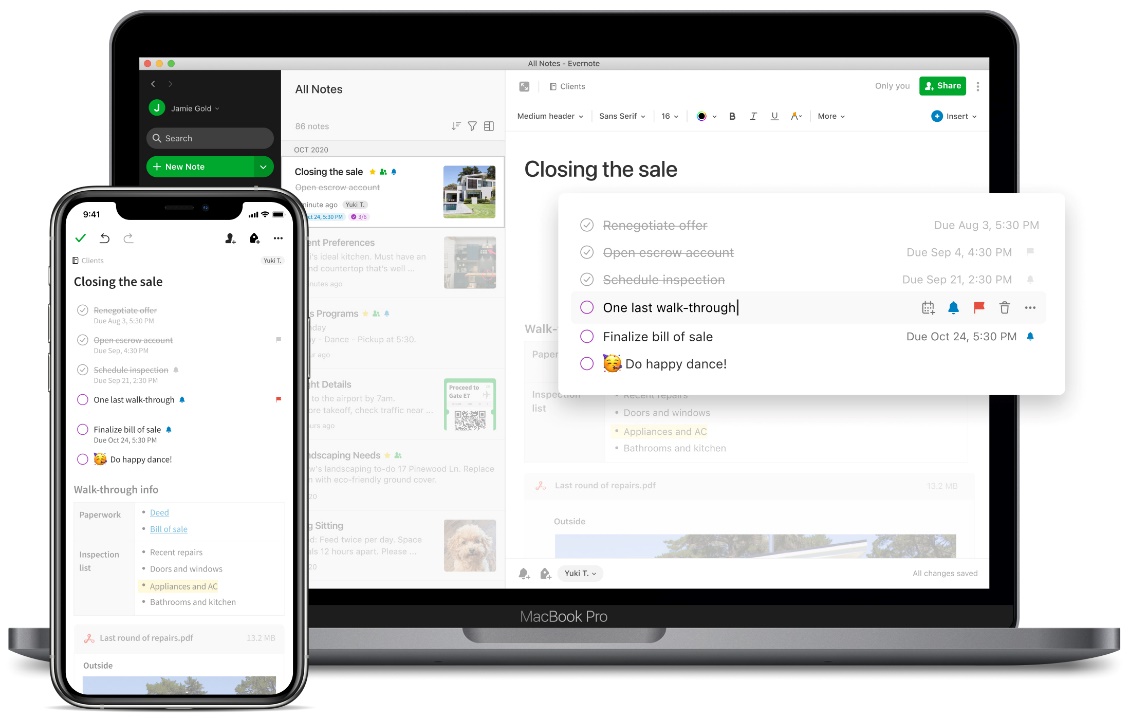


Рисунок 1.5 – Приложение Evernote

Evernote также обеспечивает возможность сканирования и распознавания текста в изображениях, что позволяет пользователям легко находить нужную информацию. Помимо этого, приложение предлагает облачное хранение данных, что гарантирует их сохранность и доступность. Пользователи могут экспортировать свои заметки в форматах HTML и ENEX для обмена информацией с другими пользователями или для резервного копирования.

*Microsoft OneNote* представляет собой инструмент для создания и организации заметок, который позволяет пользователям структурировать свою информацию в виде блокнотов, разделов и страниц. Одной из ключевых особенностей OneNote является его интеграция с облачным хранилищем OneDrive, что обеспечивает автоматическую синхронизацию данных между устройствами и доступ к заметкам через веб-версию или мобильное приложение (пример с изображением приведен на рисунке 1.6).

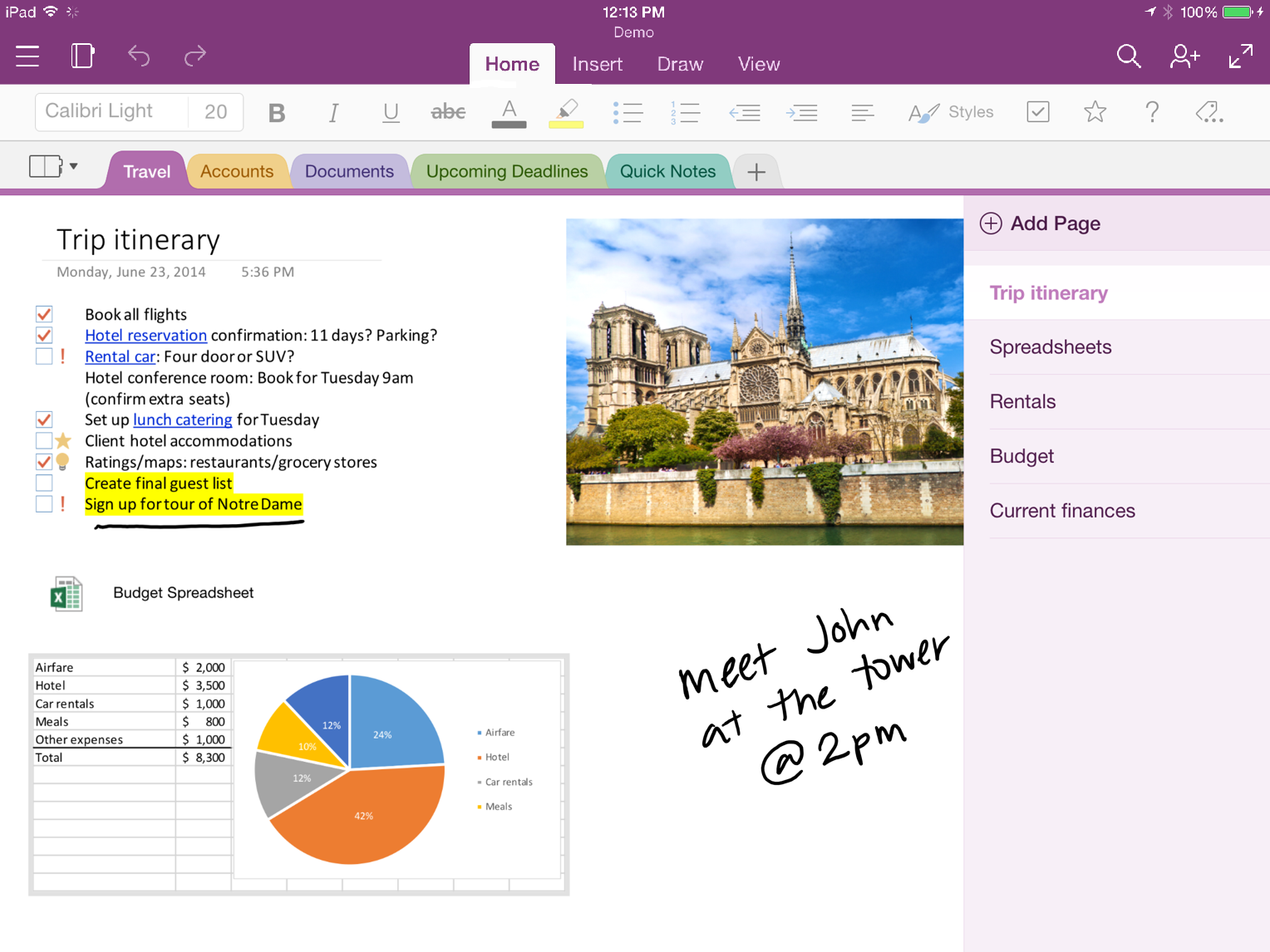


Рисунок 1.6 – Microsoft OneNote

Редактор очень удобен, немалую роль играет в этом продуманная панель инструментов Microsoft Office. Доступные средства редактирования, как текста, так и других видов данных достаточно масштабны (с учетом того, что речь идет о заметках, а не о полноценном редакторе мультимедиа файлов). Пользователи могут создавать заметки, добавлять файлы и изображения, а также использовать рисование и рукописный ввод для создания содержания. Это делает OneNote универсальным инструментом как для создания текстовых документов, так и для заметок в виде рисунков или диаграмм. Приложение также предоставляет возможность экспорта заметок в форматах PDF и OneNote для обмена информацией с другими пользователями или для сохранения данных. Интерфейс приложения прост, понятен и удобен. Открытая для редактирования заметка напоминает обычную записную книжку с разделителями. Приложение ориентировано на настольное применение, создано на базе операционной системы Windows.

*Google Keep* – это простое и интуитивно понятное приложение для создания заметок, которое предлагает пользователям возможность быстро записывать свои мысли, создавать списки дел и сохранять голосовые заметки. Одной из особенностей Google Keep является его простой интерфейс, который делает процесс создания и организации заметок быстрым и удобным.

Пользователи могут использовать метки для организации заметок по категориям, что облегчает их последующий поиск. Google Keep также интегрирован с другими сервисами Google, что позволяет пользователям легко передавать данные между различными приложениями. Пользователи могут экспортировать свои заметки в формате Google Docs для дальнейшего редактирования или обмена информацией с другими пользователями (интерфейс программы приведен на рисунке 1.7).

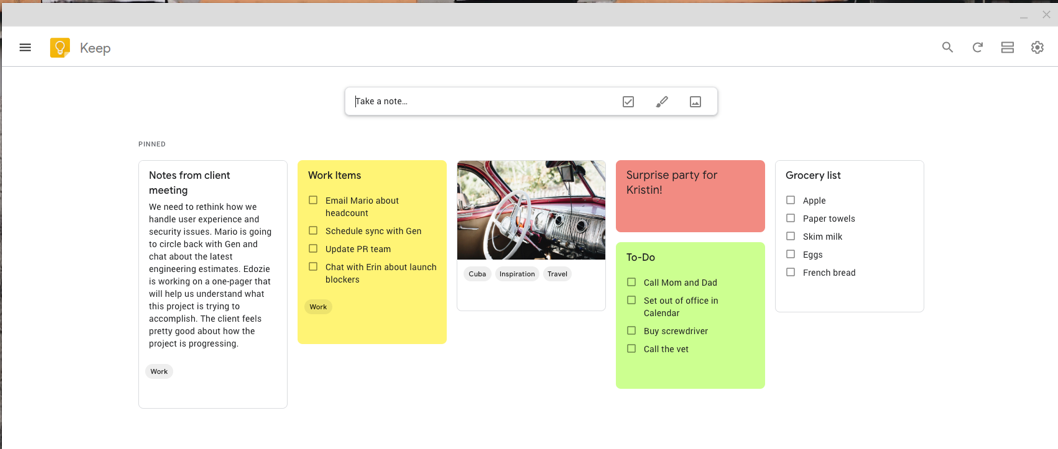


Рисунок 1.7 – Программа Google Keep

*Apple Notes* – это приложение, интегрированное в экосистему устройств Apple (iOS и macOS), которое предоставляет пользователям возможность создавать заметки с использованием текста, изображений, рисунков и чек-листов. Одной из ключевых особенностей Apple Notes является его простота использования и интуитивный интерфейс, который делает процесс создания и организации заметок легким и приятным (рисунок 1.8).

Пользователи могут также использовать рукописный ввод или рисование для создания содержания своих заметок. Apple Notes также интегрирован с другими сервисами Apple, что обеспечивает автоматическую синхронизацию данных между устройствами. Пользователи могут экспортировать свои заметки в форматах PDF и текстовых файлов для обмена информацией с другими пользователями или для сохранения данных.

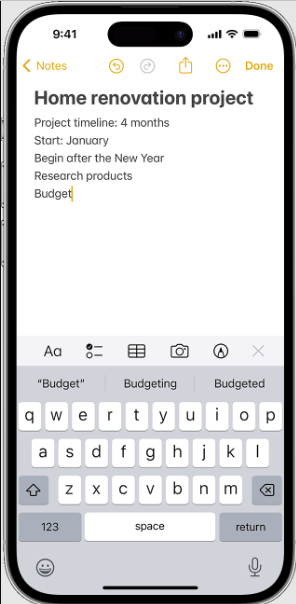


Рисунок 1.8 – AppleNotes на экране смартфона

В таблицу 1.1 сведена сравнительная характеристика рассмотренных приложений по основным показателям.

Таблица 1.1 – Сравнительная характеристика аналогов

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Характеристика | Together | Evernote | OneNote | Google Keep | AppleNotes |
| Платформы | macOS | множество | множество | множество | macOS |
| Сложность интерфейса | сложный | средняя | средняя | простой | простой |
| Редактирование медиа | + | изображения,  аудио, видео | + | + (включая голосовые заметки) | + |
| Организация заметок | простой список | метки | блокноты, разделы, страницы | метки | чек-листы, рисунки |
| Интеграция с сервисами | - | - | OneDrive | Google сервисы | экосистема Apple |

## 1.3 Выводы

В данном разделе была рассмотрена история программ для ведения записей, которая берет начало с 90-х годов 20 века, а наибольшее развитие получило в 2010-х годах. Выполнен анализ популярных программ-записных книжек, а именно: Togethe, Evernote, OneNote, Google Keep, Apple Notes.

# 2 Разработка программы «Моя записная книжка»

## 2.1 UML-диаграмма прецедентов

На рисунке 2.1 представлена диаграмма прецедентов для программы «Моя записная книжка». На ней указаны варианты использования, доступные пользователю (единственному актору) в приложении. Пользователь может не только добавлять и удалять заметки, но и выполнять поиск (отбор записей) по ключевым словам. Кроме того, записи можно упорядочивать по названию и дате занесения в записную книжку. Также приложение будет поддерживать возможности сохранения заметок в текстовый файл и последующую загрузку записей из файла.

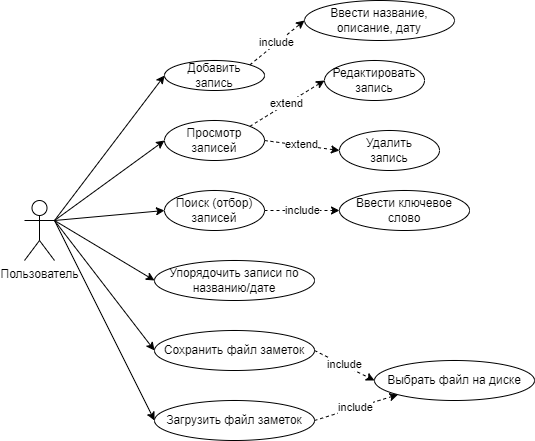


Рисунок 2.1 – Диаграмма вариантов использования

## 2.2 UML-диаграмма активностей

На рисунке 2.2 изображена диаграмма активностей для разрабатываемой программы. По существу, эта диаграмма представляет собой блок-схему, которая наглядно показывает, как поток управления переходит от одной деятельности к другой.

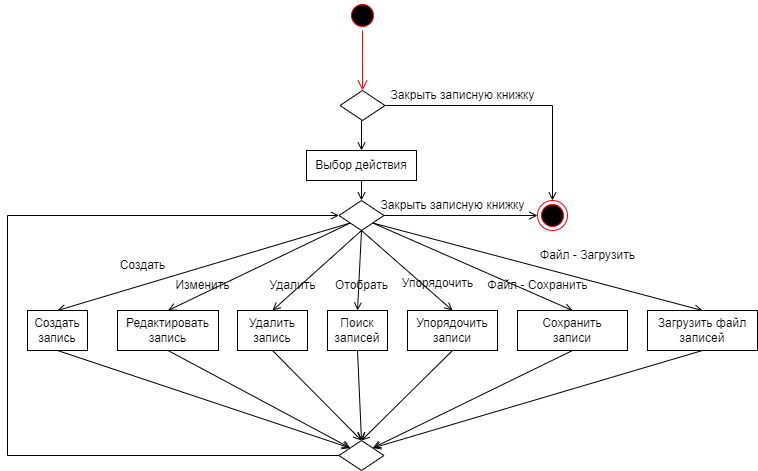


Рисунок 2.2 – Диаграмма активностей

При запуске программы записной книжки пользователь может выбрать одно из доступных действий. После выполнения той или иной операции пользователь увидит обновленный список записей и снова может работать с ним по этой же схеме. При этом в любой момент пользователь может прекратить работу программы.

## 2.3 Обоснование выбора инструментария разработки

При выборе языка программирования для разработки системы для ведения записей важно учесть различные аспекты, такие как производительность, простота синтаксиса, наличие необходимых инструментов и библиотек. Рассмотрим три популярных языка программирования: C++, C# и Java.

C++ – это мощный язык программирования, известный своей высокой производительностью и эффективным использованием ресурсов. Он часто используется для системного программирования и разработки приложений, где требуется близкое к аппаратуре управление. Однако, сложный синтаксис и управление памятью могут привести к ошибкам, что делает его менее привлекательным для начинающих программистов.

C# – язык программирования, разработанный компанией Microsoft, который отличается простым и понятным синтаксисом, похожим на Java. Он широко используется для создания приложений под платформу .NET и поддерживает объектно-ориентированное программирование. C# имеет хорошую интеграцию с платформой .NET, что обеспечивает переносимость кода и удобную среду разработки. Для разработки приложений с графическим интерфейсом в C# не требуется использование дополнительных библиотек, что делает его привлекательным выбором для создания пользовательских интерфейсов.

Java – еще один популярный язык программирования, известный своей переносимостью благодаря виртуальной машине Java (JVM). Он обладает обширной стандартной библиотекой и известен своей безопасностью и надежностью. Однако, некоторые разработчики могут посчитать его менее производительным по сравнению с другими языками.

Для разработки на C# существует несколько популярных сред разработки, таких как Visual Studio, официальная среда разработки от Microsoft, предлагающая широкие возможности для создания приложений под платформу .NET. Visual Studio Code – легковесная среда разработки, также поддерживающая C# с помощью расширений. JetBrains Rider – мощная среда разработки, поддерживающая C# и другие языки программирования.

Для разработки приложения «Моя записная книжка» было принято решение использовать язык программирования C#. Для разработки будет использоваться среда программирования Visual Studio 2022.

Выбор Visual Studio в качестве среды разработки обоснован рядом ключевых преимуществ. Во-первых, Visual Studio предлагает широкий спектр инструментов и возможностей для разработки приложений любого уровня сложности, что позволит создать надежную и функциональную систему для школьной столовой. Благодаря своей интегрированной среде разработки, Visual Studio обеспечивает удобство и эффективность работы программиста.

Важное преимущество Visual Studio 2022 – предоставление возможности работы с различными типами проектов, включая консольные приложения, веб-приложения, базы данных и другие, что позволит создать комплексное решение для школьной столовой. Кроме того, Visual Studio обладает обширными возможностями отладки, профилирования и тестирования кода, что поможет обеспечить высокое качество разрабатываемой системы.

C# предоставляет обширные возможности для создания мощных и надежных приложений благодаря своему синтаксису, богатым стандартным библиотекам и поддержке объектно-ориентированного программирования. C# является основным языком программирования платформы .NET, что обеспечивает высокую совместимость с другими технологиями Microsoft. C# имеет четкую структуру и множество инструментов разработки, что делает его привлекательным для новичков и опытных разработчиков.

## 2.4 Руководство программиста. Диаграмма классов

Проект имеет модульную структуру и включает в себя несколько классов:

* Entry – класс записи (модуль EntryList.cs);
* EntryList – класс списка записей (модуль EntryList.cs);
* Form1 – главная форма приложения (Form1.cs);
* EntryEditForm – форма для ввода новой записи (EntryEditForm.cs).

На рисунке 2.3 приведена диаграмма классов (без учета классов форм).

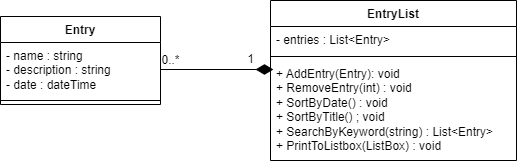


Рисунок 2.3 – Диаграмма классов

Таким образом, класс EntryList агрегирует записи Entry, которых может быть от нуля до неограниченного количества в списке. Каждая запись Entry характеризуется названием, описанием, датой добавления. Класс EntryList содержит только список (обобщенный список List<Entry>) и обладает рядом методов для работы с ним:

1. AddEntry – добавляет новую запись в список.
2. RemoveEntry – удаляет указанную запись из списка по индексу. Если передан некорректный индекс, удаления не происходит.
3. SortByDate – упорядочить записи по дате.
4. SortByTitle – упорядочить записи по названию.
5. SearchByKeyword – осуществляет поиск записей по указанному слову в названии или описании и возвращает соответствующие записи в виде нового списка List<Entry>.
6. PrintToListbox – вывести записи в список ListBox.

Методы для сортировки и выборки используют LINQ-запросы к списку entries.

В коде главной формы Form1.cs создается объект записной книжки entries. В обработчиках событий формы происходит работа с записной книгой путем вызова методов класса. Работа почти всех кнопок заключается в последовательности: выбрать действие, указать (при необходимости) данные, обновить список записей, вывести список на форму.

Кроме того, в Form1.cs реализованы дополнительные методы для работы с файлами:

* void WriteToFile(string filePath) – сохранить записи в файле filePath;
* EntryList ReadFromFile(string filePath) – загрузить записи из файла filePath.

Данные методы вызываются при выборе пунктов из меню «Файл»: «сохранить» и «загрузить» соответветственно.

Форма EditEntryForm служит как для добавления новых записей, так и для редактирования существующих. Если при создании формы передается только список entries, то форма рассчитана на добавление записи. Если передается существующая запись, то форма служит для ее редактирования.

## 2.5 Руководство пользователя

На рисунке 2.4 представлена главная форма приложения, которую пользователь видит при запуске. Ему доступны действия по нажатию кнопок на форме.

Кнопка «Новая запись» откроет новую форму для добавления записи. Пользователю необходимо ввести все данные о новой записи, после чего она будет добавлена в записную книжку по нажатию на кнопку «Добавить» (рисунок 2.5).

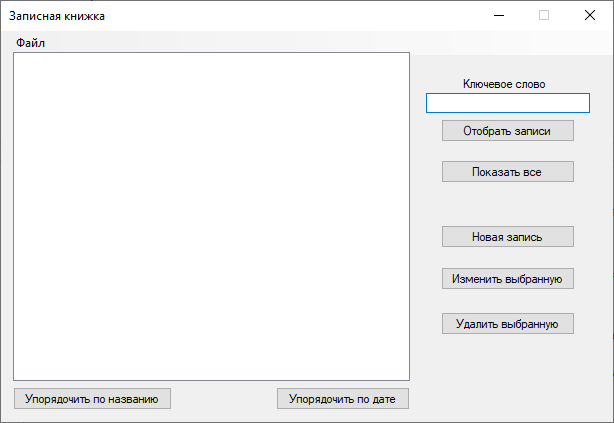


Рисунок 2.4 – Главная форма приложения

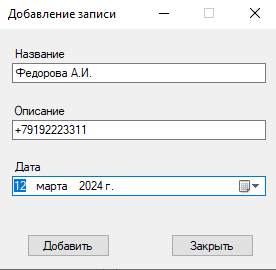


Рисунок 2.5 – Форма добавления записи

В результате нескольких добавлений получим список записей (рисунок 2.6). В нижней части формы расположены кнопки для упорядочивания записей по разным параметрам. При нажатии на кнопку «Упорядочить по названию» записи в списке сохранятся, но уже будут идти в алфавитном порядке по названию (рисунок 2.7).

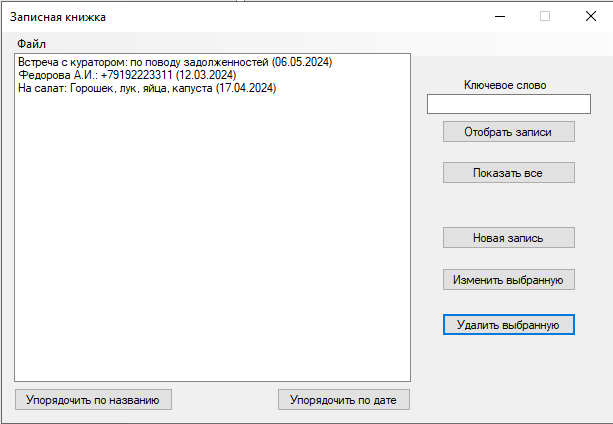


Рисунок 2.6 – Добавленные записи

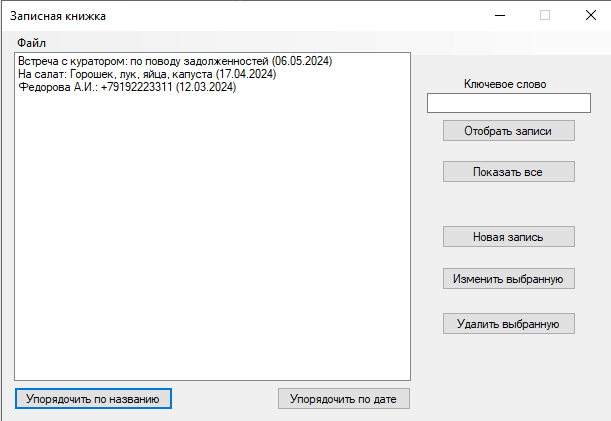


Рисунок 2.7 – Упорядочивание записей по названию

Соответствующим образом сработает нажатие по кнопке «Упорядочить по дате» – записи будут размещены в календарном порядке (рисунок 2.8).

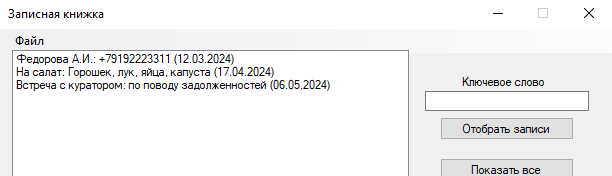


Рисунок 2.8 – Упорядочивание записей по дате

На данный момент в списке есть запись с названием «На салат». Добавим еще одну запись, которая будет содержать слово «салат» (рисунок 2.9). Затем в поле «Ключевое слово» введем «салат» и нажмем кнопку отобразить записи. В итоге будет получен список из двух записей, в которых фигурирует указанное слово.

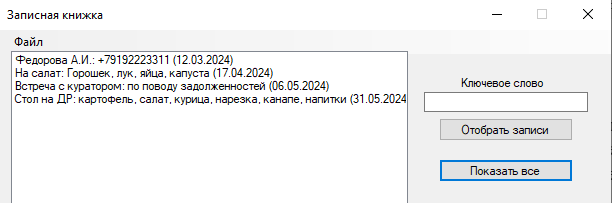


Рисунок 2.9 – Новая запись в списке

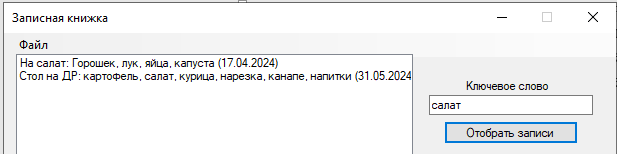


Рисунок 2.10 – Выборка по ключевому слову

Снова отобразим все записи путем нажатия на «Показать все». Теперь выберем в списке свежую запись («Стол на ДР») и нажмем «Удалить выбранную». В обновленном списке указанной записи не будет (рисунок 2.11).

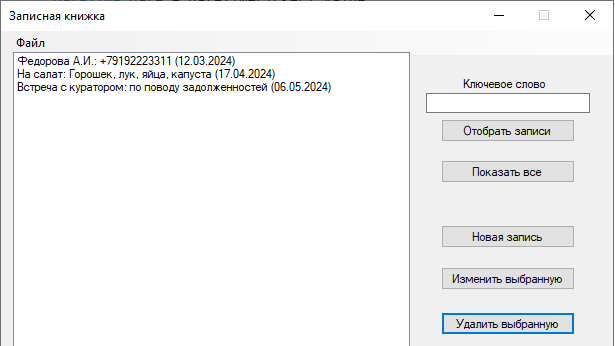


Рисунок 2.11 – Удаление записи

Записную книжку можно сохранить в файл на диске для последующей загрузки после перезапуска программы. Для этого выбираем пункт меню «Файл – Сохранить» и выбираем место для файла на диске (рисунок 2.12).

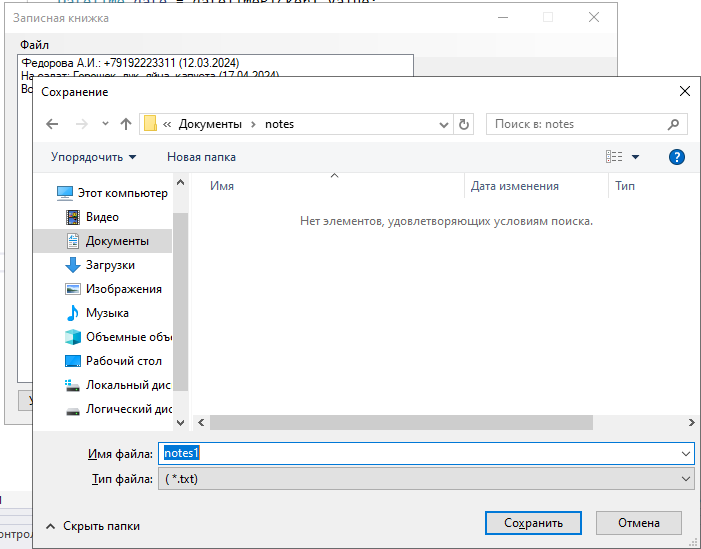


Рисунок 2.12 – Сохранение записной книжки в файле

Теперь, если перезапустить приложение, мы можем восстановить нашу записную книжку, выбрав пункт меню «Файл – Открыть». Выберем ранее сохраненный файл (рисунок 2.13) и убедимся, что загружены именно те записи, что и были сохранены (рисунок 2.14).

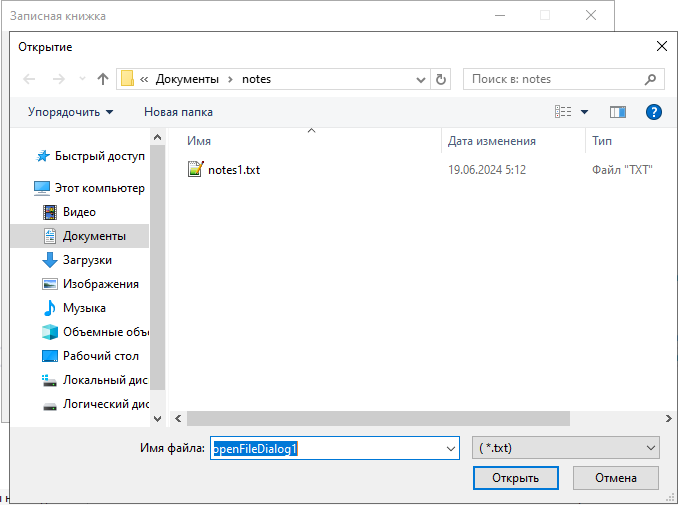


Рисунок 2.13 – Выбор файла для загрузки книжки

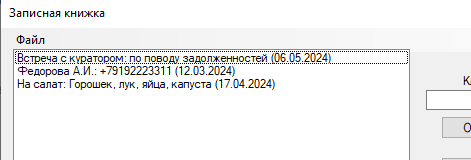


Рисунок 2.14 – Результат загрузки файла

## 2.6 Выводы

Программное средство разработано в соответствии с запланированным набором функций. Тестирование не выявило каких либо ошибок.

# Заключение

В ходе выполнения курсовой работы была рассмотрена и проанализирована предметная область программ для ведения записных книжек с заметками. Были выявлены ключевые параметры для приложений такой направленности. При выполнении работы были решены задачи:

* рассмотрение особенностей программ для ведения заметок;
* осуществлен обзор существующих аналогов;
* составлены UML-диаграммы, описывающие работу программы;
* реализовано программное средство для ведения записной книжки;
* проведено тестирование разработанной программы для записной книжки.

Второй раздел работы посвящен разработке приложения. Его функционал позволяет добавлять, редактировать, удалять заметки. А также производить поиск по ключевым словам, выполнять разное отображение заметок (по дате, по названию).

Таким образом, все поставленные в начале работы задачи можно считать выполненным, а цель – достигнутой.

# Список использованных источников

1. Голицина, О. Л., Попов, И. И. Основы алгоритмизации и программирования : учебное пособие / О. Л. Голицина, И. И. Попов. – М. : ФОРУМ: ИНФРА, 2015. – 442 c.
2. Давыдов, В.И., Кунгурцев, А.Б. Объектно-ориентированное программирование / В.И. Давыдов, А.Б. Кунгурцев. – Одесса : Автограф, 2019. – 124 с.
3. Майо, Д. Самоучитель Microsoft Visual Studio 2022 / Д. Майо. – CПб.: «БХВ-Петербург», 2021. – 464 с.
4. Павловская, Т. А. C#. Программирование на языке высокого уровня : учебник для вузов / Т. А. Павловская. – СПб.: БХВ-Петербург, 2019. – 462 c.
5. Рамбо, Дж UML 2.0: Объектно-ориентированное моделирование и разработка / Дж Рамбо - Санкт-Петербург: Питер, 2020. – 554 с.
6. Белов, В.В. Проектирование информационных систем: Учебник / В.В. Белов. - М.: Академия, 2018, – 144 с.

# Приложение. Код программы

EntryList.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Windows.Forms;

namespace MyNotes

{

//класс для записи

public class Entry

{

public string Title { get; set; } //название

public string Description { get; set; } //описание

public DateTime Date { get; set; } //дата

public Entry(string title, string description, DateTime date)

{

this.Title = title;

this.Description = description;

this.Date = date;

}

//преобразование записи в строку

public override string ToString()

{

return $"{Title}: {Description} ({Date.ToString("dd.MM.yyyy")})";

}

}

//класс для работы со списком записей

public class EntryList

{

private List<Entry> entries = new List<Entry>(); //список записей

public List<Entry> Entries() { return entries; }

// внести новую запись

public void AddEntry(Entry entry)

{

entries.Add(entry);

}

// удалить запись по индексу

public void RemoveEntry(int index)

{

if (index >= 0 && index < entries.Count) // проверка на корректность

{

entries.RemoveAt(index);

}

}

// упорядочить по дате

public void SortByDate()

{

entries = entries.OrderBy(e => e.Date).ToList();

}

// упорядочить по названию

public void SortByTitle()

{

entries = entries.OrderBy(e => e.Title).ToList();

}

// поиск по ключевому слову (и в названии, и в описании)

public List<Entry> SearchByKeyword(string keyword)

{

return entries.Where(e => e.Title.Contains(keyword) || e.Description.Contains(keyword)).ToList();

}

// метод для вывода всех записей в объект ListBox

public void PrintToListbox(ListBox listBox)

{

listBox.Items.Clear(); // очищаем перед добавлением новых элементов

foreach (var entry in entries)

{

listBox.Items.Add(entry.ToString()); // добавляем строковое представление записи

}

}

}

}

Form1.cs

using System;

using System.Collections.Generic;

using System.IO;

using System.Windows.Forms;

namespace MyNotes

{

public partial class Form1 : Form

{

private EntryList entries; //список с заметками

public Form1()

{

InitializeComponent();

// инициализация списка несколькими записями

entries = new EntryList();

//entries.AddEntry(new Entry("Встреча с куратором", "по поводу задолженностей", new DateTime(2024, 5, 6)));

//entries.AddEntry(new Entry("Федорова А.И.", "+79192223311", new DateTime(2024, 3, 12)));

//entries.AddEntry(new Entry("На салат", "Горошек, лук, яйца, капуста", new DateTime(2024, 4, 17)));

//entries.PrintToListbox(listBox1);

}

private void button4\_Click(object sender, EventArgs e)

{

if (listBox1.SelectedIndex != -1)

{

EntryEditForm editForm = new EntryEditForm(entries, entries.Entries()[listBox1.SelectedIndex]);

editForm.Show();

}

}

private void button2\_Click(object sender, EventArgs e)

{

if (listBox1.SelectedIndex != -1)

{

entries.RemoveEntry(listBox1.SelectedIndex);

entries.PrintToListbox(listBox1);

}

}

private void button5\_Click(object sender, EventArgs e)

{

entries.SortByTitle();

entries.PrintToListbox(listBox1);

}

private void button3\_Click(object sender, EventArgs e)

{

entries.SortByDate();

entries.PrintToListbox(listBox1);

}

private void button1\_Click(object sender, EventArgs e)

{

EntryEditForm editForm = new EntryEditForm(entries);

editForm.Show();

}

// метод для записи списка в текстовый файл

public void WriteToFile(string filePath)

{

using (StreamWriter writer = new StreamWriter(filePath))

{

writer.WriteLine(entries.Entries().Count); // записываем количество записей

foreach (var entry in entries.Entries())

{

string entryString = $"{entry.Title};{entry.Description};{entry.Date.ToString("dd.MM.yyyy")}";

writer.WriteLine(entryString); // записываем каждую запись в формате "название;описание;дата"

}

}

}

// метод для восстановления списка из текстового файла

public EntryList ReadFromFile(string filePath)

{

EntryList newList = new EntryList();

using (StreamReader reader = new StreamReader(filePath))

{

int count = int.Parse(reader.ReadLine()); // считываем количество записей

for (int i = 0; i < count; i++)

{

string[] entryData = reader.ReadLine().Split(';'); // считываем и разбираем строку с данными о записи

string title = entryData[0];

string description = entryData[1];

DateTime date = DateTime.ParseExact(entryData[2], "dd.MM.yyyy", null);

Entry newEntry = new Entry(title, description, date);

newList.AddEntry(newEntry); // добавляем новую запись в список

}

}

return newList;

}

//загрузить файл с заметками через меню

private void открытьToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (openFileDialog1.ShowDialog() != DialogResult.Cancel)

{

try

{

entries = ReadFromFile(openFileDialog1.FileName); //загрузка списка

entries.PrintToListbox(listBox1); //вывод на форму

}

catch (Exception exc)

{

MessageBox.Show("Ошибка чтения файла: " + exc.Message);

}

}

}

//сохранить файл с заметками через меню

private void сохранитьToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (saveFileDialog1.ShowDialog() != DialogResult.Cancel)

{

try

{

WriteToFile(saveFileDialog1.FileName); //сохранение списка

}

catch (Exception exc)

{

MessageBox.Show(exc.Message);

}

}

}

//вывести все записи

private void button7\_Click(object sender, EventArgs e)

{

entries.PrintToListbox(listBox1);

}

//выборка записей по ключевому слову

private void button6\_Click(object sender, EventArgs e)

{

string keyword = textBox1.Text;

List<Entry> selected = entries.SearchByKeyword(keyword); //выборка записей

if (selected.Count != 0) //если записи были выбраны

{

listBox1.Items.Clear();

foreach (Entry entry in selected)

{

listBox1.Items.Add(entry);

}

}

else

MessageBox.Show("Нет записей по заданному слову");

}

}

}

EditEntryForm.cs

using System;

using System.Windows.Forms;

namespace MyNotes

{

public partial class EntryEditForm : Form

{

EntryList entryList;

Entry entry;

public EntryEditForm(EntryList entryList, Entry entry=null)

{

InitializeComponent();

this.entryList = entryList;

if (entry != null)

{

button1.Text = "Изменить";

textBox1.Text = entry.Title;

textBox2.Text = entry.Description;

}

}

//добавить запись

private void button1\_Click(object sender, EventArgs e)

{

string title = textBox1.Text;

string description = textBox2.Text;

DateTime date = dateTimePicker1.Value;

if (entry == null)

{

//новая запись в списке

Entry newEntry = new Entry(title, description, date);

entryList.AddEntry(newEntry);

}

else

{

entry.Title = title;

entry.Date = date;

entry.Description = description;

}

this.Close();

}

private void button2\_Click(object sender, EventArgs e)

{

this.Close();

}

}

}