

УТВЕРЖДАЮ

Руководитель практики от Университета

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*(должность, ФИО)*

МП \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*(Подпись)*

Дата « \_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 202\_ год

**ОТЧЕТ**

**ПО УЧЕБНОЙ ПРАКТИКЕ**

ПМ.05 Проектирование и разработка информационных систем

|  |  |
| --- | --- |
| Щекотова Екатерина Алексеевна | |
| *(Ф.И.О. обучающегося)* | |
| 09.02.07 Информационные системы и программирование | |
| *(специальность)* | |
|  | |
| Учебная группа | ИСПк-203-52-00 |
|  |  |
| Место прохождения практики | ФГБОУ ВО «Вятский государственный университет», |
| Колледж ВятГУ | *(наименование организации, структурного подразделения организации)* |
|  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Итоговая оценка: |  | | | | |
| Руководитель  практики от университета |  |  |  |  |  |
|  | *(дата)* |  | *(подпись)* |  | *(Ф.И.О.)* |

Киров, 2023 г.



ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ НА ПРАКТИКУ

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Ф.И.О. обучающегося | | | Щекотова Екатерина Алексеевна | | | | | | |
| Специальность | | | 09.02.07 Информационные системы и программирование | | | | | | |
| Учебная группа | | | ИСПк-203-52-00 | | | | | | |
| Вид практики | | | учебная практика | | | | | | |
| Сроки прохождения практики с | | | | 09.01.2023 | по | 15.06.2023 | | | |
| Место прохождения практики | | | | ФГБОУ ВО «Вятский государственный университет»,  Колледж ВятГУ | | | | | |
|  | | (наименование организации, структурного подразделения организации) | | | | | | |
| № | Виды работ, выполняемых обучающимися во время практики | | | | | | Объем работ (час) | Формируемые компетенции | | |
| 1 | Пройти инструктаж по ознакомлению с правилами внутреннего трудового распорядка, охраны труда, техники безопасности, противопожарной безопасности, санитарно-эпидемиологическими правилами и гигиеническими нормативами, а также вводный инструктаж и инструктаж на рабочем месте | | | | | | 2 | - | | |
| 2 | Постановка задачи | | | | | | 8 | ОК2, ОК6, ОК7, ОК9 | | |
| 3 | Настройка рабочего окружения | | | | | | 8 | ОК1, ОК8, ОК9 | | |
| 4 | Анализ предметной области, постановка расширенного технического задания | | | | | | 10 | ПК5.1, ОК3, ОК4 | | |
| 5 | Проектирование | | | | | | 35 | ПК5.2, ПК5.3, | | |
| 6 | Реализация программного кода | | | | | | 35 | ПК5.4, ПК5.8, | | |
| 7 | Тестирование и отладка полученного кода | | | | | | 22 | ПК5.5, ПК6.4, ОК2, ОК9 | | |
| 8 | Подготовка отчетной документации, включая детальное описание разработанных и использованных алгоритмов | | | | | | 10 | ПК5.6, ПК5.7, ОК9, ОК10, ОК11 | | |
| 9 | Подготовка к сдаче и сдача промежуточной аттестации | | | | | | 2 | ОК4, ОК5 | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Индивидуальное задание на практику разработано в соответствии с рабочей программой практики. | | | | | |
| Руководитель  практики от университета |  |  |  |  |  |
|  | (дата) |  | (подпись) |  | (Ф.И.О.) |

|  |  |
| --- | --- |
| С индивидуальным заданием ознакомлен(а) |  |
|  | (дата, подпись обучающегося) |

**ОТЗЫВ РУКОВОДИТЕЛЯ ПРАКТИКИ**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Ф.И.О. обучающегося | | Щекотова Екатерина Алексеевна | | | |
| Специальность | | 09.02.07 Информационные системы и программирование | | | |
| Учебная группа | | ИСПк-203-52-00 | | | |
| Вид практики | | учебная практика | | | |
| Сроки прохождения практики с | | | 09.01.2023 | по | 15.06.2023 |
| Место прохождения практики | | | ФГБОУ ВО «Вятский государственный университет», Колледж ВятГУ | | |
|  | (наименование организации, структурного подразделения организации) | | | | |

ВИДЫ И КАЧЕСТВО ВЫПОЛНЕНИЯ РАБОТ

|  |  |  |  |
| --- | --- | --- | --- |
| Вид работ | Критерий выполнения работ | | |
| Выполнены полностью самостоятельно | Выполнены с незначительной помощью наставника | Выполнены с помощью наставника |
| Постановка задачи | V |  |  |
| Настройка рабочего окружения | V |  |  |
| Анализ предметной области, постановка расширенного технического задания | V |  |  |
| Проектирование | V |  |  |
| Реализация программного кода | V |  |  |
| Тестирование и отладка полученного кода | V |  |  |
| Подготовка отчетной документации, включая детальное описание разработанных и использованных алгоритмов | V |  |  |
| Подготовка к сдаче и сдача промежуточной аттестации | V |  |  |

Обучающийся ознакомлен с правилами внутреннего трудового распорядка, охраны труда, техники безопасности, противопожарной безопасности, санитарно-эпидемиологическими правилами и гигиеническими нормативами, а также прошел вводный инструктаж и инструктаж на рабочем месте.

Во время прохождения учебной практики обучающимся освоены следующие профессиональные и общие компетенции:

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование компетенции | Показатели оценки | Оценка | |
| Освоена | Не освоена |
| ПК 5.1. Собирать исходные данные для разработки проектной документации на информационную систему. | Способен выполнять анализ предметной области, выявляя существенные элементы, оказывающие влияние на проектируемую систему | V |  |
| ПК 5.2. Разрабатывать проектную документацию на разработку информационной системы в соответствии с требованиями заказчика. | Способен выполнять декомпозицию сущностей с целью получения наиболее полной картины о целесообразной структуре разработки | V |  |
| ПК 5.3. Разрабатывать подсистемы безопасной информационной системы в соответствии с техническим заданием. | Способен прогнозировать потенциально некорректные действия пользователя и предусматривать соответствующие реакции со стороны системы | V |  |
| ПК 5.4. Производить разработку модулей информационной системы в соответствии с техническим заданием. | Способен создавать программный код, отвечающий предъявляемым требованиям | V |  |
| ПК 5.5. Осуществлять тестирование информационной системы на этапе опытной эксплуатации с фиксацией выявленных ошибок кодирования в разрабатываемых модулях информационной системы. | Способен выполнять оценку корректности функционирования системы | V |  |
| ПК 5.6. Разрабатывать техническую документацию на эксплуатацию информационной системы. | Способен разрабатывать проектную, техническую и пользовательскую документации | V |  |
| ПК 5.7. Производить оценку информационной системы для выявления возможности ее модернизации. | Способен оценивать перспективы дальнейшего развития программной системы | V |  |
| ПК 5.8. Осуществлять интеграцию информационной системы с другими программными продуктами. | Способен повторно использовать готовые шаблонные решения при разработке программного продукта | V |  |
| ОК 01. Выбирать способы решения задач профессиональной деятельности, применительно к различным контекстам. | Способен анализировать специализированную информацию и находить оптимальные пути решения задач профессиональной деятельности | V |  |
| ОК 02. Осуществлять поиск, анализ и интерпретацию информации, необходимой для выполнения задач профессиональной деятельности. | Способен анализировать специализированную информацию и находить оптимальные пути решения задач профессиональной деятельности | V |  |
| ОК 03. Планировать и реализовывать собственное профессиональное и личностное развитие. | Способен находить пути улучшения имеющихся решений, позволяющих повысить их общий качественный уровень | V |  |
| ОК 04. Работать в коллективе и команде, эффективно взаимодействовать с коллегами, руководством, клиентами. | Способен конструктивно обмениваться информацией с коллегами, грамотно формулировать запросы в целях получения разъясняющей информации | V |  |
| ОК 05. Осуществлять устную и письменную коммуникацию на государственном языке с учетом особенностей социального и культурного контекста. | Способен конструктивно обмениваться информацией с коллегами, грамотно формулировать запросы в целях получения разъясняющей информации | V |  |
| ОК 06. Проявлять гражданско-патриотическую позицию, демонстрировать осознанное поведение на основе традиционных общечеловеческих ценностей. | Способен конструктивно обмениваться информацией с коллегами, грамотно формулировать запросы в целях получения разъясняющей информации | V |  |
| ОК 07. Содействовать сохранению окружающей среды, ресурсосбережению, эффективно действовать в чрезвычайных ситуациях. | Способен эффективно использовать компьютерное время, а также материальные ресурсы, необходимые для решения поставленных задач | V |  |
| ОК 08. Использовать средства физической культуры для сохранения и укрепления здоровья в процессе профессиональной деятельности и поддержания необходимого уровня физической подготовленности. | Способен соблюдать требования внутреннего трудового распорядка организации, охраны труда и техники безопасности в целях сохранения собственного здоровья | V |  |
| ОК 09. Использовать информационные технологии в профессиональной деятельности. | Способен применять стек современных средств разработки ПО для решения задач профессиональной деятельности в заданном контексте | V |  |
| ОК 10. Пользоваться профессиональной документацией на государственном и иностранном языке. | Способен использовать в своей работе специализированную документацию | V |  |
| ОК 11. Планировать предпринимательскую деятельность в профессиональной сфере. | Способен проектировать алгоритмические решения, принимая во внимание имеющиеся ресурсные ограничения | V |  |

**Краткая характеристика работы обучающегося**

|  |
| --- |
| Программа практики выполнена в полном объеме. Все виды работ выполнялись в срок, |
| без существенных замечаний. В достаточной степени была проявлена самостоятельность |
| и умение грамотно пользоваться источниками информации, находящимися в свободном |
| доступе. |

|  |  |  |
| --- | --- | --- |
|  |  | Руководитель практики  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Подпись ФИО  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (должность)  Дата «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 202\_\_ год |

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 5](#_Toc146249507)

[1. СВЕДЕНИЯ О РАБОТЕ, ВЫПОЛНЕННОЙ В ПЕРИОД ПРОХОЖДЕНИЯ УЧЕБНОЙ ПРАКТИКИ 6](#_Toc146249508)

[2. НАСТРОЙКА РАБОЧЕГО ОКРУЖЕНИЯ 7](#_Toc146249509)

[3. ФОРМУЛИРОВКА ИНДИВИДУАЛЬНОГО ЗАДАНИЯ 9](#_Toc146249510)

[4. ОПИСАНИЕ ВЫПОЛНЕНИЯ ИНДИВИДУАЛЬНОГО ЗАДАНИЯ 10](#_Toc146249511)

[4.1 Анализ предметной области и обзор аналогов 10](#_Toc146249512)

[4.2 Разработка структуры приложения и алгоритмов функционирования. 15](#_Toc146249513)

[4.3 Реализация программы. Разработка пользовательского интерфейса 23](#_Toc146249514)

[4.4 Тестирование программы. 35](#_Toc146249515)

[ЗАКЛЮЧЕНИЕ 38](#_Toc146249516)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 39](#_Toc146249517)

[ПРИЛОЖЕНИЕ А 40](#_Toc146249518)

[ПРИЛОЖЕНИЕ Б 45](#_Toc146249519)

[ПРИЛОЖЕНИЕ В 46](#_Toc146249520)

[ПРИЛОЖЕНИЕ Г 47](#_Toc146249521)

# ВВЕДЕНИЕ

Учебная практика ПМ.05 проходила на базе Колледжа ФГБОУ ВО «Вятский государственный университет» в период с 09.01.2023 г. по 15.06.2023 г. по субботам.

Цель практики: сформировать у обучающихся представление о процессе разработки прикладного программного обеспечения.

Задачи практики:

– дать представление о различных методах проектирования прикладного программного обеспечения;

– сформировать навыки написания документации, требуемой при разработке прикладного программного обеспечения;

– дать представление о различных инструментах, используемых при разработке прикладного программного обеспечения.

# 1. СВЕДЕНИЯ О РАБОТЕ, ВЫПОЛНЕННОЙ В ПЕРИОД ПРОХОЖДЕНИЯ УЧЕБНОЙ ПРАКТИКИ

В период с 09.01.2023 по 14.06.2023 при прохождении учебной практики ПМ.05 на базе Колледжа ФГОБУ ВО «Вятский государственный университет» был выполнен следующий перечень работ, представленный в таблице 1.

Таблица 1 – Сведения о работе, выполненной в период практики

| Дата | Краткое содержание выполненных работ |
| --- | --- |
| 11.01.2021 | Ознакомление с правилами внутреннего трудового распорядка, охраны труда, техники безопасности, противопожарной безопасности, санитарно-эпидемиологическими правилами и гигиеническими нормативами, а также вводный инструктаж и инструктаж на рабочем месте |
| 12.01.2021-25.01.2021 | Постановка задачи |
| 26.01.2021-08.02.2021 | Настройка рабочего окружения |
| 09.02.2021-22.02.2021 | Анализ предметной области, постановка расширенного технического задания |
| 23.02.2021-29.03.2021 | Проектирование |
| 30.03.2021-03.05.2021 | Реализация программного кода |
| 04.05.2021-24.05.2021 | Тестирование и отладка полученного кода |
| 25.05.2021-13.06.2021 | Подготовка отчетной документации, включая детальное описание разработанных и использованных алгоритмов |
| 14.06.2021-16.06.2021 | Подготовка к сдаче и сдача промежуточной аттестации |

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(дата) (подпись)

# 2. НАСТРОЙКА РАБОЧЕГО ОКРУЖЕНИЯ

Python — это свободный интерпретируемый объектно-ориентированный расширяемый встраиваемый язык программирования очень высокого уровня. Свободный — все исходные тексты интерпретатора и библиотек доступны для любого использования. Интерпретируемый — потому что использует «позднее связывание». Очень высокого уровня — динамическая типизация, встроенные типы данных высокого уровня, классы, модули, механизм исключений.

На рисунке ниже представлена среда разработки Visual Studio Code.

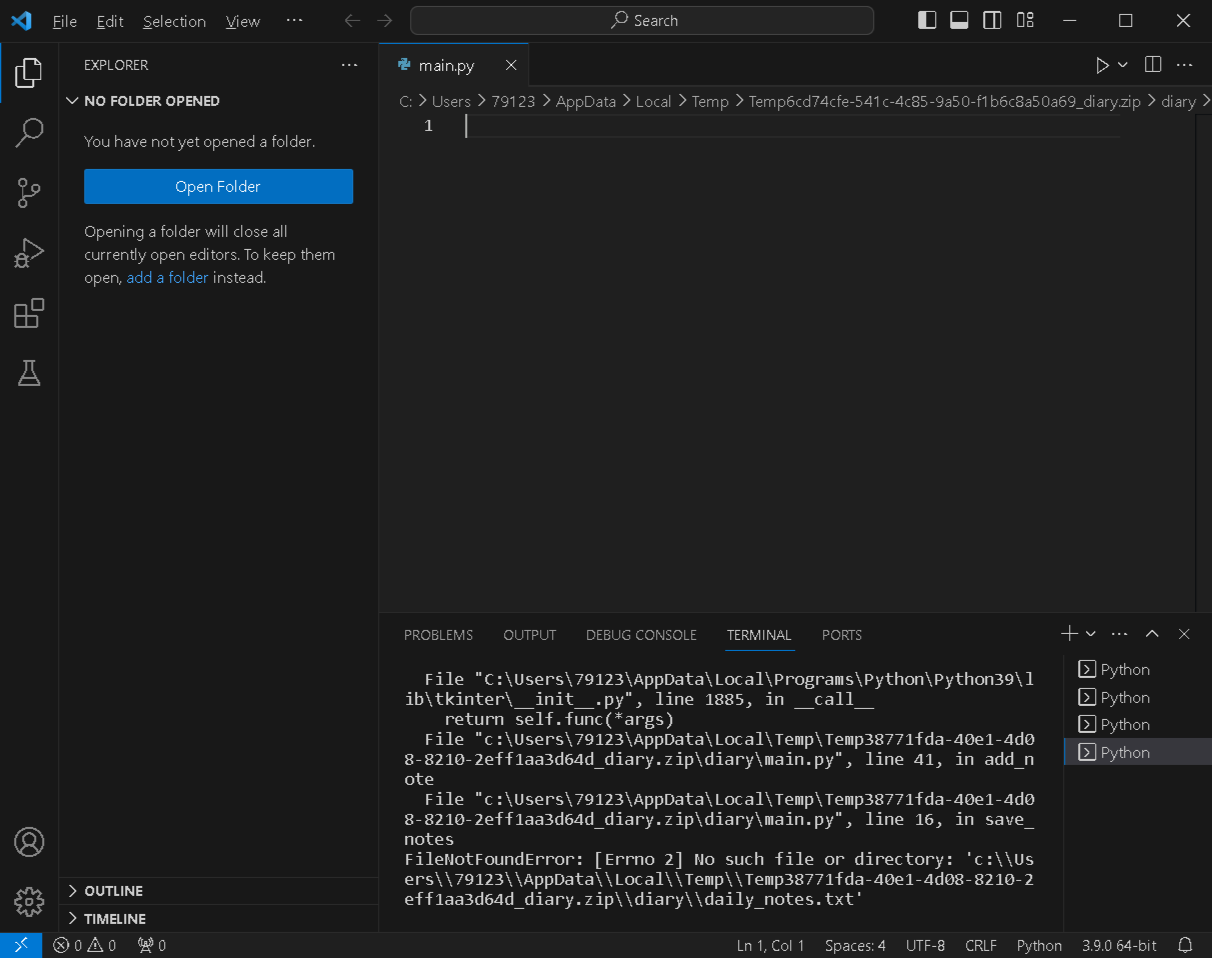


Рисунок 1 – Среда разработки Visual Studio Code.

Visual Studio Cоde – это один из наиболее популярных редакторов кода, разработанный корпорацией Micrоsoft. Он распространяется в бесплатном доступе и поддерживается всеми актуальными операционными системами: Windows, Linux и macOS. VS Code представляет собой обычный текстовый редактор с возможностью подключения различных плагинов, что дает возможность работать со всевозможными языками программирования для разработки любого IT-продукта.

# 3. ФОРМУЛИРОВКА ИНДИВИДУАЛЬНОГО ЗАДАНИЯ

В соответствии с индивидуальным заданием на учебную практику необходимо создать проект, используя любой язык программирования, описать разрабатываемое приложение, требования к функциям, внешнему виду, структурам данных, стадиям и этапам разработки. Реализовано приложение «Ежедневник».

Разработанное приложение «Ежедневник» является вспомогательным средством для ведения ежедневных записей в повседневной жизни человека. С помощью него пользователь может легко и доступно составлять списки покупок, планировать бюджет, записывать важные даты или яркие впечатления из жизни.

Это текстовый редактор, то есть приложение, специализирующееся на редактировании обычного текста. Он предлагает функции работы с текстом, такие как ввод, редактирование сохранение текста, выбор даты из календаря для сохранения заметки.

# 4. ОПИСАНИЕ ВЫПОЛНЕНИЯ ИНДИВИДУАЛЬНОГО ЗАДАНИЯ

## 4.1 Анализ предметной области и обзор аналогов

Ежедневник – одна из самых простых программ по работе с текстом. В ней мы можем вводить текст, редактировать его и сохранять в выбранную дату.

У электронных Ежедневников действительно не счесть преимуществ по сравнению с традиционными бумажными. Они доступны на любых устройствах — от мобильных гаджетов до персональных компьютеров, всегда под рукой и одинаково хорошо подходят как для текстовых записей, так и составления списков дел, записи сиюминутных мыслей и свежих идей. С их помощью можно удобно упорядочивать и структурировать информацию, быстро находить нужные заметки, планировать задачи и объединять все записи в одном приложении.

Приложение «Ежедневник» – приложение для создания, сортировки и редактирования заметок. Помимо написания текстовых записок, поддерживается внесение записей на определенные дни, изменение и удаление их.

На данный момент существуют следующие аналоги приложения «Ежедневник»:

Аналог №1

Программа «W[eeek](https://weeek.net/ru)«.

Ссылка :<https://weeek.net/ru>

Интерфейс приложения представлен на Рисунке 2.

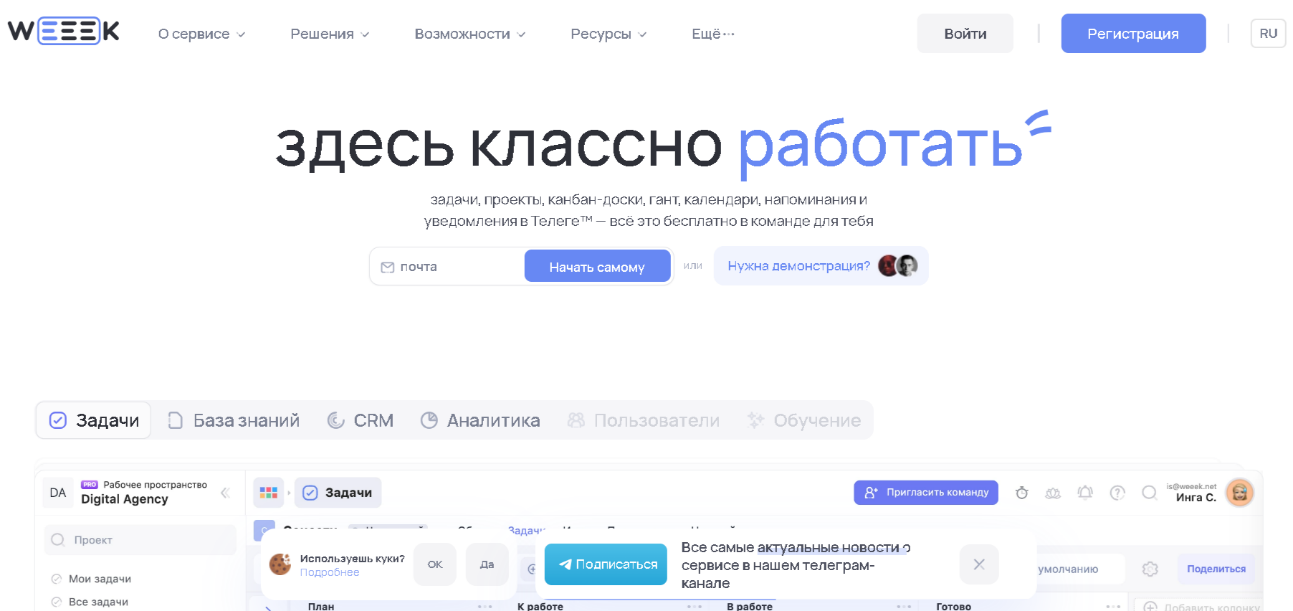


Рисунок 2 – Программа «WordPad»

**К достоинствам рассматриваемой программы можно отнести:**

* Российское производство
* Можно работать как в одиночку, так и целой командой
* На каждую задачу ставится дедлайн и таймер
* Получение гибких уведомлений в Telegram

**К недостаткам программы относятся:**

* Для адаптации все же потребуется немного времени, ведь надо разобраться во всех фичах и научиться не путаться в функционале. К счастью, команда WEEEK записывают бесплатные онлайн-инструкции и объясняет что к чему.
* Бесплатно с ограничениями, платно от 253р.

Аналог №2

Приложение «Todoist»

Ссылка: <https://todoist.com/auth/signup>

Интерфейс приложения представлен на Рисунке 3.

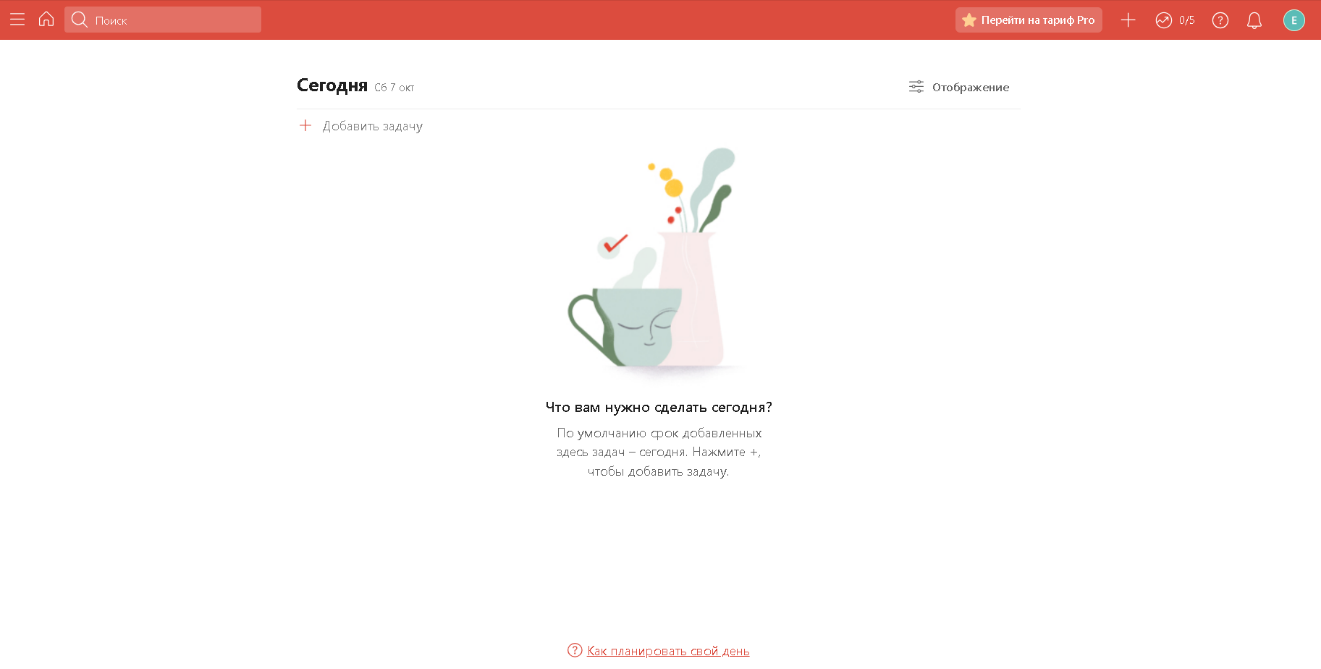


Рисунок 3 – Приложение «Evernote»

**К достоинствам рассматриваемой программы можно отнести:**

* Удобная система добавления дел.
* Потрясающий дизайн без излишеств, вдохновляющий на продуктивность.

**К недостаткам программы относятся:**

* + Возможные проблемы с оплатой в РФ, так как веб-сервис иностранный.
  + Невозможно создавать подзадачи.
  + До 5 проектов и до 5 человек — бесплатно.

Аналог №3

Приложение «Simplenote»

Ссылка: <https://simplenote.com/>

Интерфейс предоставлен на Рисунке 4.

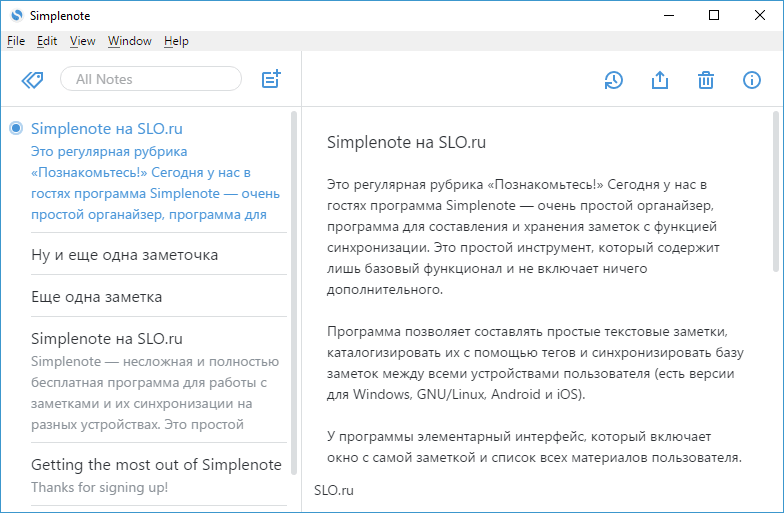


Рисунок 4 – Приложение «Simplenote»

**К достоинствам рассматриваемой программы можно отнести:**

* Симпатичный пользовательские интерфейсы, к которому быстро привыкаешь
* Поддержка на любых устройствах, включая смартфоны
* Сохранение заметок в облаке и доступ к ним с любого компьютера, ноутбука или телефона
* Низкие системные требования и отличное быстродействие

**К недостаткам программы относятся:**

* чувствительный к регистру поиск и невозможность открыть к заметке общий доступ для любого пользователя через сам Android клиент.

Аналог №4

Приложение «WorkFlowy».

Ссылка: <https://workflowy.com/>

Интерфейс предоставлен на Рисунке 5.

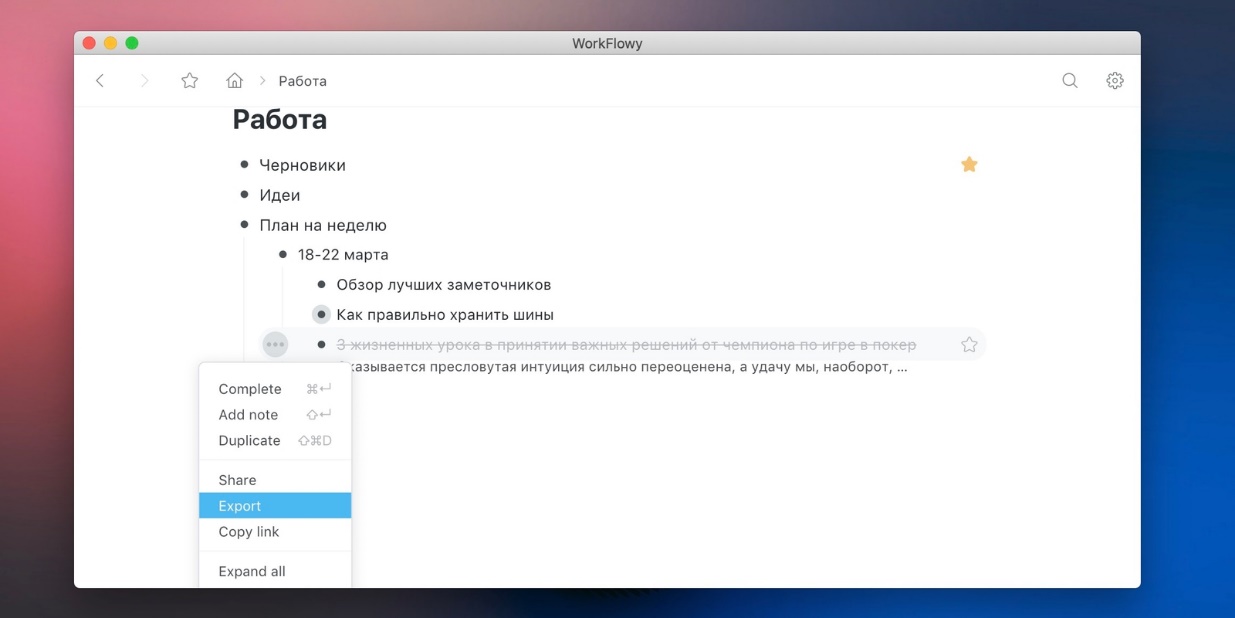


Рисунок 5 – Приложение «WorkFlowy»

**К достоинствам рассматриваемой программы можно отнести:**

* Интуитивно понятный интерфейс
* Синхронизируется в облако
* Есть версия для ПК, Android и iOS

**К недостаткам программы относятся:**

Нельзя добавлять изображения и файлы.

Из анализа аналогов можно сделать вывод, что необходимо разработать такое приложение, в котором будут реализованы следующие возможности:

* Понятный интерфейс;
* Занимает немного памяти;
* Низкие системные требования и отличное быстродействие
* Приложение должно быть на русском языке.

## 4.2 Разработка структуры приложения и алгоритмов функционирования.

Простой текстовый редактор с базовыми функциями, такими как открытие, сохранение и редактирование текста. Когда пользователь взаимодействует с элементами интерфейса, соответствующие действия выполняются на основе выбора в меню или других действий пользователя.

Программа состоит из следующих методов: загрузка заметок из словаря, создание окна, выбор даты из календаря, добавление заметок из текстового поля ввода в словарь для выбранной даты, вывод всех заметок для выбранной даты, удаление выбранной заметки из словаря, замена выбранной заметки на новую из текстового поля ввода в словарь, сохранение заметок в словарь. (Рисунок 6).

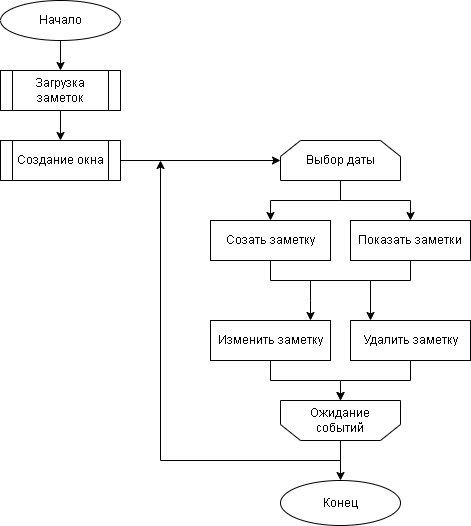


Рисунок 6 – Структура программы.

Схема алгоритма загрузки заметок и создания окна (Рисунок 7,8).

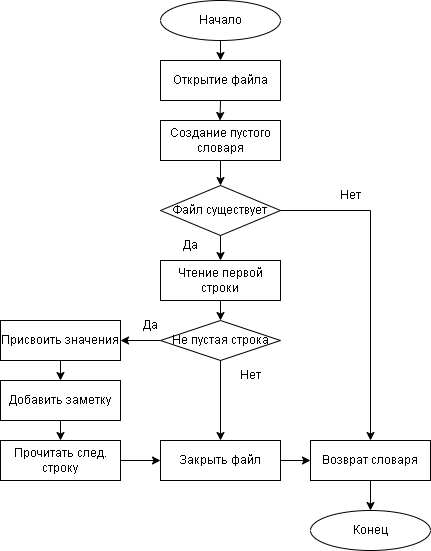


Рисунок 7 – Алгоритм загрузки заметок.

Эта блок-схема описывает процесс открытия файла, чтения строк из файла и сохранения заметок в словаре «daily\_notes». Заметки в файле хранятся в формате «дата:заметка», разделенные определенным разделителем. В блок-схеме также присутствуют проверки на существование файла и обработка каждой строки в файле, чтобы добавить заметки в словарь. В конце, функция возвращает словарь «daily\_notes».



Рисунок 8 – Алгоритм создания окна.

Эта блок-схема описывает основные шаги для создания окна с использованием графической библиотеки, например, Tkinter. Окно создается путем создания экземпляра основного окна и настройки его атрибутов, таких как размеры и заголовок. Затем в окне создаются и настраиваются различные виджеты, такие как кнопки, текстовые поля и Listbox. Виджеты размещаются в окне с использованием менеджера геометрии, такого как pack() или grid(). В конце, запускается основной цикл обработки событий для взаимодействия с пользователем.

Алгоритм выбора даты из календаря (Рисунок 9).



Рисунок 9 – Алгоритм выбора даты.

Эта блок-схема описывает базовые шаги для выбора даты из календаря. Пользователь открывает календарь, который отображает текущий месяц и год. Затем пользователь выбирает день из календаря. Выбранная дата может быть сохранена в переменной или передана в функцию обработки для дальнейшего использования. После выбора даты календарь закрывается, и выбор даты считается завершенным.

Алгоритм добавления и показа заметок (Рисунки 10,11).



Рисунок 10 – Алгоритм сохранения файла.

Эта блок-схема описывает базовые шаги для добавления заметки. Пользователь выбирает опцию «Добавить заметку» или просто начинает вводить текст заметки. Затем пользователь вводит текст заметки, который может быть сохранен в переменной или передан в функцию обработки для дальнейшей обработки. Заметка добавляется в список или хранилище заметок, где она может быть отображена на экране или сохранена в файл для последующего использования. После добавления заметки процесс считается завершенным.

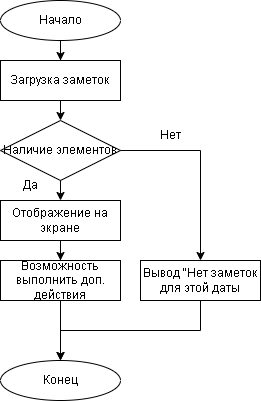


Рисунок 11– Алгоритм показа заметок.

Эта блок-схема описывает базовые шаги для показа заметок. Сначала происходит загрузка списка заметок из хранилища или файла. Затем происходит проверка списка на наличие элементов. Если список заметок пуст, процесс показа заметок завершается. Если в списке есть заметки, они отображаются на экране для просмотра пользователем. Пользователь может просмотреть заметки или выполнить дополнительные действия в зависимости от функциональности приложения или системы. После этого процесс показа заметок считается завершенным.

Алгоритмы изменения и удаления заметок (Рисунок 9).

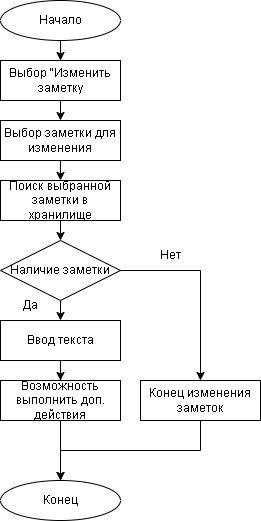


Рисунок 9 – Алгоритм изменения заметки.

Данная блок-схема описывает базовые шаги для изменения заметок. Пользователь выбирает опцию «Изменить заметку» или указывает идентификатор заметки, которую хочет изменить. Затем система ищет выбранную заметку в списке или хранилище заметок. Если заметка не найдена, процесс изменения заметок завершается. Если заметка найдена, пользователь вводит новый текст заметки или выбирает действия для изменения заметки (например, добавление/удаление содержимого). Заметка обновляется с учетом пользовательских изменений и сохраняется обратно в списке или хранилище заметок. После этого процесс изменения заметок считается завершенным.

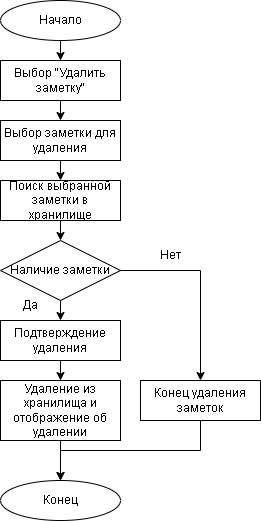


Рисунок 9 – Алгоритм удаления заметки.

Эта блок-схема описывает базовые шаги для удаления заметок. Пользователь выбирает опцию «Удалить заметку» или указывает идентификатор заметки, которую хочет удалить. Затем система ищет выбранную заметку в списке или хранилище заметок. Если заметка не найдена, процесс удаления заметок завершается. Если заметка найдена, система запрашивает подтверждение удаления заметки у пользователя. После подтверждения удаления, заметка удаляется из списка или хранилища заметок, и успешное подтверждение удаления отображается пользователю. После этого процесс удаления заметок считается завершенным.

## 

## 4.3 Реализация программы. Разработка пользовательского интерфейса

Для реализации программы был выбран язык программирования Python, использующий графический интерфейс Tkinter и календарь библиотеки tkcalendar для создания ежедневника. Пользователь может добавлять, просматривать, удалять и редактировать заметки. Заметки сохраняются в текстовый файл и загружаются из него при запуске программы. Исходный код представлен в приложении А.

Вот краткий обзор на код:

Глобальный словарь daily\_notes используется для хранения заметок, где каждая дата является ключом, связанным со списком заметок на эту дату.

add\_note, show\_notes, delete\_note и edit\_note - это функции обратного вызова, которые привязаны к соответствующим кнопкам для выполнения соответствующих задач.

save\_notes и load\_notes - это служебные функции для хранения и извлечения заметок в/ из daily\_notes.txt текстовый файл.

Виджет календаря используется для выбора даты, текстовые виджеты используются для ввода и редактирования заметок, поле списка используется для отображения и выбора заметок на определенную дату, а виджеты кнопок используются для запуска действий.

В конце вызывается load\_notes для загрузки существующих заметок из файла, и основной цикл обработки событий запускается root.mainloop().

Если вы хотите добавить какие-либо конкретные функции или улучшения, я был бы рад помочь советом или примерами кода. Например, вы могли бы добавить функцию «поиск» для поиска заметок, содержащих определенный текст, или реализовать функцию редактирования заметок на основе графического интерфейса пользователя.

**Инициализация компонентов и формы:**

* Создание корневого окна Tkinter;
* Создание и размещение календаря на форме;
* Создание надписи, поля для ввода текста и кнопки «Добавить заметку»;
* Создание надписи, списка заметок и кнопки «Удалить заметку»;
* Создание надписи, текстового поля и кнопки «Изменить заметку»;
* Загрузка заметок при запуске;
* Запуск главного цикла обработки событий окна;

Метод pack() используется для автоматической разметки элементов в окне. Кнопкам добавляются функции-обработчики событий, которые вызываются при нажатии кнопки. Каждая из этих функций (таких как add\_note, show\_notes, delete\_note, edit\_note) выполняет определенное действие с заметками или интерфейсом.

**Объявление глобальных переменных:**  
В программе объявлены две глобальные переменные - FILE\_PATH и daily\_notes , потому что они используются в различных функциях. Это позволяет каждой функции общаться с другими функциями и вместе совершать задачи работы с заметками.

**Обработчики событий:**

«add\_note()» - Обработчик для кнопки «Добавить заметку»:

- Этот обработчик вызывается при нажатии на кнопку «Добавить заметку».

- Сначала он получает выбранную дату из календаря и текст заметки из текстового поля «note\_entry».

- Затем проверяется, что текст заметки не пустой.

- Если текст не пустой, он добавляет заметку в словарь «daily\_notes» для выбранной даты.

- После добавления заметки, происходит очистка текстового поля «note\_entry», и выводится информационное окно об успешном добавлении записи.

- Затем вызывается «save\_notes()», чтобы сохранить обновленные записи.

2. «show\_notes()» - Обработчик для кнопки «Показать заметки»:

- Этот обработчик вызывается при нажатии на кнопку «Показать заметки».

- Он получает выбранную дату из календаря и отображает все заметки для этой даты в виджете «notes\_listbox».

3. «delete\_note()» - Обработчик для кнопки «Удалить заметку»:

- Этот обработчик вызывается при нажатии на кнопку «Удалить заметку».

- Он проверяет, есть ли выбранная заметка в списке «notes\_listbox».

- Если заметка выбрана, то она удаляется из словаря «daily\_notes», и после удаления обновляется отображение в «notes\_listbox».

- Затем вызывается «save\_notes()», чтобы сохранить обновленные записи.

4. «edit\_note()» - Обработчик для кнопки «Изменить заметку»:

- Этот обработчик вызывается при нажатии на кнопку «Изменить заметку».

- Он проверяет, есть ли выбранная заметка в списке «notes\_listbox».

- Если заметка выбрана, то он получает новый текст заметки из текстового поля «edit\_entry».

- Затем он обновляет выбранную заметку в словаре «daily\_notes» с новым текстом.

- После обновления заметки, обновляется отображение в «notes\_listbox».

- Затем вызывается «save\_notes()», чтобы сохранить обновленные записи.

**Реализация implementation:**  
Раздел implementation содержит реализацию всех идентификаторов и объявленных методов, исполняемых в программа, и объединяет все вышеперечисленные элементы в единую целостность.

**В программном коде реализуются следующие функции:**

**save\_notes()** (Сохранение заметок в файл).

* Функция save\_notes открывает daily\_notes.txt для записи. Если файл не существует, он будет создан. Если файл уже существует, его содержимое будет уничтожено, так как вы используете режим записи с обнулением ("w")
* Затем она перебирает каждую запись в daily\_notes. Для каждой записи, date будет датой, а notes - это список заметок за эту дату.
* Каждая запись записывается в файл в формате date::note1;note2;...;noteN. Это делается путем объединения списка заметок в одну строку с помощью ';'.join(notes). Затем дата и объединенная строка заметок объединяются с помощью f"{date}::{';'.join(notes)}\n" и записываются в файл. Перевод строки \n используется для создания новой строки для каждой записи.
* Файл автоматически закрывается после выхода из блока with...as. Это гарантирует, что даже в случае ошибки файл все равно закроется.

Исходный код функции представлен на рисунке 12.

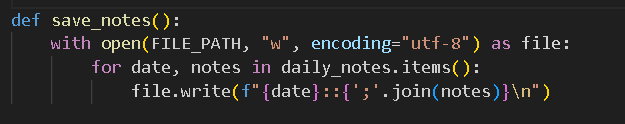


Рисунок 12 – Исходный код функции save\_notes().

**load\_notes()** (Функция для загрузки заметок из файла при запуске программы).

* Создается пустой словарь daily\_notes, в который будут загружены заметки из файла.
* Затем файл daily\_notes.txt открывается для чтения с помощью open и обрабатывается в блоке with...as file. Обработка файла происходит построчно.
* Каждая строка line разделяется на две части, используя разделитель "::" с помощью line.strip().split("::"). Затем проверяется, содержит ли строка ровно две части (дату и заметки).
* Если строка содержит две части, дата сохраняется в переменную date, а заметки разделяются с помощью разделителя ";" с помощью parts[1].split(";"). Затем создается запись в словаре daily\_notes с датой в качестве ключа и списком заметок в качестве значения.
* Если файл daily\_notes.txt не найден, функция возвращает пустой словарь daily\_notes. В противном случае возвращается словарь daily\_notes со всеми загруженными заметками.

Исходный код представлен на рисунке 13.

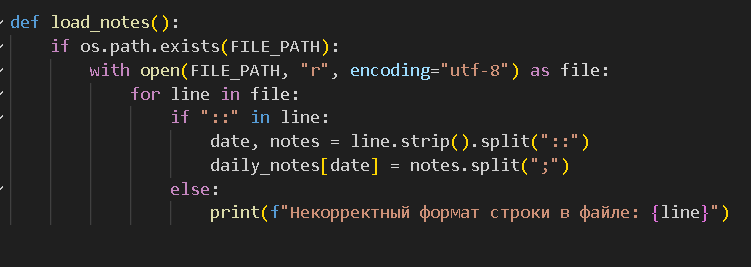


Рисунок 13 – Исходный код функции load\_notes().

**add\_note()** (Функция для добавления новой заметки).

* Запрашивается у пользователя ввод даты в формате "ГГГГ-ММ-ДД" и заметки.
* Затем мы получаем список заметок для указанной даты, используя метод get словаря daily\_notes. Если список не существует (дата встречается впервые), создается пустой список.
* Добавляем новую заметку в список с помощью метода append.
* Затем обновляем словарь daily\_notes, присваивая обновленный список заметок для указанной даты.
* Наконец, выводится сообщение о успешном добавлении заметки.

Этот код предполагает наличие словаря daily\_notes в программе, в котором будут храниться заметки. Вы можете вызвать функцию add\_note в своей программе, чтобы добавить заметки по мере необходимости. Обратите внимание, что функция add\_note не сохраняет заметки в файле. Мы должны вызывать функцию save\_notes, чтобы сохранить заметки в файл, после добавления.

Исходный код представлен на рисунке 14.

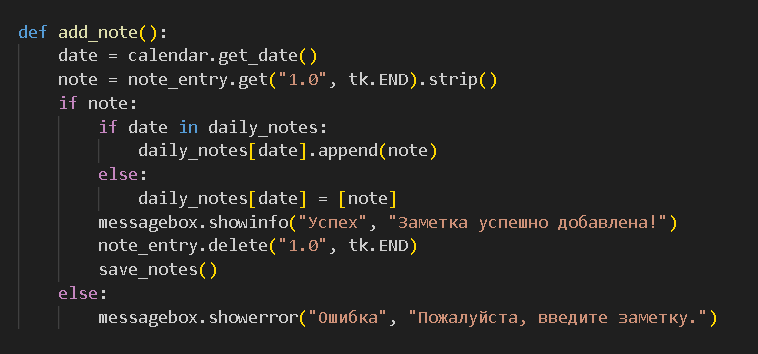


Рисунок 14 – Исходный код функции MenuItem9Click.

**show\_notes()** (Функция для отображения заметок для выбранной даты).

* Запрашивается у пользователя ввод даты в формате "ГГГГ-ММ-ДД" для отображения заметок.
* Затем мы получаем список заметок для указанной даты из словаря daily\_notes, используя метод get. Мы сохраняем этот список в переменную notes.
* Далее мы проверяем, есть ли заметки для указанной даты. Если список notes не пустой, то выводим заголовок с датой и каждую заметку на отдельной строке с помощью цикла for.
* Если список notes пустой, то выводится сообщение о том, что на указанную дату заметок нет.

Этот код предполагает наличие словаря daily\_notes с заметками в программе. Вы можете вызвать функцию show\_notes в своей программе, чтобы отобразить заметки для определенной даты.

Исходный код представлен на рисунке 15.

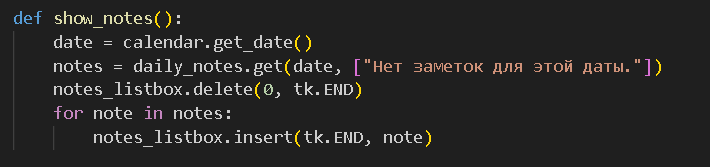
****

Рисунок 15 – Исходный код функции show\_notes().

**delete\_note()** (Функция для удаления выбранной заметки).

* Запрашивается у пользователя ввод даты в формате "ГГГГ-ММ-ДД" и заметки для удаления.
* Затем мы получаем список заметок для указанной даты из словаря daily\_notes, используя метод get. Мы сохраняем этот список в переменную notes.
* Далее мы проверяем, есть ли заметки для указанной даты. Если список notes не пустой, мы проверяем, есть ли указанная заметка в списке. Если заметка найдена, мы используем метод remove, чтобы удалить ее из списка.
* Если удаление успешно выполнено, выводится сообщение "Заметка успешно удалена".
* Если указанная заметка не найдена (отсутствует в списке заметок), выводится сообщение "Указанная заметка не найдена".
* Если список notes пустой, выводится сообщение о том, что на указанную дату заметок нет.

Этот код предполагает наличие словаря daily\_notes с заметками в программе. Вы можете вызвать функцию delete\_note в своей программе, чтобы удалить заданную заметку для определенной даты.

Исходный код представлен на рисунке 16.

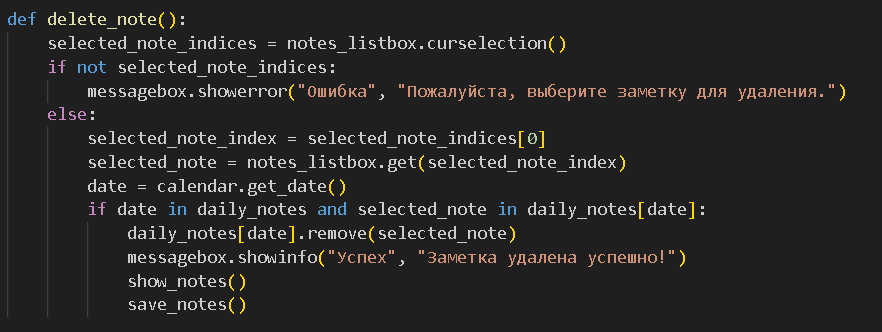
****

Рисунок 16 – Исходный код функции delete\_note().

**edit\_note()** (Функция для изменения выбранной заметки).

Запрашивается у пользователя ввод даты в формате "ГГГГ-ММ-ДД", заметки для редактирования и новой версии заметки.

Затем мы получаем список заметок для указанной даты из словаря daily\_notes, используя метод get. Мы сохраняем этот список в переменную notes.

Далее мы проверяем, есть ли заметки для указанной даты. Если список notes не пустой, мы проверяем, есть ли указанная заметка в списке. Если заметка найдена, мы получаем индекс этой заметки в списке с помощью метода index.

Затем мы заменяем заметку в списке на новую версию, присваивая значение переменной new\_note по индексу index.

Если изменение успешно выполнено, выводится сообщение "Заметка успешно изменена".

Если указанная заметка не найдена (отсутствует в списке заметок), выводится сообщение "Указанная заметка не найдена".

Если список notes пустой, выводится сообщение о том, что на указанную дату заметок нет.

Этот код предполагает наличие словаря daily\_notes с заметками в программе. Вы можете вызвать функцию edit\_note в своей программе, чтобы изменить заданную заметку для определенной даты.

Исходный код представлен на рисунке 16.

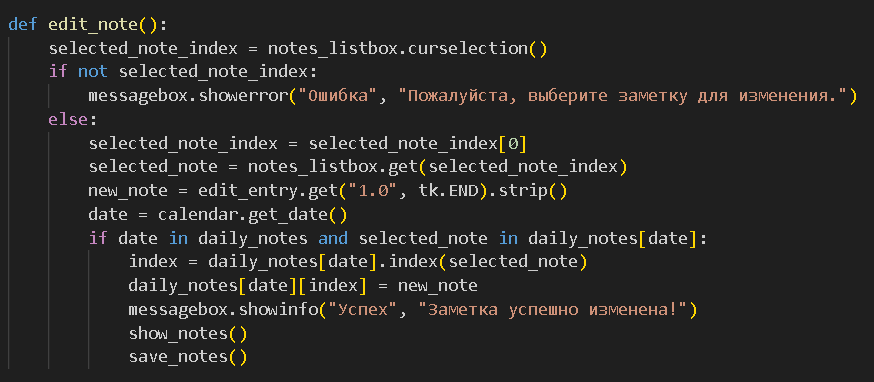
****

Рисунок 16 – Исходный код функции edit\_note().

**Разработка пользовательского интерфейса.**

На рисунке 17 представлен интерфейс приложения «Ежедневник».

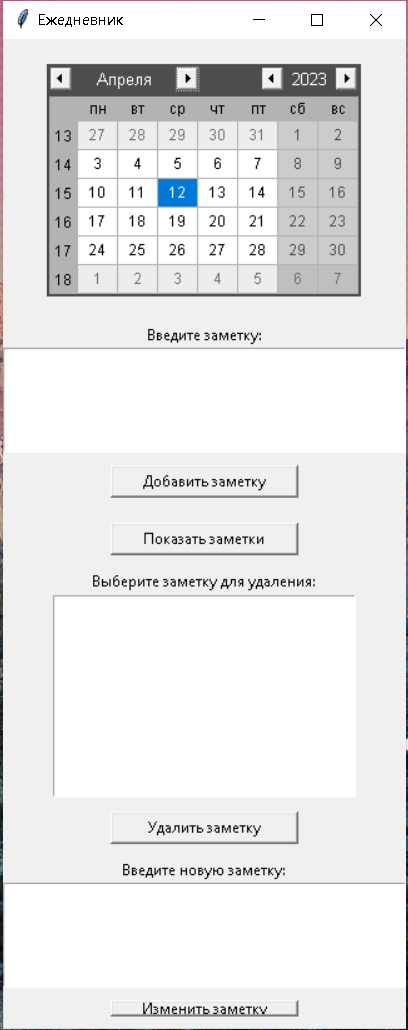


Рисунок 17– Интерфейс приложения. Главная форма.

Интерфейс создается с помощью библиотеки tkinter в Python. tkinter является стандартным интерфейсом Python для Tk GUI toolkit и представляет собой очень мощный и гибкий метод создания графического интерфейса пользователя (GUI).

Помимо этого, использован модуль tkcalendar для создания виджета календаря, позволяющего выбирать конкретные даты.

Вот некоторые основные виджеты и элементы:

* root = tk.Tk(): Создает главное окно приложения.
* calendar = Calendar(...): Создает виджет календаря.
* note\_label = tk.Label(...), note\_entry = tk.Text(...): Создаются Label и Text виджеты для ввода заметки пользователем.
* add\_button = tk.Button(...), show\_button = tk.Button(...), delete\_button = tk.Button(...), edit\_button = tk.Button(...): Создаются кнопки для различных функций: добавления, отображения, удаления и изменения заметок.
* notes\_listbox = tk.Listbox(...): Создается список (Listbox), где отображаются заметки.
* edit\_entry = tk.Text(...): Создается Text виджет для изменения заметки.
* root.mainloop(): Запускает основной цикл обработки событий Tkinter, что держит окно открытым и ждет взаимодействия пользователя.

Все эти элементы вместе образуют интерфейс ежедневника. При взаимодействии пользователя с интерфейсом (например, нажатии на кнопку) вызываются соответствующие функции.

Описание элементов на Главном окне:

* "Добавить заметку". Эта кнопка используется для добавления новой заметки для выбранной даты.

При нажатии на кнопку вызывается функция add\_note(), которая получает текст из поля ввода и добавляет его в список заметок для выбранной даты.

* "Показать заметки". Эта кнопка позволяет отобразить все заметки для выбранной даты.

При нажатии на кнопку вызывается функция show\_notes(), которая отображает все заметки для выбранной даты в виджете Listbox.

* "Удалить заметку" Эта кнопка используется для удаления выбранной заметки.

При нажатии на кнопку вызывается функция delete\_note(), которая удаляет выбранную заметку из списка заметок для выбранной даты.

* "Изменить заметку": Эта кнопка позволяет изменить выбранную заметку.

При нажатии на кнопку вызывается функция edit\_note(), которая отображает выбранную заметку в текстовом поле для редактирования и позволяет пользователю изменить текст.

Каждая из этих кнопок выполняет определенное действие, связанное с операциями добавления, отображения, удаления и изменения заметок.

## 

## 4.4 Тестирование программы.

Тестирование программы проведено в соответствии с Программой и методикой исследования, написанной в процессе разработки программного продукта.

Таблица 1 – Результаты испытаний

| № | Шаг испытаний (проверок) | № пункта  Методики | Отметка о прохождении (да/нет) | Примечания |
| --- | --- | --- | --- | --- |
| 1 | Проверка соответствия технического задания требованиям, предъявляемым к студентам направления подготовки по профильным дисциплинам. | 1 | Да |  |
| 2 | Проверка соответствия программы и методики тестирования требованиям, предъявляемым к студентам направления подготовки по профильным дисциплинам. | 1 | Да |  |
| 3 | Проверка соответствия руководства пользователя требованиям, предъявляемым к студентам направления подготовки по профильным дисциплинам. | 1 | Да |  |
| 6 | Проверка на корректность создания формы . | 2 | Да |  |
| 7 | Проверка на корректность активации кнопок. | 2 | Да |  |

Таблица 1 – Продолжение

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 8 | Проверка функции кнопки «Добавить заметку» на добавление новой заметки. | 2 | Да |  |
| 9 | Проверка функции кнопки «Показать заметки» на отображение заметок, уже внесенных в календарь. | 3 | Да |  |
| 10 | Проверка функции кнопки «Удалить заметку» на удаление выбранной заметки. | 4 | Да |  |
| 12 | Проверка функции кнопки «Изменить заметку» на изменение выбранной заметки. | 5 | Да |  |
| 13 | Проверка работы календаря для выбора даты. | 6 | Да |  |
| 14 | Проверка функции выбора необходимой заметки для изменения или удаления. | 7 | Да |  |
| 15 | Проверка формата сохранения формы. | 8 | Да |  |
| 16 | Проверка места сохранения формы. | 8 | Да |  |
| 17 | Проверка скорости генерации формы. | 9 | Да |  |
| 18 | Проверка скорости сохранения формы. | 9 | Да |  |
| 19 | Проверка эффективности изменения параметров настройки генерации. | 9 | Да |  |

Таблица 1 – Продолжение

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 25 | Проверка на возникновение системных ошибок. | 10 | Да |  |
| 26 | Проверка корректной работы игры при попытках вызвать системную ошибку. | 10 | Да |  |

# ЗАКЛЮЧЕНИЕ

Для создания пакета документации были изучены ГОСТы и требования к написанию технического задания, программы и методики тестирования, руководству пользователя. Благодаря этому я получила навыки по написанию документов.

В рамках индивидуального задания было разработано приложение «Ежедневник», в ходе которого были изучены различные ресурсы по программированию на языке Python, благодаря чему мы поняли как правильно и грамотно писать код, познакомились с различными полезными функциями. Так же для реализации приложения было необходимо создать графический интерфейс пользователя, для этого мы изучили разные библиотеки доступные на Python. Наш выбор пал на библиотеку Tkinter, при использовании которой был получен навык создания GUI.

Таким образом, программа учебной практики ПМ.05 в период с 09.01.2023 по 14.06.2023 была выполнена мной в полном объеме.

При выполнении работ, предусмотренных заданием на практику, в организации ФГБОУ ВО «Вятский государственный университет» мной, Щекотовой Екатериной Алексеевной, соблюдались правила внутреннего трудового распорядка, охраны труда, техники безопасности, противопожарной безопасности, санитарно-эпидемиологические правила и гигиенические нормативы. Пройден вводный инструктаж и инструктаж на рабочем месте. Замечаний не имела.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Учебник по Tkinter в Python – руководство и примеры // Помощник Python URL:<https://pythonpip.ru/osnovy/uchebnik-po-tkinter-v-python-rukovodstvo-i-primery> (дата обращения: 01.03.23).
2. Модуль random на примерах — Изучение методов генерации случайных данных // Python URL: <https://python-scripts.com/random> (дата обращения: 15.03.23).
3. Доусон М. Программируем на Python. – СПб.: Питер, 2014. – 416 с.
4. Обучение Python GUI (уроки по Tkinter) // PythonRu URL: <https://pythonru.com/uroki/obuchenie-python-gui-uroki-po-tkinter> (дата обращения: 03.03.23).
5. 13 GUI-библиотек Python // Techrocks.ru URL: https://techrocks.ru/2018/04/26/13-python-gui-frameworks/ (дата обращения: 01.03.23).

# ПРИЛОЖЕНИЕ А

**ИСХОДНЫЙ КОД**

import tkinter as tk

from tkinter import messagebox

from tkcalendar import Calendar

import os

current\_directory = os.path.dirname(os.path.abspath(\_\_file\_\_))

FILE\_PATH = os.path.join(current\_directory, «daily\_notes.txt»)

daily\_notes = {}

def save\_notes():

with open(FILE\_PATH, «w», encoding=«utf-8») as file:

for date, notes in daily\_notes.items():

file.write(f»{date}::{';'.join(notes)}\n»)

def load\_notes():

if os.path.exists(FILE\_PATH):

with open(FILE\_PATH, «r», encoding=«utf-8») as file:

for line in file:

if «::» in line:

date, notes = line.strip().split(«::»)

daily\_notes[date] = notes.split(«;»)

else:

print(f»Некорректный формат строки в файле: {line}»)

def add\_note():

date = calendar.get\_date()

note = note\_entry.get(«1.0», tk.END).strip()

if note:

if date in daily\_notes:

daily\_notes[date].append(note)

else:

daily\_notes[date] = [note]

messagebox.showinfo(«Успех», «Заметка успешно добавлена!»)

note\_entry.delete(«1.0», tk.END)

save\_notes()

else:

messagebox.showerror(«Ошибка», «Пожалуйста, введите заметку.»)

def show\_notes():

date = calendar.get\_date()

notes = daily\_notes.get(date, [«Нет заметок для этой даты.»])

notes\_listbox.delete(0, tk.END)

for note in notes:

notes\_listbox.insert(tk.END, note)

def delete\_note():

selected\_note\_indices = notes\_listbox.curselection()

if not selected\_note\_indices:

messagebox.showerror(«Ошибка», «Пожалуйста, выберите заметку для удаления.»)

else:

selected\_note\_index = selected\_note\_indices[0]

selected\_note = notes\_listbox.get(selected\_note\_index)

date = calendar.get\_date()

if date in daily\_notes and selected\_note in daily\_notes[date]:

daily\_notes[date].remove(selected\_note)

messagebox.showinfo(«Успех», «Заметка удалена успешно!»)

show\_notes()

save\_notes()

def edit\_note():

selected\_note\_index = notes\_listbox.curselection()

if not selected\_note\_index:

messagebox.showerror(«Ошибка», «Пожалуйста, выберите заметку для изменения.»)

else:

selected\_note\_index = selected\_note\_index[0]

selected\_note = notes\_listbox.get(selected\_note\_index)

new\_note = edit\_entry.get(«1.0», tk.END).strip()

date = calendar.get\_date()

if date in daily\_notes and selected\_note in daily\_notes[date]:

index = daily\_notes[date].index(selected\_note)

daily\_notes[date][index] = new\_note

messagebox.showinfo(«Успех», «Заметка успешно изменена!»)

show\_notes()

save\_notes()

root = tk.Tk()

root.title(«Ежедневник»)

calendar = Calendar(root, selectmode=«day», date\_pattern=«yyyy-mm-dd», locale='ru\_RU')

calendar.pack(padx=20, pady=20)

note\_label = tk.Label(root, text=«Введите заметку:»)

note\_label.pack()

note\_entry = tk.Text(root, height=5, width=40)

note\_entry.pack()

add\_button = tk.Button(root, text=«Добавить заметку», command=add\_note, width=20)

add\_button.pack(pady=10)

show\_button = tk.Button(root, text=«Показать заметки», command=show\_notes, width=20)

show\_button.pack(pady=10)

delete\_label = tk.Label(root, text=«Выберите заметку для удаления:»)

delete\_label.pack()

notes\_listbox = tk.Listbox(root, height=10, width=40)

notes\_listbox.pack()

delete\_button = tk.Button(root, text=«Удалить заметку», command=delete\_note, width=20)

delete\_button.pack(pady=10)

edit\_label = tk.Label(root, text=«Введите новую заметку:»)

edit\_label.pack()

edit\_entry = tk.Text(root, height=5, width=40)

edit\_entry.pack()

edit\_button = tk.Button(root, text=«Изменить заметку», command=edit\_note, width=20)

edit\_button.pack(pady=10)

# Загрузка заметок из файла при запуске

load\_notes()

root.mainloop()

# ПРИЛОЖЕНИЕ Б

**ТЕХНИЧЕСКОЕ ЗАДАНИЕ**

# ПРИЛОЖЕНИЕ В

**ПРОГРАММА И МЕТОДИКА ИССЛЕДОВАНИЯ**

# ПРИЛОЖЕНИЕ Г

**РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ**