

Министерство высшего образования и науки Российской Федерации
Пензенский государственный университет
Кафедра «Вычислительная техника»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе

по курсу «Программирование»

на тему «Разработка программы сложной структуры методом
нисходящего программирования. Программа «Планер».

*отметки
27.05.2026
Г. Генералов*

Выполнил:

студент группы 25BBB3

Иванова С. А.

Принял:

к.н.т., доцент

Генералова А.А.

Пенза, 2026

ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
Факультет Вычислительной техники
Кафедра "Вычислительная техника"

"УТВЕРЖДАЮ"

Зав. кафедрой ВТ

М.А. Митрохин

« 19 » Февраля 2026г

ЗАДАНИЕ

на курсовое проектирование по курсу

Программирование

Студенту Ивановой Снежане Алексеевне Группа 25 БВВ 3

Тема проекта: **Разработка программы сложной структуры методом нисходящего программирования.**

Исходные данные (технические требования) на проектирование

Программа "Плакер". Программа представляет собой графический интерфейс. Временное время и составляющие плама вводятся пользователем. В файл сохраняются данные на текущую неделю.

Обязательные требования к программе:

1. Многомодульность.
2. Использование сложных типов данных – структур, списков, файлов.
3. Режим работы видеосистемы – текстовый / графический.
4. Устройство ввода информации – клавиатура, мышь.
5. Пользовательский интерфейс должен быть построен на основе меню и панели инструментов.
6. Наличие заставки.
7. Операционная система MS Windows.
8. Язык программирования – Си и Ассемблер.
9. Среда разработки ПО – Microsoft Visual Studio.

Объем работы по курсу

1. Расчетная часть

Разработка программы.

2. Графическая часть

Схема данных, схема ресурсов системы, схема работы системы, иерархическая структура программы, схема взаимодействия программы.

3. Экспериментальная часть

Отладка программы.

Срок выполнения проекта по разделам

В соответствии с графиком.

Дата выдачи задания "19" февраля

Дата защиты проекта "27" мая 2026

Руководитель Генералов И.А.

Задание получил "19" февраля 2026г.

Студент Иванов С.А.

Содержание

Введение	5
Постановка задачи.....	6
Выбор решения.....	7
Описание разработки программы	9
Отладка и тестирование	11
Описание программы.....	12
Руководство пользователя	24
Заключение	31
Список используемых источников.....	32
Приложение А Листинги программы	33
Приложение В Основные интерфейсы программы.....	57

Введение

В современном мире информация играет ключевую роль в организации деятельности человека и общества в целом. Компьютерные технологии значительно упрощают процессы обработки, хранения и систематизации данных, что позволяет повысить эффективность работы в самых разных сферах. Одной из востребованных задач является управление личным временем – планирование дел, распределение задач и контроль их выполнения. Программные средства для ведения расписания, или планировщики, помогают пользователям структурировать свою активность, снижая риск забыть важные события и оптимизируя использование временных ресурсов.

История развития планировщиков уходит корнями в эпоху появления персональных компьютеров. Первые электронные календари и органайзеры, появившиеся в 1980-х годах, представляли собой простые текстовые формы для ввода заметок. С развитием графических интерфейсов и операционных систем планировщики обрели более удобные формы: таблицы, напоминания, возможность сохранения данных в файлы.

Данная курсовая работа посвящена разработке приложения «Планер», которое позволяет пользователю создавать расписание на неделю с разбивкой по дням и временным интервалам, а также сохранять и загружать планы в текстовые файлы. Программа реализована на языке C# с использованием Windows Forms [1] и включает в себя ассемблерную вставку для шифрования сохраняемых данных, что демонстрирует взаимодействие высокоуровневого и низкоуровневого кода.

Постановка задачи

Необходимо разработать программу «Планер» – приложение с графическим интерфейсом, в котором пользователь ведёт ежедневное расписание на неделю, имея возможность создавать, редактировать, сохранять и загружать планы. Программа должна обладать интуитивно понятным интерфейсом, включающим главное меню с пунктами для создания нового плана, открытия существующего, переключения между светлой и тёмной цветовыми темами, а также редактор плана, содержащий семь таблиц для дней недели (понедельник, вторник, среда, четверг, пятница, суббота, воскресенье) и отдельное многострочное текстовое поле для заметок. Каждая таблица должна состоять из двух колонок: «Время» и «Задача» с возможностью пользователю добавлять и удалять строки. Необходимо реализовать многострочный режим ввода в ячейках с автоматическим растягиванием высоты строки под содержимое.

Программа должна быть поделена на логические блоки: главное меню, редактор плана, модули цветовых тем (светлая и тёмная), внешний модуль шифрования на ассемблере. Это упростит поиск ошибок при отладке и тестировании, а также позволит легко расширять функционал. Использование сложных типов данных (массивов для хранения ссылок на таблицы, списков строк в таблицах, файлов для сохранения и загрузки планов) необходимо для более простой и интуитивной обработки данных в коде программы. Графический интерфейс будет представлен на основе Windows Forms [2].

Устройства ввода-вывода – клавиатура и мышь. Необходимо различать и идентифицировать действия, производимые с их помощью: клики по кнопкам и пунктам меню, двойной клик для выбора файла в списке, использование клавиши Delete для удаления строк. Также требуется реализовать шифрование текстовых файлов планов методом XOR с использованием отдельной консольной программы на C++, содержащей ассемблерную вставку [3].

Выбор решения

При запуске программы открывается главное меню, управление производится с помощью мыши.

1. Главное меню включает следующие пункты:

- 1.1. «Создать план» – инициирует процедуру создания нового плана.
- 1.2. «Открыть план» – отображает перечень существующих планов для выбора.
- 1.3. «Сменить тему» – переключает цветовое оформление интерфейса между светлым и тёмным.

2. Создание плана. После выбора пункта «Создать план» главное меню скрывается, и пользователю предоставляется форма для ввода названия создаваемого плана. Форма содержит текстовое поле с поясняющей надписью «Введите название плана», а также кнопки «Далее» и «Назад». Программа осуществляет проверку наличия файла с указанным именем. При обнаружении существующего файла выводится диалоговое окно с запросом: «План с таким именем уже существует. Открыть его?». При положительном ответе открывается редактор плана с загрузкой содержимого данного файла. При отрицательном ответе пользователь возвращается к форме ввода имени. Если файл с указанным именем отсутствует, программа автоматически создаёт новый пустой план и переходит к редактору.

3. Открытие плана. При выборе пункта «Открыть план» главное меню скрывается, а вместо него отображается панель со списком всех ранее сохранённых планов. Пользователь видит перечень файлов, находящихся в папке хранения планов. Для открытия нужного плана необходимо выполнить двойное нажатие левой клавиши мыши на соответствующем названии, после этого сразу загружается редактор плана с данными выбранного файла. Кнопка «Назад» закрывает панель выбора и возвращает главное меню.

4. Редактирование плана. Редактор плана является основным рабочим окном приложения. Вся рабочая область равномерно разделена на восемь

частей, образующих сетку. В первых семи ячейках этой сетки расположены таблицы, каждая из которых соответствует одному дню недели: понедельник, вторник, среда, четверг, пятница, суббота.

Каждая таблица содержит две колонки: «Время» и «Задача». В колонке «Время» по умолчанию указаны временные интервалы с 8:00 до 18:00. Пользователь может изменить любое значение времени по своему усмотрению. В колонке «Задача» вводится текст планируемого дела. Для добавления новой строки достаточно начать ввод в последней пустой строке таблицы. Удаление строки выполняется после её выделения с последующим нажатием клавиши Delete

Восьмая ячейка сетки содержит многострочное текстовое поле с заголовком «Заметки».

В нижней части окна редактора расположены три кнопки:

4.1. «Сохранить изменения» – записывает содержимое таблиц и поля заметок в текущий файл. Перед записью данные шифруются на ассемблере. После успешного сохранения выводится соответствующее сообщение.

4.2. «Удалить план» – удаляет текущий файл плана после подтверждения пользователем. После удаления редактор закрывается.

4.3. «Главное меню» – возвращает пользователя в главное меню. Если с момента последнего сохранения были внесены изменения, программа предупреждает об этом и запрашивает подтверждение.

5. Обработка ошибок и подтверждение действий. При удалении плана, выходе без сохранения или возникновении ошибок (невозможность открыть или записать файл) программа выводит диалоговое окно с понятным сообщением и кнопками выбора. Пользователь всегда может подтвердить или отменить действие.

6. Переключение цветовых тем. В главном меню присутствует кнопка «Сменить тему». Светлая тема (светлый фон, тёмный текст, зелёные кнопки) меняется на тёмную (тёмный фон, белый текст, серые кнопки) и наоборот.

Описание разработки программы

Разработанная программа состоит из нескольких модулей, которые реализовывались в следующем порядке:

1. Form1.cs – главное меню программы.
 - 1.1. Form1() – конструктор, инициализация компонентов и применение светлой темы.
 - 1.2. Theme_Click() – переключение между светлой и тёмной темами.
 - 1.3. NewPlan_Click() – отображение панели ввода названия нового плана.
 - 1.4. OpenPlan_Click() – вызов LoadFiles() и отображение панели выбора существующего плана.
 - 1.5. LoadFiles() – чтение списка файлов из папки Plans, заполнение ListBox.
 - 1.6. ShowPanel() – управление видимостью панели выбора файлов.
 - 1.7. listBox1_MouseDoubleClick() – открытие выбранного плана, запуск PlanerForm.
 - 1.8. Back_Click() – возврат в главное меню.
 - 1.9. button2_Click() – обработка ввода названия плана, создание или открытие файла.
 - 1.10. button1_Click() – отмена создания плана.
 - 1.11. CreateEmptyPlanFile() – формирование пустого файла плана с начальной структурой (дни, время, заметки).
 - 1.12. HideCreationControls() и ShowMainMenu() – скрытие/отображение элементов.
2. PlanerForm.cs – форма редактора плана.
 - 2.1. PlanerForm() – конструктор, загрузка данных, настройка таблиц, подписка на события изменения размера, применение темы.
 - 2.2. ApplyTheme() – применение выбранной цветовой темы.
 - 2.3. SetupDataGridViews() – инициализация семи таблиц (DataGridView): установка колонок «Время» и «Задача», разрешение добавления/удаления

строк, многострочный режим с авто-высотой, подписка на события изменения данных.

2.4. `FillDefaultTimes()` – заполнение таблиц временными интервалами (08:00, 10:00, 12:00, 14:00, 16:00, 18:00).

2.5. `LoadFromFile()` – расшифровка (вызов `EncryptDecryptFile`) и чтение данных из временного файла, парсинг строк, восстановление переносов строк, заполнение таблиц и поля заметок.

2.6. `SaveToFile()` – запись данных всех таблиц и заметок во временный файл, копирование в целевой файл, шифрование вызовом `EncryptDecryptFile`.

2.7. `SaveDayToFile()` – запись одного дня в файл в формате «время|задача», экранирование переносов строк.

2.8. `GetTableByDayName()` и `GetAllTables()` – вспомогательные методы для доступа к таблицам.

2.9. `Save_Click()` – сохранение плана с выводом сообщения.

2.10. `Exit_Click()` – закрытие редактора с проверкой несохранённых изменений.

2.11. `PlanerForm_Resize()` и `UpdateFonts()` – динамическое изменение шрифтов и размеров при максимизации окна.

2.12. `btnDeletePlan_Click()` – удаление текущего файла плана после подтверждения.

2.13. `EncryptDecryptFile()` – запуск внешнего шифровальщика `XORCrypter.exe` с передачей пути к файлу.

3. Модули цветовых тем (`LightTheme.cs`, `DarkTheme.cs`).

3.1. `LightTheme.ApplyToAll()` и `DarkTheme.ApplyToAll()` – рекурсивный обход всех элементов управления формы и установка цветов фона, текста, стилей кнопок, таблиц, текстовых полей, вкладок.

3.2. `ApplyToControl()` – внутренняя рекурсивная функция, определяющая тип элемента и применяющая соответствующие цветовые настройки.

4. Внешний модуль шифрования (XORCrypter.exe) – консольное приложение на C++ с ассемблерной вставкой.

4.1. Функция `xor_crypt()` с ассемблерной вставкой `__asm`, выполняющая побайтовое XOR с ключом 0xAA.

4.2. Основная функция `main()` – чтение аргумента командной строки (путь к файлу), открытие файла в бинарном режиме, чтение всего содержимого в буфер, вызов `xor_crypt()`, запись буфера обратно в файл.

4.3. Проект скомпилирован для платформы x86 (требование для работы `__asm`).

5. Папка и файлы данных (Plans*.txt) – автоматически создаётся при первом сохранении; файлы хранятся в зашифрованном виде.

По приведённому порядку создания модулей видно, что в работе применялось нисходящее программирование. Суть этого метода заключается в последовательной детализации: сначала прорабатываются крупные блоки программы, а затем добавляются вспомогательные возможности. Благодаря такому подходу каждый функциональный элемент можно проверить сразу после его написания, не дожидаясь завершения всей программы.

Отладка и тестирование

В качестве среды разработки была выбрана программа Microsoft Visual Studio 2026 [4]. Данная среда обладает всеми средствами, необходимыми для разработки и отладки программы. Для отладки использовались возможности Visual Studio: точки останова, трассировка, анализ содержимого переменных, а также просмотр состояния регистров при отладке ассемблерного кода.

Тестирование проводилось во время разработки и после её завершения. В ходе тестирования серьёзных проблем выявлено не было. Все модули программы работают корректно: главное меню правильно обрабатывает ввод пользователя, редактор плана позволяет добавлять, изменять и удалять задачи, файлы успешно сохраняются с шифрованием и загружаются обратно,

цветовые темы переключаются без ошибок. Ассемблерная вставка в программе XORCrypter.exe работает стабильно, обеспечивая побайтовое XOR-шифрование файлов. Программа прошла проверку в различных сценариях использования и показала себя надёжной и устойчивой.

Описание программы

В таблице 1 представлено описание работы главного меню приложения. Данное меню отображается сразу после запуска программы и содержит четыре основные кнопки: «Создать план», «Открыть план» и «Сменить тему». Все элементы управления имеют крупный размер и чёткие подписи. Навигация осуществляется исключительно с помощью мыши: пользователь наводит курсор на нужную кнопку и выполняет однократное нажатие левой кнопкой. Клавиатурные комбинации не используются для обеспечения максимальной интуитивности интерфейса.

Таблица 1 – Главное меню

Клавиши и действия, вызывающие событие	Действие пользователя	Действие программы
Клик левой кнопкой мыши	Нажатие кнопки «Создать план»	Переход к окну ввода имени
Клик левой кнопкой мыши	Нажатие кнопки «Открыть план»	Показ списка файлов из папки Plans
Клик левой кнопкой мыши	Нажатие кнопки «Сменить тему»	Переключение цветовой темы (светлая/тёмная)

В таблице 2 описан режим создания нового плана. После выбора пункта «Создать план» главное меню скрывается, и пользователю предоставляется форма с текстовым полем, подписью «Введите название плана», а также кнопками «Далее» и «Назад». Ввод имени осуществляется с клавиатуры с

поддержкой букв, цифр, пробелов и знаков подчёркивания. Программа автоматически добавляет расширение .txt. При попытке сохранения с пустым полем выводится предупреждение. Если файл с указанным именем уже существует в папке Plans, система запрашивает у пользователя подтверждение на открытие существующего плана вместо создания нового.

Таблица 2 – Режим создания нового плана

Клавиши и действия, вызывающие событие	Действие пользователя	Действие программы
Любая буквенная/цифровая клавиша, пробел, подчёркивание	Ввод текста в поле	Запоминание введённого имени
Клик левой кнопкой мыши	Нажатие кнопки «Далее»	Проверка имени: если пустое, то ошибка; если файл есть, то предложение открыть; если нет, создание пустого плана.
Клик левой кнопкой мыши	Нажатие кнопки «Назад»	Возврат в главное меню без сохранения

В таблице 3 представлен режим выбора существующего плана. При активации пункта «Открыть план» на экране отображается панель с прокручиваемым списком имён всех доступных файлов из директории Plans (расширение .txt скрыто). В случае отсутствия сохранённых планов в списке выводится сообщение «(Нет планов)». Для загрузки расписания необходимо выполнить двойной щелчок левой кнопкой мыши по соответствующей строке. Кнопка «Назад» обеспечивает возврат в главное.

Таблица 3 – Режим выбора существующего плана

Клавиши и действия, вызывающие событие	Действие пользователя	Действие программы
Двойной клик левой кнопкой мыши	Выбор строки в списке	Загрузка выбранного плана, открытие редактора
Клик левой кнопкой мыши	Нажатие кнопки «Назад»	Возврат в главное меню

В таблице 4 приведено описание работы редактора плана, являющегося основным окном приложения. Рабочая область структурирована в виде сетки и разделена на восемь функциональных блоков. Первые семь блоков содержат таблицы для каждого дня недели (с понедельника по воскресенье). Каждая таблица включает две колонки: «Время» (с предустановленными интервалами 08:00–18:00) и «Задача».

Пользователь может редактировать временные метки, вносить задачи, добавлять новые строки путём ввода в последнюю пустую ячейку и удалять строки с помощью клавиши Delete после их выделения. Длинный текст автоматически переносится, а высота строк динамически подстраивается под содержимое. Восьмой блок отведён под многострочное текстовое поле «Заметки» с вертикальной прокруткой. В нижней части формы расположены кнопки управления: «Сохранить изменения», «Удалить план» и «Главное меню».

Таблица 4 – Редактор плана

Клавиши и действия, вызывающие событие	Действие пользователя	Действие программы
Клик левой кнопкой мыши	Выбор ячейки в таблице	Переход в режим редактирования

Продолжение таблицы 4

Любая печатная клавиша, пробел, Enter (перенос строки)	Набор текста в ячейке	Отображение текста, перенос, авто-высота строки
Delete	Предварительное выделение строки	Удаление выделенной строки
Начало печати в последней пустой строке	—	Автоматическое добавление новой строки внизу
Любая печатная клавиша	Ввод текста в поле «Заметки»	Сохранение текста заметок
Клик левой кнопкой мыши	Нажатие кнопки «Сохранить изменения»	Шифрование (XORCrypter.exe) и запись в файл
Клик левой кнопкой мыши	Нажатие кнопки «Удалить план»	Диалог подтверждения, удаление файла, закрытие окна
Клик левой кнопкой мыши	Нажатие кнопки «Гляньте меню»	Если есть изменения, предупреждение; затем закрытие и возврат в меню
Перетаскивание края окна (мышью)	Изменение размера окна	Автоматическая подгонка шрифтов, высоты строк, ширины колонок
Нажатие на крестик окна	Закрытие окна	Если есть изменения, запрос подтверждения, иначе закрытие

В таблице 5 описан процесс шифрования и расшифровки данных. При операциях сохранения и загрузки программа в фоновом режиме автоматически запускает внешнюю консольную утилиту XORCrypter.exe,

разработанную на C++ с использованием ассемблерной вставки (`__asm`). Для защиты данных применяется алгоритм побайтового XOR с фиксированным ключом 0xAA. Благодаря симметричности данного алгоритма одна и та же утилита выполняет как шифрование при записи, так и расшифровку при чтении файлов. Взаимодействие с консольным приложением происходит скрыто от пользователя, окно консоли не отображается на экране.

Таблица 5 – Шифрование

Клавиши и действия, вызывающие событие	Действие пользователя	Действие программы
—	Нажатие «Сохранить изменения»	Запуск XORCrypter.exe для временного файла, шифрование, замена целевого файла
—	Выбор плана через двойной клик	Копирование файла, расшифровка копии, чтение данных

На рисунке 1 изображена блок-схема общей работы приложения «Планер». После запуска программы отображается заставка, затем открывается главное меню. Программа ожидает выбора пользователя. При нажатии на кнопку «Создать план» выполняется переход к форме редактора плана с созданием нового файла. При нажатии на кнопку «Открыть план» происходит загрузка списка файлов из папки Plans и переход к редактору после выбора файла. При нажатии на кнопку «Сменить тему» происходит переключение цветовой гаммы интерфейса между светлой и тёмной. При нажатии на кнопку «Выход» или закрытии окна программа завершается.

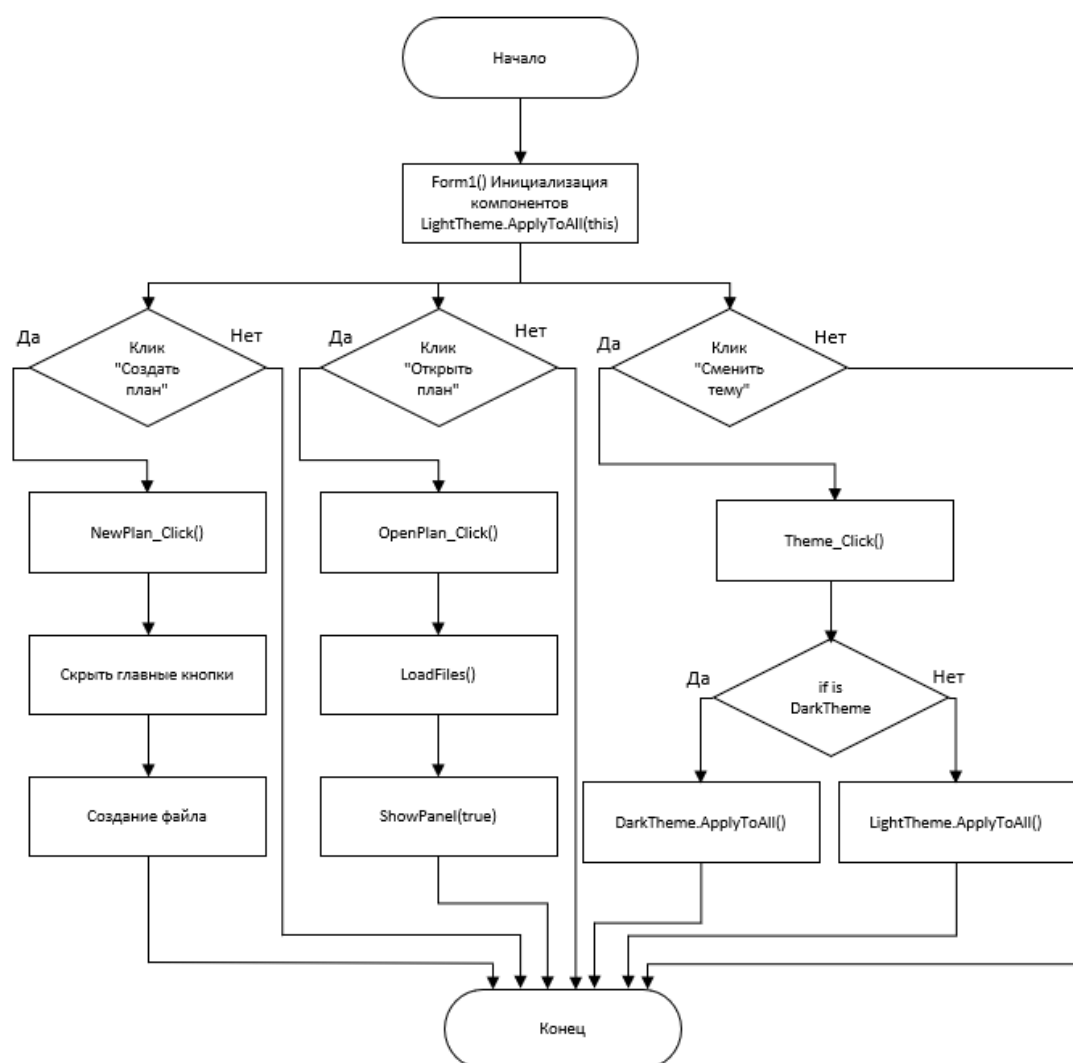


Рисунок 1 – Общая блок-схема работы программы

На рисунке 2 представлена блок-схема создания нового плана (метод `button2_Click`). Сначала программа получает текст из поля ввода и проверяет, не пустая ли строка. Если строка пуста, выводится сообщение об ошибке, и пользователь возвращается к вводу. Если строка не пуста, формируется путь к файлу в папке `Plans`. Далее проверяется существование файла с таким именем. Если файл уже существует, программа запрашивает подтверждение на его открытие. При согласии открывается редактор плана с существующим файлом, при отказе пользователь возвращается к вводу имени. Если файл не существует, создаётся пустой файл с предопределённой структурой, после чего открывается редактор плана. После закрытия редактора главное меню снова становится видимым.

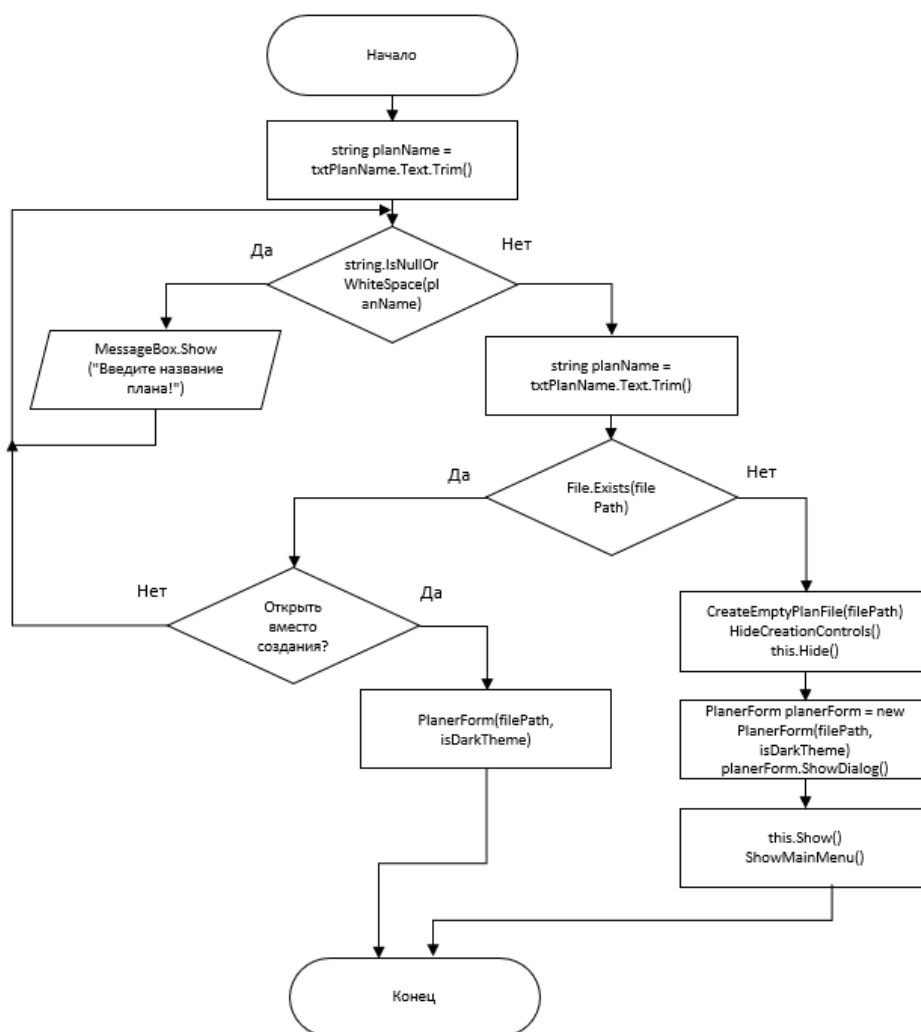


Рисунок 2 – Блок-схема создания нового плана

На рисунке 3 изображена блок-схема открытия существующего плана (методы `OpenPlan_Click` и `listBox1_MouseDoubleClick`). При выборе пункта «Открыть план» программа загружает список всех текстовых файлов из папки `Plans` и отображает панель выбора. Программа ожидает действия пользователя. При двойном клике по названию файла программа скрывает главное меню, создаёт форму редактора плана с передачей пути к выбранному файлу и открывает её. После закрытия редактора главное меню снова отображается, а панель выбора скрывается. При нажатии на кнопку «Назад» панель выбора скрывается, и пользователь возвращается в главное меню.

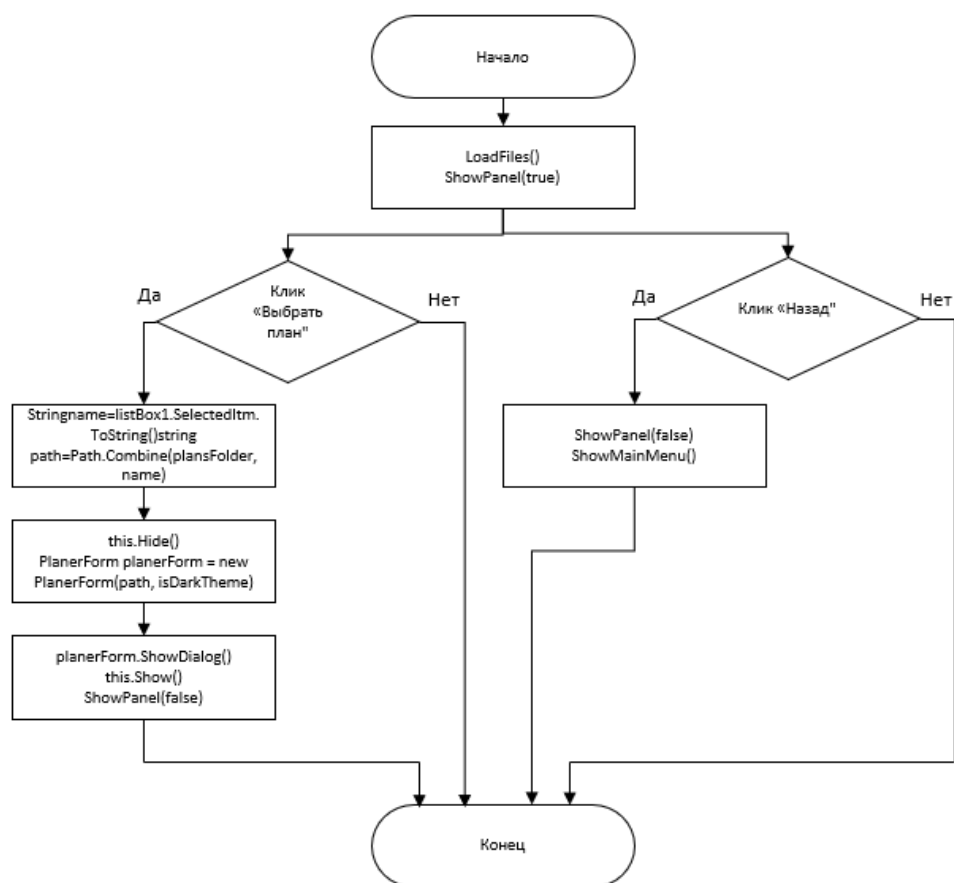


Рисунок 3 – Блок-схема открытия существующего плана

На рисунке 4 представлена блок-схема загрузки данных из файла и их расшифровки (метод LoadFromFile в PlanerForm). Сначала проверяется существование файла. Если файл не найден, выполнение прекращается. Если файл существует, создаётся временный файл, в него копируется содержимое исходного файла, после чего вызывается внешняя программа XORCrypter.exe для расшифровки (побайтовое XOR). Далее программа очищает все таблицы и построчно читает расшифрованный временный файл. При обнаружении строки с маркером дня (начинается с «---») определяется соответствующая таблица. При обнаружении строки с разделителем «|» строка разбивается на время и задачу, восстанавливаются переносы строк, и новая запись добавляется в таблицу. При обнаружении маркера «=== ЗАМЕТКИ ===» весь оставшийся текст загружается в поле заметок. После чтения временный файл

удаляется. Если в файле не было данных, таблицы заполняются временем по умолчанию.

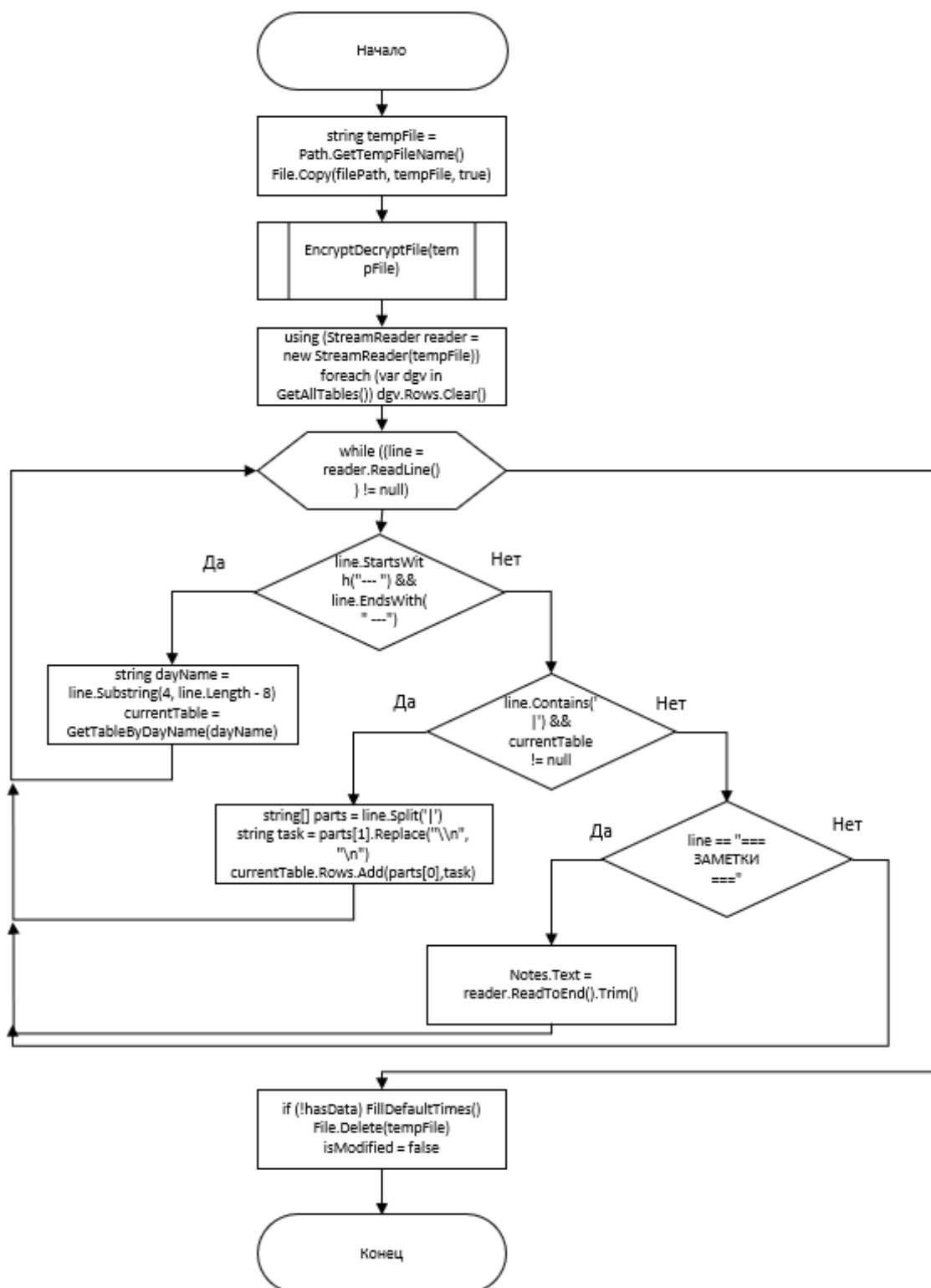


Рисунок 4 – Блок-схема загрузки и расшифровки файла

На рисунке 5 представлена блок-схема сохранения данных в файл и их шифрования (метод SaveToFile в PlanerForm).

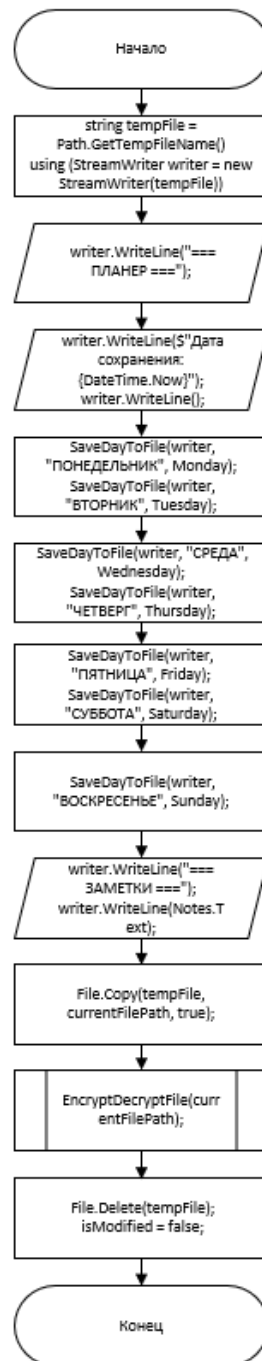


Рисунок 5 – Блок-схема сохранения и шифрования файла

Сначала создаётся временный файл. В него записываются заголовок, текущая дата и время. Затем для каждого из семи дней недели вызывается вспомогательная функция SaveDayToFile, которая записывает маркер дня, а затем все строки таблицы, кроме пустой строки для добавления новых

записей. При записи переносы строк в тексте задачи экранируются, чтобы не нарушать структуру файла. После записи всех дней в файл добавляется маркер «=== ЗАМЕТКИ ===» и текст из поля заметок. Временный файл копируется на место целевого файла с перезаписью, затем вызывается внешняя программа XORCrypter.exe для шифрования. Временный файл удаляется, флаг несохранённых изменений сбрасывается, и выводится сообщение об успешном сохранении.

На рисунке 6 изображена блок-схема вызова внешнего шифровальщика (функция main на C++).

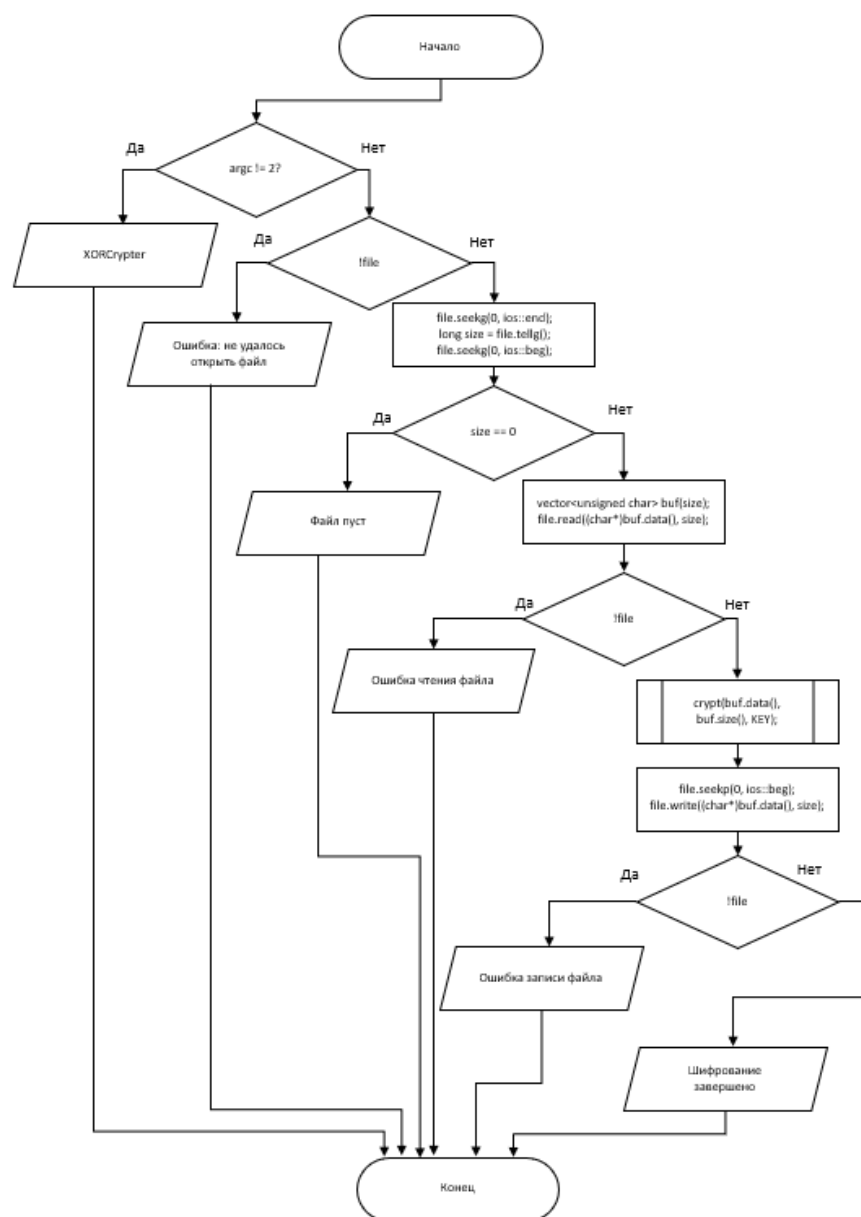


Рисунок 6 – Блок-схема шифрования на СИ++

Сначала формируется путь к исполняемому файлу XORCrypter.exe. Проверяется его наличие. Если файл не найден, выводится сообщение об ошибке, и выполнение прекращается. Если файл существует, создается объект ProcessStartInfo, в котором задаются имя файла шифровальщика, аргумент командной строки (путь к целевому файлу), отключается использование оболочки и окно консоли делается невидимым. Затем процесс запускается, и программа ожидает его завершения. После завершения работы шифровальщика управление возвращается в вызывающий метод.

На рисунке 7 представлена блок-схема ассемблерной функции xor_crypt, реализующей побайтовое XOR-шифрование [5].

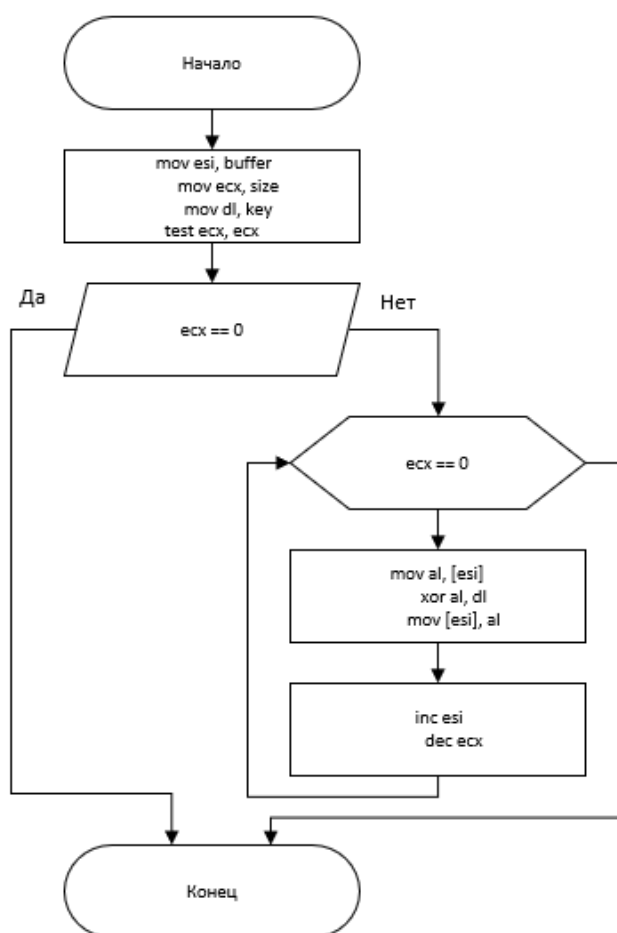


Рисунок 7 – Блок-схема ассемблерной функции xor_crypt

На вход функция получает указатель на буфер данных (buffer), размер буфера (size) и ключ шифрования (key = 0xAA). В начале работы указатель буфера загружается в регистр ESI, размер – в регистр ECX, ключ – в регистр DL. Выполняется проверка: если размер равен нулю, функция завершается без изменений. В цикле из памяти по адресу [ESI] считывается байт в регистр AL, выполняется операция XOR между AL и DL, результат записывается обратно в память по тому же адресу. Указатель ESI увеличивается на 1 для перехода к следующему байту, счётчик ECX уменьшается на 1. Цикл повторяется, пока ECX не станет равен нулю. Функция изменяет массив байтов «на месте» и используется как для шифрования, так и для расшифровки благодаря симметричности операции XOR.

Разработанные блок-схемы являются наглядным представлением алгоритмов, реализованных в программе «Планер», и подтверждают корректность логики работы всех модулей: главного меню, создания и открытия планов, загрузки и сохранения файлов с шифрованием, а также ассемблерной обработки данных.

Руководство пользователя

Программа «Планер» предназначена для ведения ежедневного расписания. Программа имеет интуитивно понятный графический интерфейс и поддерживает такие операции как создание нового плана, открытие существующего плана, редактирование расписания по дням недели, добавление и удаление строк в таблицах, ввод многострочных заметок, сохранение плана в зашифрованный файл, загрузка плана из файла, переключение между светлой и тёмной цветовыми темами, а также удаление файла плана.

Для запуска программы необходимо открыть исполняемый файл Planer.exe. После короткой заставки на экране появляется главное меню, содержащее четыре кнопки: «Создать план», «Открыть план» и «Сменить

тему». Управление в главном меню осуществляется с помощью мыши — нажатием левой кнопки на нужном пункте.

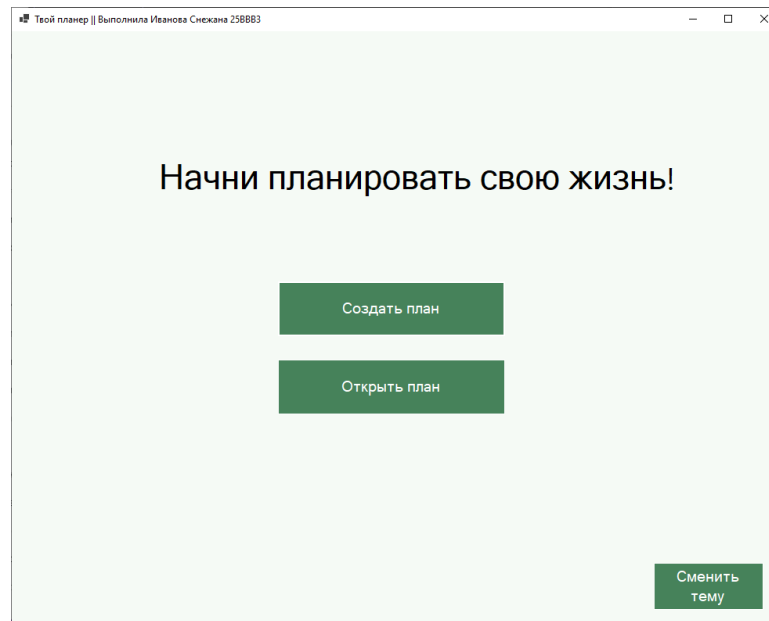


Рисунок 8 – Главное меню

Для создания нового плана нужно нажать кнопку «Создать план», ввести имя и нажать «Далее». Если файл с таким именем отсутствует, создаётся новый план и открывается редактор. Если файл уже существует, программа предложит открыть его. Кнопка «Назад» возвращает в главное меню.

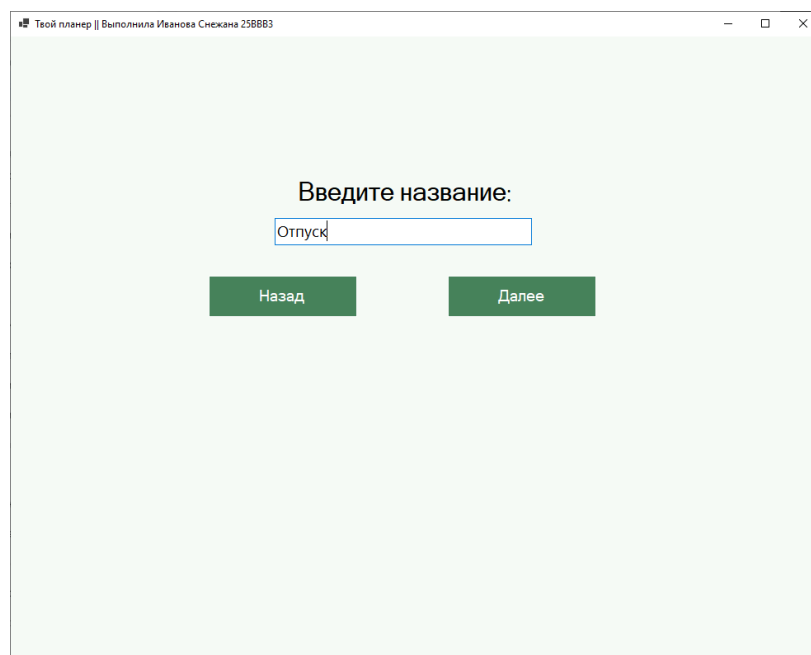


Рисунок 9 – Окно ввода названия плана

Для открытия плана нажмите «Открыть план», выберите файл двойным щелчком – откроется редактор с загруженными данными. Кнопка «Назад» возвращает в главное меню.

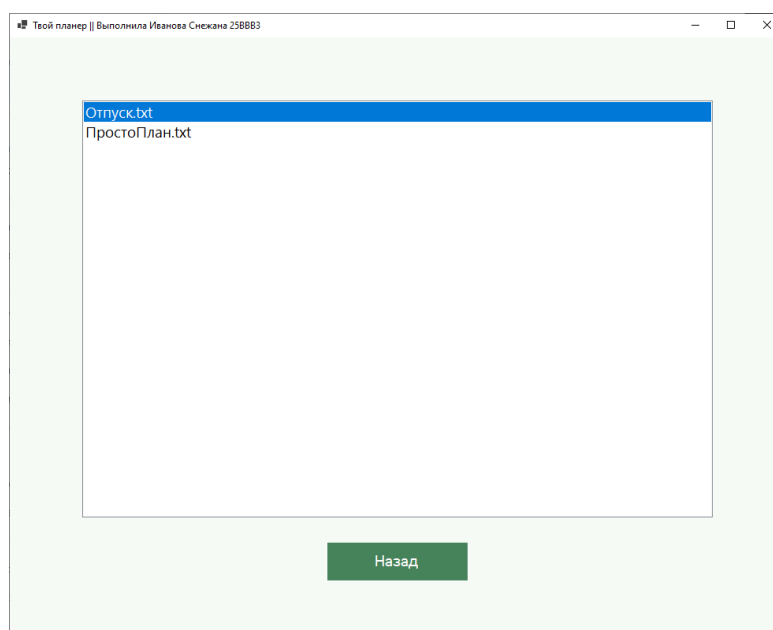


Рисунок 10 – Список сохранённых планов

Редактор разделён на восемь ячеек (сетка 4×2). В первых семи – таблицы для дней недели, в восьмой – поле для заметок. Это позволяет видеть расписание на всю неделю сразу.

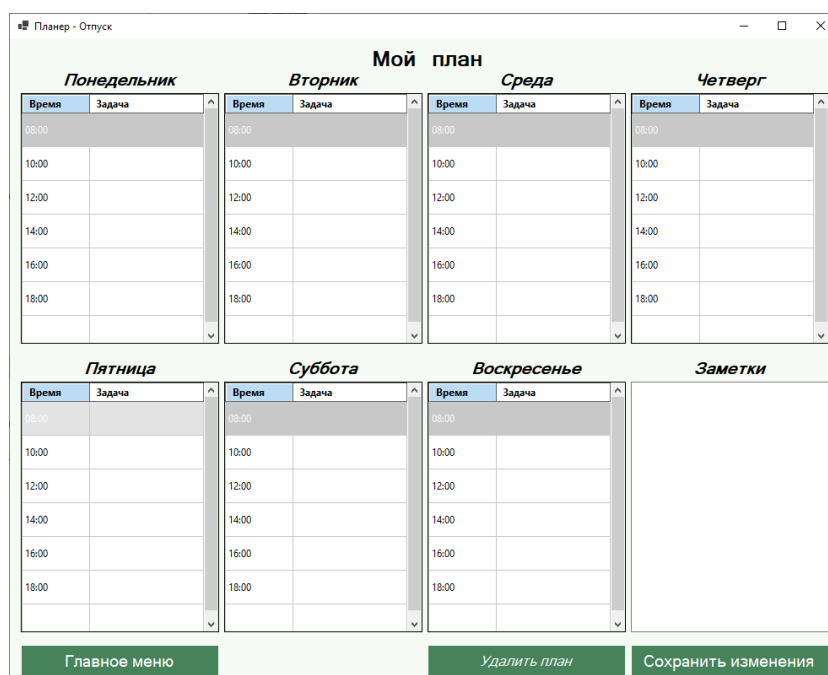


Рисунок 11 – Редактор плана

Каждая таблица имеет две колонки: «Время» и «Задача». По умолчанию в колонке «Время» указаны временные интервалы с 8:00 до 18:00 с шагом 2 часа. Пользователь может изменить любое значение времени по своему усмотрению. Для ввода задачи необходимо щёлкнуть левой кнопкой мыши в нужной ячейке колонки «Задача» и начать печатать текст с клавиатуры. При вводе длинного текста слова автоматически переносятся на следующую строку внутри ячейки, а высота строки таблицы увеличивается, чтобы весь текст был виден. Это позволяет записывать подробные описания задач.

Мой план

Понедельник

Время	Задача
08:00	Сбор чемоданов, проверить документы
10:00	Выезд в аэропорт
12:00	Регистрация на рейс, сдача багажа
14:00	Вылет
18:00	Прилёт, трансфер до отеля
20:00	Заселение, ужин

Вторник

Время	Задача
-	Пляж, купание, шезлонги

Среда

Время	Задача
09:00	Автобусная экскурсия по городу
12:00	Посещение исторического музея

Четверг

Время	Задача
15:00	Дайвинг

Пятница

Время	Задача
-	Поездка на местный рынок
-	Покупка сувениров, подарков

Суббота

Время	Задача
-	Пляж, купание, шезлонги

Воскресенье

Время	Задача
08:00	Прилёт, получение багажа
10:00	Такси домой
12:00	Разбор чемоданов
14:00	Отдых, стирка
16:00	Обед
20:00	Ужин, лёгкий сон

Заметки

Список вещей: паспорт, билеты, страховка, наличные, карта, телефон, зарядка, пауэрбанк, наушники, купальник, крем от загара, зубная щётка, ветровка.

Главное меню Удалить план Сохранить изменения

Рисунок 12 – Заполненный план

Для добавления новой строки в таблицу необходимо начать ввод в последней пустой строке, которая отмечена звёздочкой. Программа автоматически создаст новую пустую строку ниже. Для удаления строки

необходимо выделить её, щёлкнув по левому серому заголовку, и нажать клавишу Delete на клавиатуре.

В поле «Заметки» пользователь может записывать любые дополнительные сведения, не привязанные к конкретному дню или времени – например, список важных дел на неделю, напоминания или контактную информацию. Поле поддерживает ввод большого объёма текста и оснащено вертикальной полосой прокрутки.

Для редактирования ранее введённой задачи пользователь находит нужный день в соответствующей таблице, щёлкает в ячейке колонки «Задача» и вносит изменения. Для изменения времени аналогично щёлкается ячейка колонки «Время» и вводится новое значение. Все изменения автоматически отслеживаются программой.

В нижней части редактора расположены три кнопки: «Сохранить изменения», «Удалить план» и «Гляньте меню».

Для сохранения плана необходимо нажать кнопку «Сохранить изменения». Программа записывает содержимое всех таблиц и поле заметок в файл, предварительно зашифровав его с помощью ассемблерного модуля. После успешного сохранения появляется сообщение «План сохранён!».

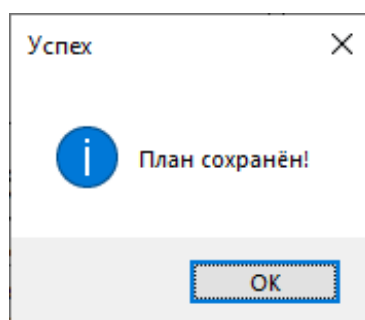


Рисунок 13 – Сообщение об успешном сохранении

Для удаления текущего плана необходимо нажать кнопку «Удалить план». Программа запрашивает подтверждение, и при согласии пользователя файл плана удаляется с диска, а редактор закрывается.

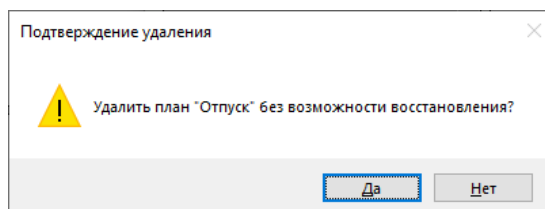


Рисунок 14 – Диалог подтверждения удаления плана

Для выхода в главное меню необходимо нажать кнопку «Гляньте меню». Если с момента последнего сохранения были внесены изменения, программа предупредит об этом и запросит подтверждение выхода без сохранения.

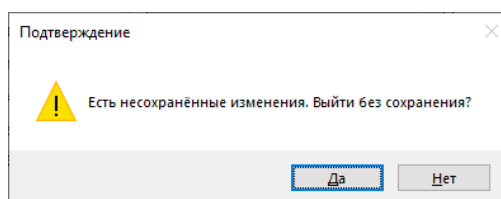


Рисунок 15 – Предупреждение о несохранённых изменениях

Для переключения цветовой темы необходимо в главном меню нажать кнопку «Сменить тему». Интерфейс полностью перекрашивается: светлая тема (светлый фон, тёмный текст, зелёные кнопки) меняется на тёмную (тёмный фон, белый текст, серые кнопки) и наоборот. Это позволяет адаптировать внешний вид под условия работы.

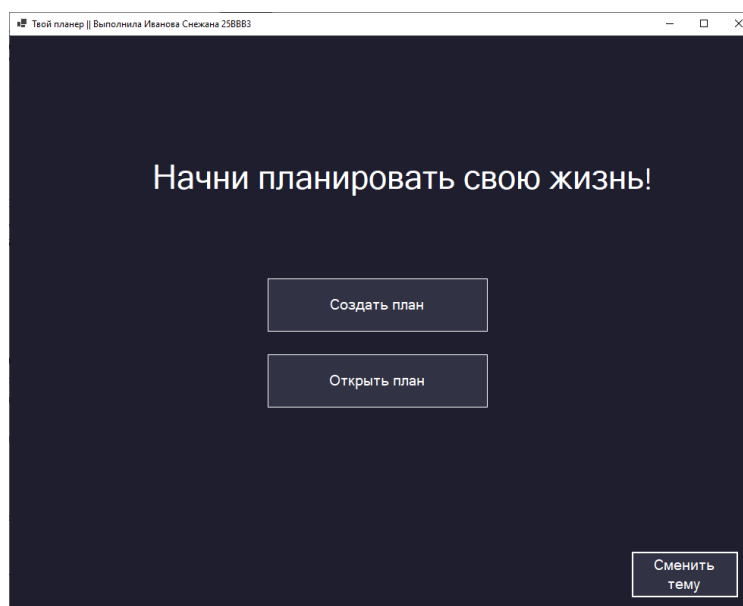


Рисунок 19 – Тёмная тема интерфейса

Заключение

При выполнении данной курсовой работы были получены навыки разработки многомодульных программ на языке C# с использованием Windows Forms. Была освоена работа с графическим интерфейсом, таблицами DataGridView, контейнерами TableLayoutPanel, файловой системой и шифрованием данных. Также были получены навыки программирования на языке C++ и Assembler при разработке внешнего модуля XORCrypter.exe, включая работу с аргументами командной строки, бинарными файлами и низкоуровневую обработку массивов байтов.

В рамках курсовой работы была написана программа «Планер» с графическим интерфейсом. Она имеет все необходимые функции для ведения ежедневного расписания: создание и открытие планов, редактирование таблиц с возможностью добавления и удаления строк, ввод многострочных заметок, сохранение и загрузку файлов, а также переключение между светлой и тёмной темами. В программе использована вставка на языке Assembler (ключевое слово `__asm` в C++) для побайтового XOR-шифрования файлов, что демонстрирует взаимодействие высокоуровневого и низкоуровневого кода.

В дальнейшем программу можно улучшить путём добавления автоматического сохранения, экспорта в форматы CSV или PDF, напоминаний о задачах и облачной синхронизации.

Список используемых источников

1. Документация по Windows Forms. – URL: <https://learn.microsoft.com/en-us/dotnet/desktop/winforms/>
2. Мурадханов С. Э. Разработка на языке C# приложений с графическим интерфейсом. 2019 г. – 396 с.
3. Керниган Б.У., Ритчи Д.М. Язык программирования C. 2020 г. – 288 с.
4. Microsoft Learn. Настройка целевых платформ в проектах - Visual Studio. – URL: <https://learn.microsoft.com/ru-ru/visualstudio/ide/how-to-configure-projects-to-target-platforms?view=visualstudio>
5. Фомичева С. Г., Варига О. С., Попкова А. А. Основы программирования на языке низкого уровня Assembler: учебное пособие. 2022 г. – 198 с.

Приложение А

Листинги программы

Файл Program.cs

```
namespace Planer
{
    internal static class Program
    {
        [STAThread]
        static void Main()
        {
            ApplicationConfiguration.Initialize();
            Application.Run(new Form1());
        }
    }
}
```

Файл Form1.cs

```
using System;
using System.IO;
using System.Windows.Forms;
using Planer.Themes;

namespace Planer
{
    public partial class Form1 : Form
    {
        private bool isDarkTheme = false;

        public Form1()
        {
            InitializeComponent();
            LightTheme.ApplyToAll(this);
        }

        private void Theme_Click(object sender, EventArgs e)
        {
        }
    }
}
```

```

{
    if (isDarkTheme)
    {
        LightTheme.ApplyToAll(this);
        isDarkTheme = false;
    }
    else
    {
        DarkTheme.ApplyToAll(this);
        isDarkTheme = true;
    }
}

private void NewPlan_Click(object sender, EventArgs e)
{
    NewPlan.Visible = false;
    OpenPlan.Visible = false;
    label1.Visible = false;
    Theme.Visible = false;

    lName.Visible = true;
    txtPlanName.Visible = true;
    button2.Visible = true;
    button1.Visible = true;
}

private void OpenPlan_Click(object sender, EventArgs e)
{
    LoadFiles();
    ShowPanel(true);
}

private void LoadFiles()
{

```

```

        string folder = Path.Combine(Application.StartupPath,
"Plans");

        if (!Directory.Exists(folder))
            Directory.CreateDirectory(folder);

        string[] files = Directory.GetFiles(folder, "*.txt");
        listBox1.Items.Clear();
        foreach (string file in files)
            listBox1.Items.Add(Path.GetFileName(file));

        if (listBox1.Items.Count == 0)
            listBox1.Items.Add("(Нет планов)");
    }

    private void ShowPanel(bool show)
    {
        NewPlan.Visible = !show;
        OpenPlan.Visible = !show;
        label1.Visible = !show;
        Theme.Visible = !show;
        listBox1.Visible = show;
        label2.Visible = show;
        Back.Visible = show;
    }

    private void listBox1_MouseDoubleClick(object sender,
MouseEventArgs e)
    {
        if (listBox1.SelectedItem != null &&
listBox1.SelectedItem.ToString() != "(Нет планов)")
        {
            string name = listBox1.SelectedItem.ToString();
            string path =
Path.Combine(Application.StartupPath, "Plans", name);

```

```

        this.Hide();
        PlanerForm planerForm = new PlanerForm(path,
isDarkTheme);

        planerForm.ShowDialog();
        this.Show();

        ShowPanel(false);
    }
}

private void Back_Click(object sender, EventArgs e)
{
    ShowPanel(false);
    ShowMainMenu();
}

private void button2_Click(object sender, EventArgs e)
{
    string planName = txtPlanName.Text.Trim();
    if (string.IsNullOrEmpty(planName))
    {
        MessageBox.Show("Введите название плана!",
"Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Warning);
        return;
    }

    string plansFolder =
Path.Combine(Application.StartupPath, "Plans");
    if (!Directory.Exists(plansFolder))
        Directory.CreateDirectory(plansFolder);

    string filePath = Path.Combine(plansFolder, planName
+ ".txt");

    if (File.Exists(filePath))

```

```

        {
            DialogResult result = MessageBox.Show(
                $"План с именем \"{planName}\" уже существует.
Открыть его?",
                "Файл существует",
                MessageBoxButtons.YesNo,
                MessageBoxIcon.Question);

            if (result == DialogResult.Yes)
            {
                this.Hide();
                PlanerForm planerForm = new
PlanerForm(filePath, isDarkTheme);
                planerForm.ShowDialog();
                this.Show();
            }

            HideCreationControls();
            ShowMainMenu();
            return;
        }

        CreateEmptyPlanFile(filePath);
        HideCreationControls();

        this.Hide();
        PlanerForm planerForm2 = new PlanerForm(filePath,
isDarkTheme);
        planerForm2.ShowDialog();
        this.Show();

        ShowMainMenu();
    }

    private void button1_Click(object sender, EventArgs e)

```

```

        {
            HideCreationControls();
            ShowMainMenu();
        }

        private void CreateEmptyPlanFile(string filePath)
        {
            using (StreamWriter writer = new
StreamWriter(filePath))
            {
                writer.WriteLine("=== ПЛАНЕР ===");
                writer.WriteLine($"Дата создания:
{DateTime.Now}");
                writer.WriteLine();

                string[] days = { "ПОНЕДЕЛЬНИК", "ВТОРНИК",
"СРЕДА", "ЧЕТВЕРГ",
                                "ПЯТНИЦА", "СУББОТА",
"ВОСКРЕСЕНЬЕ" };

                string[] times = { "08:00", "10:00", "12:00",
"14:00", "16:00", "18:00" };

                foreach (string day in days)
                {
                    writer.WriteLine($"--- {day} ---");
                    foreach (string time in times)
                    {
                        writer.WriteLine($"{time}|");
                        writer.WriteLine();
                    }

                    writer.WriteLine("=== ЗАМЕТКИ ===");
                    writer.WriteLine();
                }
            }
        }
    }

```

```

private void HideCreationControls()
{
    lName.Visible = false;
    txtPlanName.Visible = false;
    button2.Visible = false;
    button1.Visible = false;
}

public void ShowMainMenu()
{
    HideCreationControls();

    listBox1.Visible = false;
    label2.Visible = false;
    Back.Visible = false;

    NewPlan.Visible = true;
    OpenPlan.Visible = true;
    label1.Visible = true;
    Theme.Visible = true;
}

private void listBox1_SelectedIndexChanged(object sender,
EventArgs e) { }
}

```

Файл PlanerForm.cs

```

using System;
using System.IO;
using System.Windows.Forms;
using System.Diagnostics;
using Planer.Themes;

namespace Planer

```

```

{
    public partial class PlanerForm : Form
    {
        private readonly string currentFilePath;
        private bool isModified;

        public PlanerForm(string filePath, bool isDarkTheme)
        {
            InitializeComponent();
            currentFilePath = filePath;
            SetupDataGridViews();
            LoadFromFile(filePath);
            Text = $"Планер - {Path.GetFileNameWithoutExtension(filePath)}";
            Resize += PlanerForm_Resize;
            UpdateFonts();
            ApplyTheme(isDarkTheme);
        }

        private void ApplyTheme(bool isDark)
        {
            if (isDark)
                DarkTheme.ApplyToAll(this);
            else
                LightTheme.ApplyToAll(this);
        }

        private void SetupDataGridViews()
        {
            DataGridView[] tables = GetAllTables();

            foreach (DataGridView dgv in tables)
            {
                dgv.RowHeadersVisible = false;
                dgv.ColumnCount = 2;
            }
        }
    }
}

```

```

        dgv.Columns[0].Name = "Time";
        dgv.Columns[0].HeaderText = "Время";
        dgv.Columns[0].Width = 80;
        dgv.Columns[1].Name = "Task";
        dgv.Columns[1].HeaderText = "Задача";
        dgv.Columns[1].AutoSizeMode =
DataGridViewAutoSizeColumnMode.Fill;
        dgv.AllowUserToAddRows = true;
        dgv.RowTemplate.MinimumHeight = 40;
        dgv.ScrollBars = ScrollBars.Vertical;
        dgv.Columns[1].DefaultCellStyle.WrapMode =
DataGridViewTriState.True;
        dgv.AutoSizeRowsMode =
DataGridViewAutoSizeRowsMode.AllCells;

        dgv.EditingControlShowing +=
Dgv_EditingControlShowing;
        dgv.CellValueChanged += Dgv_CellValueChanged;
        dgv.RowsAdded += Dgv_RowsChanged;
        dgv.RowsRemoved += Dgv_RowsChanged;
    }

    Notes.TextChanged += Notes_TextChanged;
}

private void Dgv_CellValueChanged(object sender,
DataGridViewCellEventArgs e)
{
    isModified = true;
}

private void Dgv_RowsChanged(object sender, EventArgs e)
{
    isModified = true;
}

```

```

private void Notes_TextChanged(object sender, EventArgs e)
{
    isModified = true;
}

private void Dgv_EditingControlShowing(object sender,
DataGridViewEditingControlShowingEventArgs e)
{
    if (e.Control is TextBox textBox)
    {
        textBox.WordWrap = true;
    }
}

private void LoadFromFile(string filePath)
{
    if (!File.Exists(filePath)) return;

    string tempFile = Path.GetTempFileName();
    File.Copy(filePath, tempFile, true);
    EncryptDecryptFile(tempFile);

    bool hasData = false;

    using (StreamReader reader = new
StreamReader(tempFile))
    {
        string line;
        DataGridView currentTable = null;

        foreach (var dgv in GetAllTables())
            dgv.Rows.Clear();

        while ((line = reader.ReadLine()) != null)

```

```

        {
            if (line.StartsWith("---") &&
line.EndsWith(" ---"))
            {
                string dayName = line.Substring(4,
line.Length - 8);

                currentTable =
GetTableByDayName(dayName);
            }
            else if (line.Contains('|') && currentTable !=
null)
            {
                hasData = true;
                string[] parts = line.Split('|');
                string time = parts.Length > 0 ? parts[0]
: "";

                string task = parts.Length > 1 ? parts[1]
: "";

                task = task.Replace("\n", "\n");
                currentTable.Rows.Add(time, task);
            }
            else if (line == "=== ЗАМЕТКИ ===")
            {
                Notes.Text = reader.ReadToEnd().Trim();
                break;
            }
        }
    }
    if (!hasData)
    {
        FillDefaultTimes();
    }

    File.Delete(tempFile);
    isModified = false;

```

```

    }
    private void FillDefaultTimes()
    {
        string[] times = { "08:00", "10:00", "12:00", "14:00",
"16:00", "18:00" };

        foreach (var dgv in GetAllTables())
        {
            dgv.Rows.Clear();
            foreach (string time in times)
            {
                dgv.Rows.Add(time, "");
            }
        }
    }
    private void SaveToFile()
    {
        string tempFile = Path.GetTempFileName();
        using (StreamWriter writer = new
StreamWriter(tempFile))
        {
            writer.WriteLine("=== ПЛАНЕР ===");
            writer.WriteLine($"Дата                сохранения:
{DateTime.Now}");
            writer.WriteLine();

            SaveDayToFile(writer, "ПОНЕДЕЛЬНИК", Monday);
            SaveDayToFile(writer, "ВТОРНИК", Tuesday);
            SaveDayToFile(writer, "СРЕДА", Wednesday);
            SaveDayToFile(writer, "ЧЕТВЕРГ", Thursday);
            SaveDayToFile(writer, "ПЯТНИЦА", Friday);
            SaveDayToFile(writer, "СУББОТА", Saturday);
            SaveDayToFile(writer, "ВОСКРЕСЕНЬЕ", Sunday);

            writer.WriteLine("=== ЗАМЕТКИ ===");

```

```

        writer.WriteLine(Notes.Text);
    }

    File.Copy(tempFile, currentFilePath, true);
    EncryptDecryptFile(currentFilePath);

    File.Delete(tempFile);
    isModified = false;
}

private void SaveDayToFile(StreamWriter writer, string
dayName, DataGridView dgv)
{
    writer.WriteLine($"--- {dayName} ---");
    foreach (DataGridViewRow row in dgv.Rows)
    {
        if (!row.IsNewRow)
        {
            string time = row.Cells[0].Value?.ToString()
?? "";

            string task = row.Cells[1].Value?.ToString()
?? "";

            task = task.Replace("\n",
"\n\n").Replace("\r\n", "\n").Replace("\n", "\n");
            writer.WriteLine($"{time}|{task}");
        }
    }
    writer.WriteLine();
}

private DataGridView GetTableByDayName(string dayName)
{
    switch (dayName)
    {
        case "ПОНЕДЕЛЬНИК": return Monday;
    }
}

```

```

        case "ВТОРНИК": return Tuesday;
        case "СРЕДА": return Wednesday;
        case "ЧЕТВЕРГ": return Thursday;
        case "ПЯТНИЦА": return Friday;
        case "СУББОТА": return Saturday;
        case "ВОСКРЕСЕНЬЕ": return Sunday;
        default: return null;
    }
}

private DataGridView[] GetAllTables()
{
    return new DataGridView[]
    {
        Monday, Tuesday, Wednesday, Thursday,
        Friday, Saturday, Sunday
    };
}

private void Save_Click(object sender, EventArgs e)
{
    SaveToFile();
    MessageBox.Show("План сохранён!", "Успех",
    MessageBoxButtons.OK, MessageBoxIcon.Information);
}

private void Exit_Click(object sender, EventArgs e)
{
    if (isModified)
    {
        DialogResult result = MessageBox.Show(
            "Есть несохранённые изменения. Выйти без
сохранения?",
            "Подтверждение",
            MessageBoxButtons.YesNo,

```

```

        MessageBoxIcon.Warning);

        if (result != DialogResult.Yes)
            return;
    }

    Close();
}

private void PlanerForm_Resize(object sender, EventArgs e)
{
    UpdateFonts();
}

private void UpdateFonts()
{
    bool      isMaximized      =      (WindowState      ==
FormWindowState.Maximized);
    int fontSize = isMaximized ? 14 : 9;
    int rowHeight = isMaximized ? 40 : 30;
    int timeWidth = isMaximized ? 100 : 80;

    DataGridView[] tables = GetAllTables();
    foreach (DataGridView dgv in tables)
    {
        dgv.DefaultCellStyle.Font = new Font("Segoe UI",
fontSize);
        dgv.ColumnHeadersDefaultCellStyle.Font      =      new
Font("Segoe UI", fontSize, FontStyle.Bold);
        dgv.RowTemplate.Height = rowHeight;
        dgv.RowTemplate.MinimumHeight = rowHeight;
        dgv.Columns[0].Width = timeWidth;

        dgv.EditingControlShowing      ==
Dgv_EditingControlShowing;
    }
}

```

```

        dgv.EditingControlShowing                                     +=
Dgv_EditingControlShowing;
    }

    Notes.Font = new Font("Segoe UI", fontSize);
}

private void btnDeletePlan_Click(object sender, EventArgs
e)
{
    string planName =
Path.GetFileNameWithoutExtension(currentFilePath);
    DialogResult result = MessageBox.Show(
        $"Удалить план \"{planName}\" без возможности
восстановления?",
        "Подтверждение удаления",
        MessageBoxButtons.YesNo,
        MessageBoxIcon.Warning);

    if (result == DialogResult.Yes)
    {
        try
        {
            File.Delete(currentFilePath);
            MessageBox.Show("План удалён.", "Удаление",
MessageBoxButtons.OK, MessageBoxIcon.Information);
            this.Close();
        }
        catch (Exception ex)
        {
            MessageBox.Show($"Ошибка при удалении:
{ex.Message}", "Ошибка", MessageBoxButtons.OK,
MessageBoxIcon.Error);
        }
    }
}

```

```

    }
    private void EncryptDecryptFile(string filePath)
    {
        string crypterPath =
Path.Combine(Application.StartupPath, "XORCrypter.exe");

        if (!File.Exists(crypterPath))
        {
            MessageBox.Show("Не найден XORCrypter.exe",
"Ошибка",
            MessageBoxButtons.OK, MessageBoxIcon.Error);
            return;
        }

        Process process = new Process();
        process.StartInfo.FileName = crypterPath;
        process.StartInfo.Arguments = "\"" + filePath + "\"";
        process.StartInfo.UseShellExecute = false;
        process.StartInfo.CreateNoWindow = true;
        process.Start();
        process.WaitForExit();
    }
}

```

Файл LightTheme.cs

```

using System.Drawing;
using System.Windows.Forms;

namespace Planer.Themes
{
    public static class LightTheme
    {
        public static void Apply(Form form, params Button[]
buttons)

```

```

{
    form.BackColor = Color.FromArgb(245, 250, 245);

    foreach (Button btn in buttons)
    {
        btn.BackColor = Color.FromArgb(70, 130, 90);
        btn.ForeColor = Color.White;
        btn.FlatStyle = FlatStyle.Flat;
    }
}

public static void ApplyToAll(Form form)
{

    form.BackColor = Color.FromArgb(245, 250, 245);

    ApplyToControl(form);
}

private static void ApplyToControl(Control parent)
{
    foreach (Control ctrl in parent.Controls)
    {

        if (ctrl is Button btn)
        {
            btn.BackColor = Color.FromArgb(70, 130, 90);
            btn.ForeColor = Color.White;
            btn.FlatStyle = FlatStyle.Flat;
        }
        else if (ctrl is Label lbl)
        {
            lbl.ForeColor = Color.Black;
        }
    }
}

```

```

else if (ctrl is DataGridView dgv)
{
    dgv.BackgroundColor = Color.White;
    dgv.GridColor = Color.FromArgb(200, 200,
200);

    dgv.DefaultCellStyle.BackColor = Color.White;
    dgv.DefaultCellStyle.ForeColor = Color.Black;
    dgv.ColumnHeadersDefaultCellStyle.BackColor =
Color.FromArgb(70, 130, 90);
    dgv.ColumnHeadersDefaultCellStyle.ForeColor =
Color.White;
}

else if (ctrl is TextBox txt)
{
    txt.BackColor = Color.White;
    txt.ForeColor = Color.Black;
}

else if (ctrl is TabControl tabs)
{
    tabs.BackColor = Color.FromArgb(245, 250,
245);

    foreach (TabPage page in tabs.TabPages)
    {
        page.BackColor = Color.FromArgb(245, 250,
245);
        page.ForeColor = Color.Black;
    }
}

ApplyToControl(ctrl);
}
}
}

```

```
}
```

Файл DarkTheme.cs

```
using System.Drawing;
```

```
using System.Windows.Forms;
```

```
namespace Planer.Themes
```

```
{
```

```
    public static class DarkTheme
```

```
    {
```

```
        public static void Apply(Form form, params Button[]  
buttons)
```

```
        {
```

```
            form.BackColor = Color.FromArgb(30, 30, 47);
```

```
            foreach (Button btn in buttons)
```

```
            {
```

```
                btn.BackColor = Color.FromArgb(49, 50, 68);
```

```
                btn.ForeColor = Color.White;
```

```
                btn.FlatStyle = FlatStyle.Flat;
```

```
            }
```

```
        }
```

```
        public static void ApplyToAll(Form form)
```

```
        {
```

```
            form.BackColor = Color.FromArgb(30, 30, 47);
```

```
            ApplyToControl(form);
```

```
        }
```

```
        private static void ApplyToControl(Control parent)
```

```
        {
```

```

foreach (Control ctrl in parent.Controls)
{

    if (ctrl is Button btn)
    {
        btn.BackColor = Color.FromArgb(49, 50, 68);
        btn.ForeColor = Color.White;
        btn.FlatStyle = FlatStyle.Flat;
    }
    else if (ctrl is Label lbl)
    {
        lbl.ForeColor = Color.White;

    }

    else if (ctrl is DataGridView dgv)
    {
        dgv.BackgroundColor = Color.FromArgb(24, 24,
37);

        dgv.GridColor = Color.FromArgb(49, 50, 68);
        dgv.DefaultCellStyle.BackColor =
Color.FromArgb(24, 24, 37);
        dgv.DefaultCellStyle.ForeColor = Color.White;
        dgv.ColumnHeadersDefaultCellStyle.BackColor =
Color.FromArgb(49, 50, 68);
        dgv.ColumnHeadersDefaultCellStyle.ForeColor =
Color.White;
    }

    else if (ctrl is TextBox txt)
    {
        txt.BackColor = Color.FromArgb(24, 24, 37);
        txt.ForeColor = Color.White;
    }
}

```

```

        else if (ctrl is TabControl tabs)
        {
            tabs.BackColor = Color.FromArgb(30, 30, 47);
            foreach (TabPage page in tabs.TabPages)
            {
                page.BackColor = Color.FromArgb(30, 30,
47);

                page.ForeColor = Color.White;
            }
        }

        ApplyToControl(ctrl);
    }
}
}
}
}

```

Файл XORCrypter.cpp

```

#include <iostream>
#include <fstream>
#include <vector>
using namespace std;

const unsigned char KEY = 0xAA;

void crypt(unsigned char* data, unsigned int len, unsigned char
key) {
    __asm {
        mov esi, data
        mov ecx, len
        mov dl, key
        test ecx, ecx
        jz done
        loop_start :
        mov al, [esi]
        xor al, dl
    }
}

```

```

        mov[esi], al
        inc esi
        dec ecx
        jnz loop_start
    done :
}
}

int main(int argc, char* argv[]) {
    system("chcp 1251 > nul");
    if (argc != 2) {
        cout << "XORCrypter" << endl;
        return 1;
    }

    fstream file(argv[1], ios::binary | ios::in | ios::out);
    if (!file) {
        cout << "Ошибка: не удалось открыть файл" << endl;
        return 1;
    }

    file.seekg(0, ios::end);
    long size = file.tellg();
    file.seekg(0, ios::beg);

    if (size == 0) {
        cout << "Файл пуст" << endl;
        return 0;
    }

    vector<unsigned char> buf(size);
    file.read((char*)buf.data(), size);
    if (!file) {
        cout << "Ошибка чтения файла" << endl;
        return 1;
    }
}

```

```

    }

    crypt(buf.data(), buf.size(), KEY);

    file.seekp(0, ios::beg);
    file.write((char*)buf.data(), size);
    if (!file) {
        cout << "Ошибка записи файла" << endl;
        return 1;
    }

    cout << "Шифрование завершено" << endl;
    return 0;
}

```

Приложение В

Основные интерфейсы программы

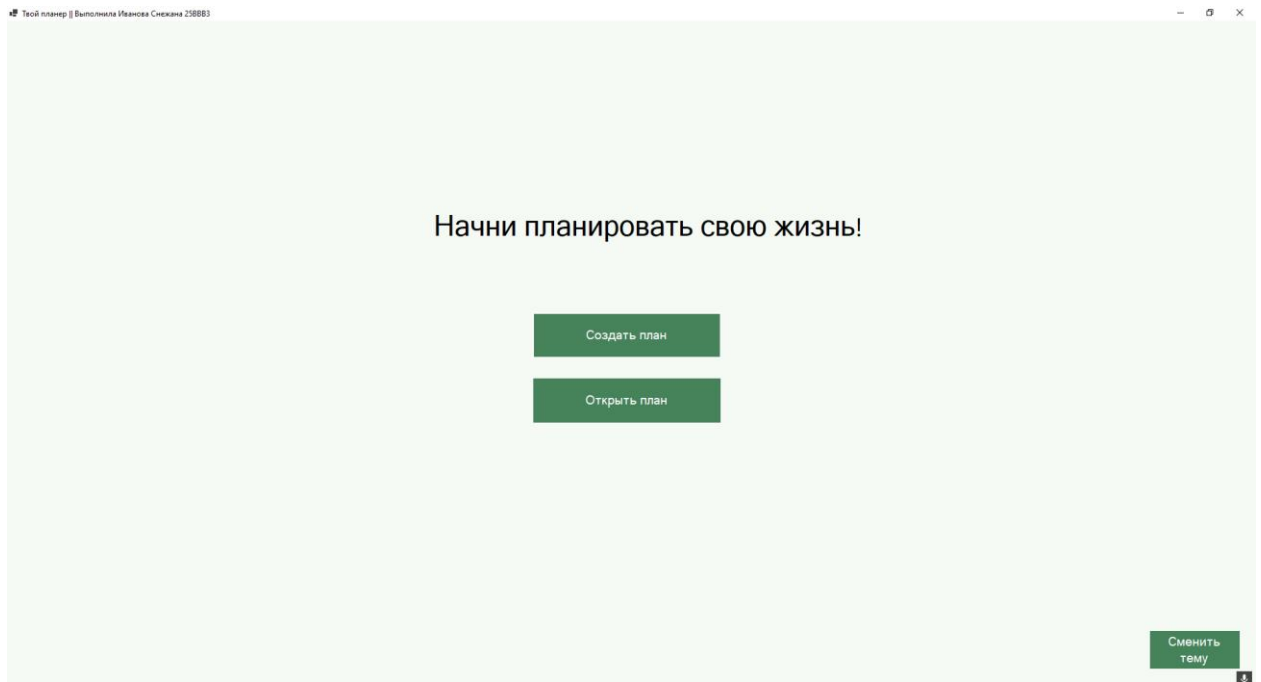


Рисунок В.1 – Главное меню

Главное меню содержит четыре кнопки: «Создать план», «Открыть план», «Сменить тему» и «Выход».

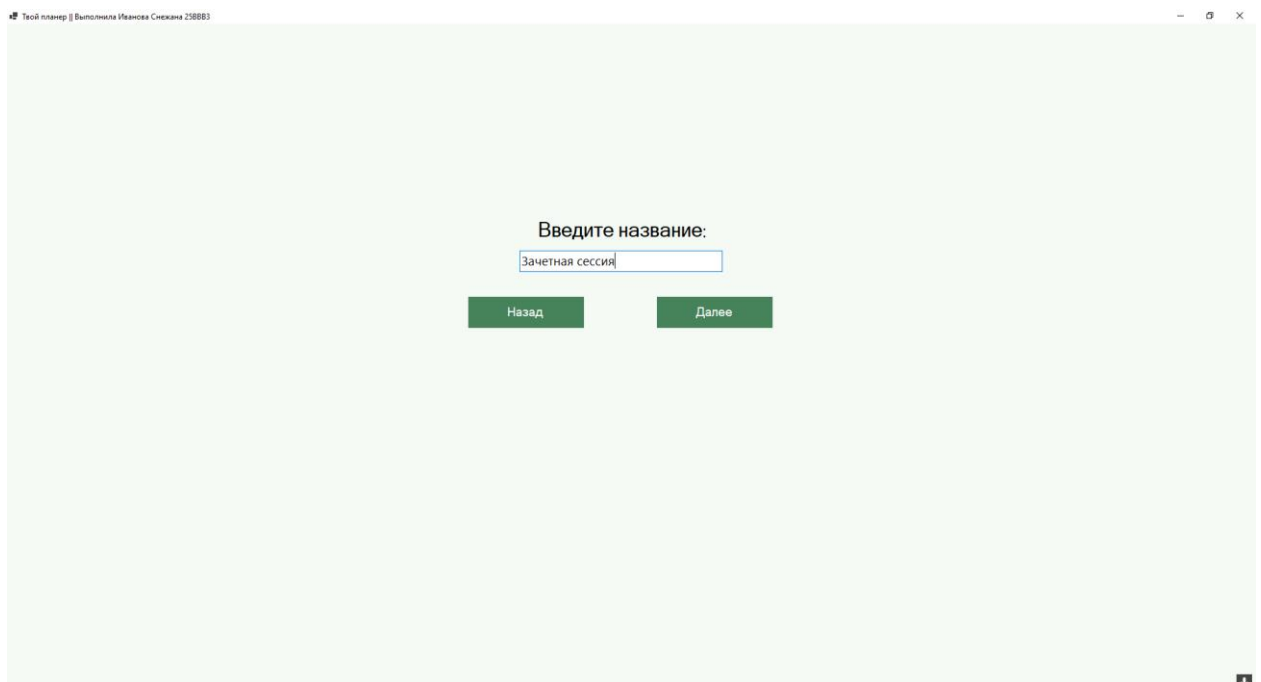


Рисунок В.2 – Создание нового плана

При нажатии на кнопку «Создать план» открывается панель с полем для ввода имени плана и кнопками «Далее» и «Назад».

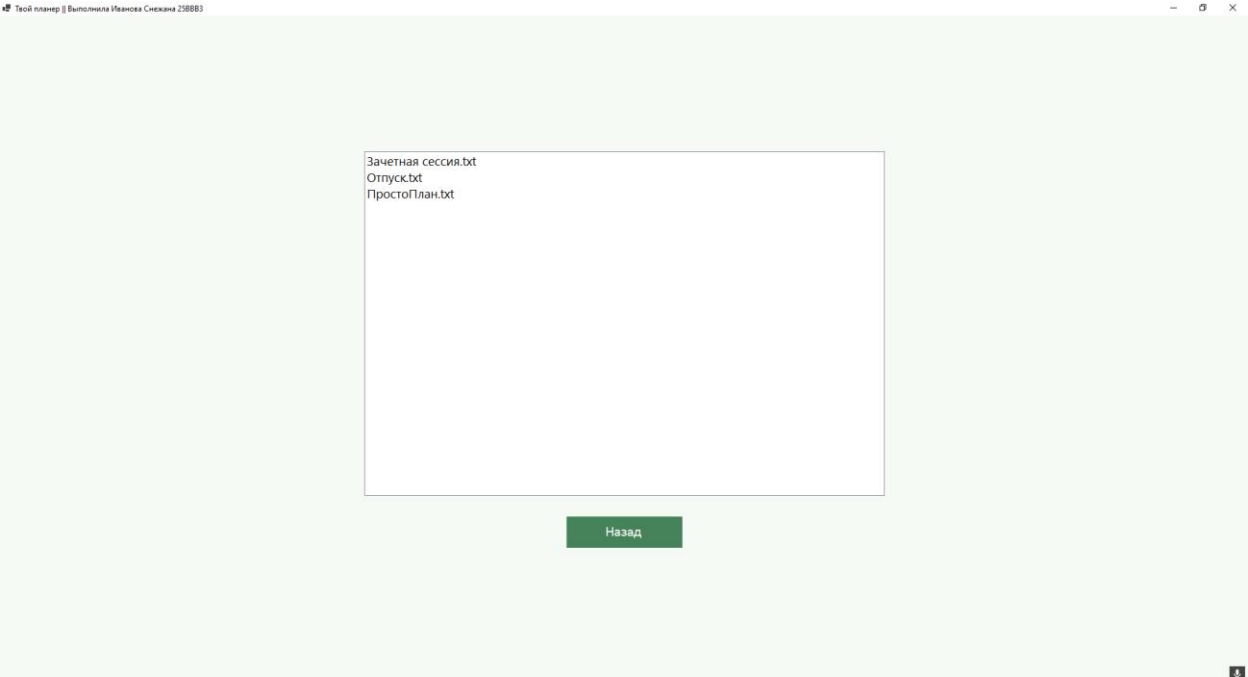


Рисунок В.3 – Открытие существующего плана

При нажатии на кнопку «Открыть план» открывается панель со списком всех сохранённых планов. Для открытия нужно дважды щёлкнуть по названию.

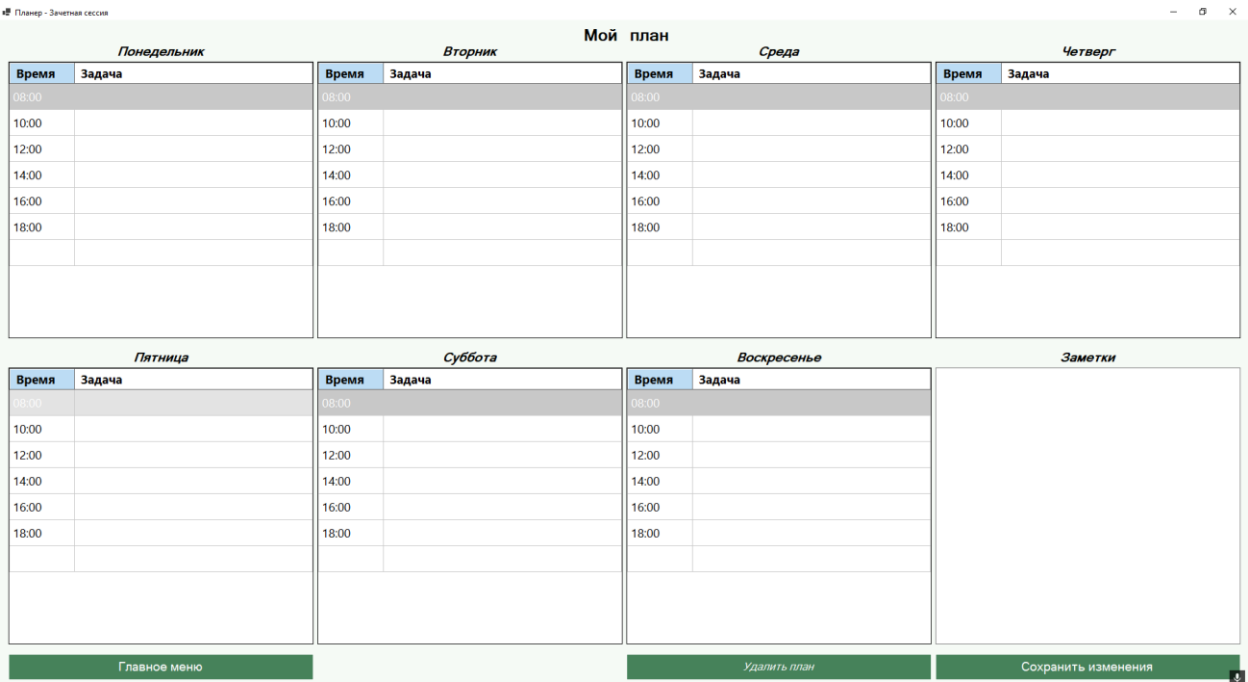


Рисунок В.4 – Редактор плана

Редактор разделён на восемь ячеек (сетка 4×2). В первых семи расположены таблицы для дней недели, в восьмой – поле для заметок. Внизу находятся кнопки «Сохранить изменения», «Удалить план» и «Гляньте меню».

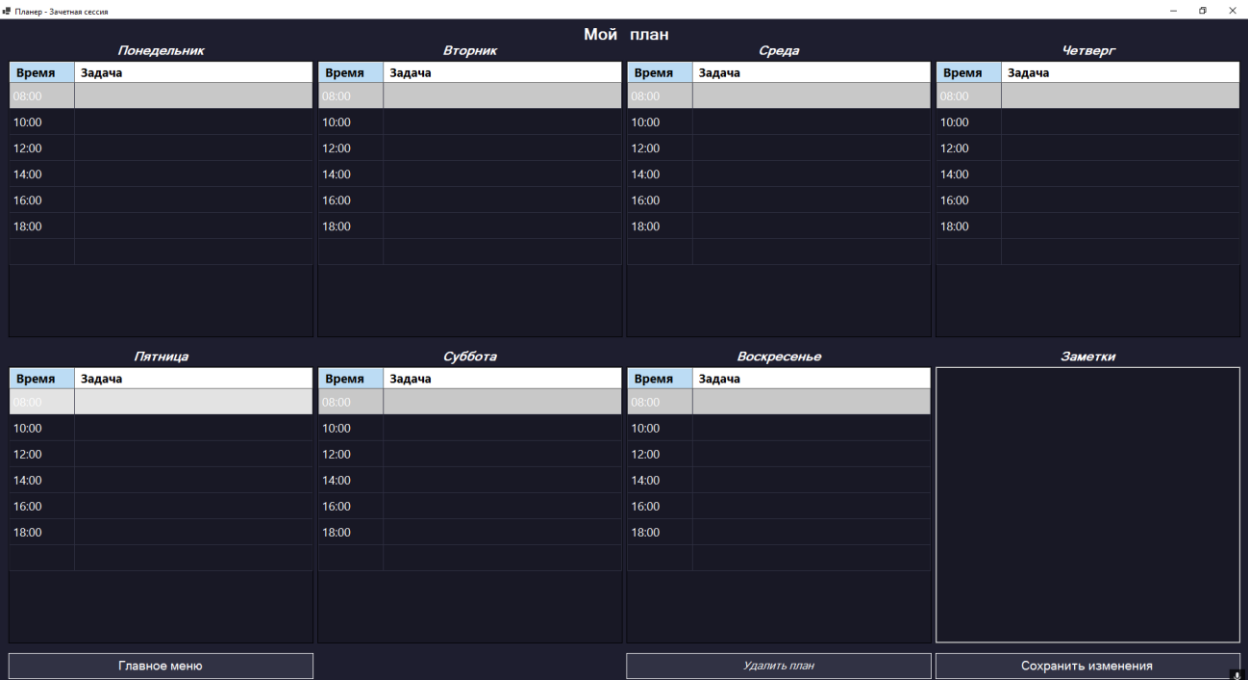


Рисунок В.5 – Тёмная тема интерфейса

Пример внешнего вида программы в тёмной цветовой теме.