КОМИТЕТ ПО ОБРАЗОВАНИЮ МИНГОРИСПОЛКОМА

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

«МИНСКИЙ ГОСУДАРСТВЕННЫЙ КОЛЛЕДЖ ЭЛЕКТРОНИКИ»

Специальность: 2-40 01 01

«Программное обеспечение информационных технологий»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

*«*Разработка приложения «INFINITY (Регистратура ­­- управление прайс-листом, занятость манипуляционных/операционных)»

НА КУРСОВОЕ ПРОЕКТИРОВАНИЕ

КП 53ТП.003102.081 ПЗ

Автор проекта (В. П. Богданович)

Руководитель проекта (Е.А. Артемьева )

2022

СОДЕРЖАНИЕ

[Введение 4](file:///G:\шарага\ОАиП\Пример%20КП%20(full%20ver)\DOC\Пояснительная%20записка%20Баразна%2035ТП.docx#_Toc12365726)

[1 Постановка задачи 4](file:///G:\шарага\ОАиП\Пример%20КП%20(full%20ver)\DOC\Пояснительная%20записка%20Баразна%2035ТП.docx#_Toc12365727)

[1.1 Организационная сущность задачи 5](file:///G:\шарага\ОАиП\Пример%20КП%20(full%20ver)\DOC\Пояснительная%20записка%20Баразна%2035ТП.docx#_Toc12365728)

[1.2 Функциональная модель 5](file:///G:\шарага\ОАиП\Пример%20КП%20(full%20ver)\DOC\Пояснительная%20записка%20Баразна%2035ТП.docx#_Toc12365729)

[1.3 Входные данные 6](file:///G:\шарага\ОАиП\Пример%20КП%20(full%20ver)\DOC\Пояснительная%20записка%20Баразна%2035ТП.docx#_Toc12365730)

[1.4 Выходные данные 7](file:///G:\шарага\ОАиП\Пример%20КП%20(full%20ver)\DOC\Пояснительная%20записка%20Баразна%2035ТП.docx#_Toc12365731)

[2 Вычислительная система 9](file:///G:\шарага\ОАиП\Пример%20КП%20(full%20ver)\DOC\Пояснительная%20записка%20Баразна%2035ТП.docx#_Toc12365732)

[2.1 Используемые технические средства 9](file:///G:\шарага\ОАиП\Пример%20КП%20(full%20ver)\DOC\Пояснительная%20записка%20Баразна%2035ТП.docx#_Toc12365733)

[2.2 Инструменты разработки 9](file:///G:\шарага\ОАиП\Пример%20КП%20(full%20ver)\DOC\Пояснительная%20записка%20Баразна%2035ТП.docx#_Toc12365734)

[3 Проектирование задачи 11](file:///G:\шарага\ОАиП\Пример%20КП%20(full%20ver)\DOC\Пояснительная%20записка%20Баразна%2035ТП.docx#_Toc12365735)

[3.1 Организация данных 11](file:///G:\шарага\ОАиП\Пример%20КП%20(full%20ver)\DOC\Пояснительная%20записка%20Баразна%2035ТП.docx#_Toc12365736)

[3.2 Концептуальный прототип 11](file:///G:\шарага\ОАиП\Пример%20КП%20(full%20ver)\DOC\Пояснительная%20записка%20Баразна%2035ТП.docx#_Toc12365737)

[3.3 Функции: логическая и физическая организация 14](file:///G:\шарага\ОАиП\Пример%20КП%20(full%20ver)\DOC\Пояснительная%20записка%20Баразна%2035ТП.docx#_Toc12365738)

[4 Описание программного средства 17](file:///G:\шарага\ОАиП\Пример%20КП%20(full%20ver)\DOC\Пояснительная%20записка%20Баразна%2035ТП.docx#_Toc12365739)

[4.1 Общие сведения 17](file:///G:\шарага\ОАиП\Пример%20КП%20(full%20ver)\DOC\Пояснительная%20записка%20Баразна%2035ТП.docx#_Toc12365740)

[4.2 Функциональное назначение 17](file:///G:\шарага\ОАиП\Пример%20КП%20(full%20ver)\DOC\Пояснительная%20записка%20Баразна%2035ТП.docx#_Toc12365741)

[4.3 Процедуры и функции 17](file:///G:\шарага\ОАиП\Пример%20КП%20(full%20ver)\DOC\Пояснительная%20записка%20Баразна%2035ТП.docx#_Toc12365742)

[4.4 Организация хранения данных 17](file:///G:\шарага\ОАиП\Пример%20КП%20(full%20ver)\DOC\Пояснительная%20записка%20Баразна%2035ТП.docx#_Toc12365743)

[5 Методика испытаний 18](file:///G:\шарага\ОАиП\Пример%20КП%20(full%20ver)\DOC\Пояснительная%20записка%20Баразна%2035ТП.docx#_Toc12365744)

[5.1 Технические требования 18](file:///G:\шарага\ОАиП\Пример%20КП%20(full%20ver)\DOC\Пояснительная%20записка%20Баразна%2035ТП.docx#_Toc12365745)

[5.2 Демонстрационный пример использования 18](file:///G:\шарага\ОАиП\Пример%20КП%20(full%20ver)\DOC\Пояснительная%20записка%20Баразна%2035ТП.docx#_Toc12365746)

[6 Применение 19](file:///G:\шарага\ОАиП\Пример%20КП%20(full%20ver)\DOC\Пояснительная%20записка%20Баразна%2035ТП.docx#_Toc12365747)

[6.1 Назначение программы 19](file:///G:\шарага\ОАиП\Пример%20КП%20(full%20ver)\DOC\Пояснительная%20записка%20Баразна%2035ТП.docx#_Toc12365748)

[6.2 Условия применения 19](file:///G:\шарага\ОАиП\Пример%20КП%20(full%20ver)\DOC\Пояснительная%20записка%20Баразна%2035ТП.docx#_Toc12365749)

[6.3 Справочная система 20](file:///G:\шарага\ОАиП\Пример%20КП%20(full%20ver)\DOC\Пояснительная%20записка%20Баразна%2035ТП.docx#_Toc12365750)

[Заключение 21](file:///G:\шарага\ОАиП\Пример%20КП%20(full%20ver)\DOC\Пояснительная%20записка%20Баразна%2035ТП.docx#_Toc12365751)

[Список используемых источников 22](file:///G:\шарага\ОАиП\Пример%20КП%20(full%20ver)\DOC\Пояснительная%20записка%20Баразна%2035ТП.docx#_Toc12365752)

[Приложение А 23](file:///G:\шарага\ОАиП\Пример%20КП%20(full%20ver)\DOC\Пояснительная%20записка%20Баразна%2035ТП.docx#_Toc12365753)

ВВЕДЕНИЕ

Тема психического здоровья последних лет как никогда актуальна. Наши родители — это дети, которые были воспитаны людьми, родители встретились своими глазами. Это не проходит бесследно, это все пагубно возникает на их психику, и даже после окончания войны многие годы большой проблемой является нехватка еды, одежды, совершение преступления и самое банальное место жительства. Основным заданием являлось восстановление архитектурных построений, налаживание развития хозяйства, работы заводов и т.д. По сути к людям относились как к рабочей силе людей, не волновало состояние их психики. В свою очередь это и привело к получению количества нестабильных людей.

В те времена к подобным проблемам относились с чувством и страхом. Поэтому к врачу никто не обращался, да и вряд ли в окружающих условиях люди сами задумывались о том, что у них что-то не в порядке, что и осуществляются к усугублению ситуации. Психически больной не может воспитать здорового человека. Это все передавалось из поколения в поколение. И только спустя почти 70 лет после окончания войны люди заговорили. Люди заговорили, не боясь быть отвергнутыми и отрицательными. Из-за этой популяризации в настоящее время начинаются контактные центры помощи людям, страдающим заболеваниями.

Данный проект представляет собой консольный вариант сайта для одного из таких центров. Из-за этой популяризации в настоящее время начинаются контактные центры помощи людям, страдающим заболеваниями. Данный проект представляет собой консольный вариант сайта для одного из таких центров.

Начало формы





Конец формы

Начало формы





Конец формы

# 1 Постановка задачи

## 1.1 Организационная сущность задачи

Целью данного курсового проекта является создание программного приложения на тему «Медицинские услуги (Регистратура ­­- управление прайс-листом, занятость манипуляционных/операционных)». Чтобы создать данную программу, необходимо исследовать предметную область.

Предметная область данной задачи будет включать в себя перечень доступных к записи специалистов, цены на их услуги и расписание работы.

Данная программа будет выполнять следующие функции:

* предоставлять потенциальным клиентам необходимую информацию;
* производить поиск;
* проводить анализ спроса.

В настоящее время подобные программные приложения широко используются крупными и не очень крупными медицинскими центрами. Дело в том, что это выгодно – иметь сайт своего стартапа. Так как большинство потребляемой информации последнее десятилетие сосредоточено в сети интернет.

* Конец формы

## 1.2 Функциональная модель

Для разрабатываемого программного продукта входной информацией будет служить ввод критериев для поиска и файлы с необходимой информацией.

Выходной информацией для данного проекта являются результат поиска по запросу, результат добавления специалиста и его образования, просмотр сохранённого отчета в файле.

Данный проект выполняет следующие основные функции:

* добавление специалиста и его образования;
* просмотр расписания специалистов;
* просмотр образования специалистов;
* просмотр услуг специалистов;
* просмотр справки.

Функциональная модель представлена на рисунке 1.1.

Справка

Измененные данные

Медицинские услуги (Регистратура ­­- управление прайс-листом, занятость манипуляционных/операционных)

Сведения о работе медицинского центра

Текстовые файлы

Информация, вводимая пользователем

Админ

Рисунок 1.1 – Функциональная модель

## 1.3 Входные данные

Входными данными будет являться информация, выбранная пользователем. Пользователь выбирает специалиста, информацию о котором необходимо узнать.

Администратору будет предоставлена возможность добавления данных о новых специалистах медицинского центра и их образовании, для этого ему потребуется ввести новые данные.

Входные данные для списка специалистов представлены в таблице 1.1.

Таблица 1.1 – Входные данные базы тур

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип поля | Размер поля | Назначение поля |
| showGreeting | void | 10 | Шапка сайта |
| chooseUserOrAdmin | int | 11 | Выбор роли |
| showUserText | void | 9 | Демонстрация возможных действий для пользователя |
| showZeroOptionText | void | 5 | Выход в главное меню + справка |
| performDoctorsTimetableAction | int | 38 | Расписание специалистов |
| performPrisesAction | Int | 40 | Стоимость на услуги специалистов |
| performDoctorsEducatoin | int | 5 | Образование специалистов |

## 1.4 Выходные данные

К выходным данным можно отнести информацию, которая предоставляется пользователю после добавления нового специалиста и его образования в базу.

Вид отчета представлен на рисунках 1.2, 1.3

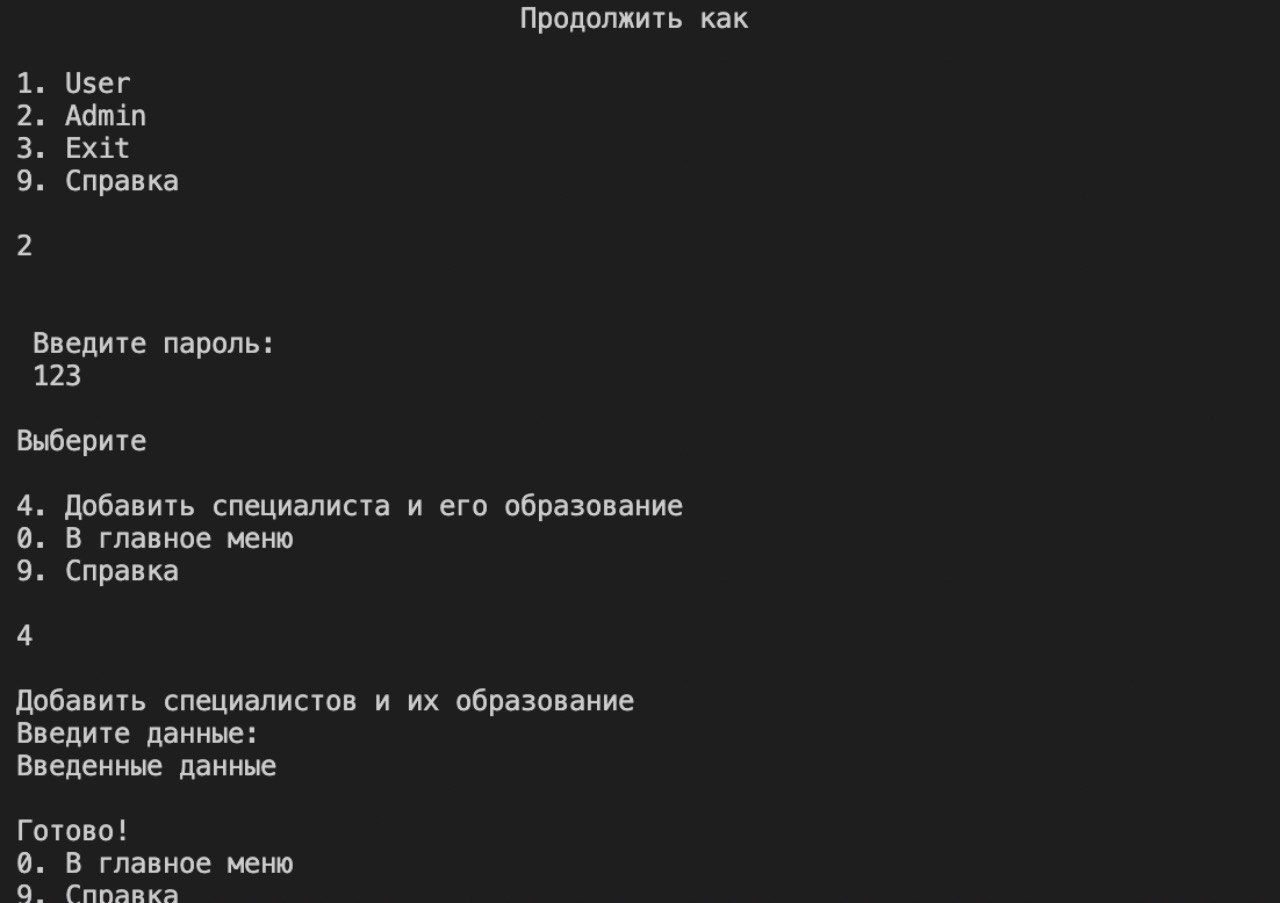


Рисунок 1.2 – Процесс изменения файла

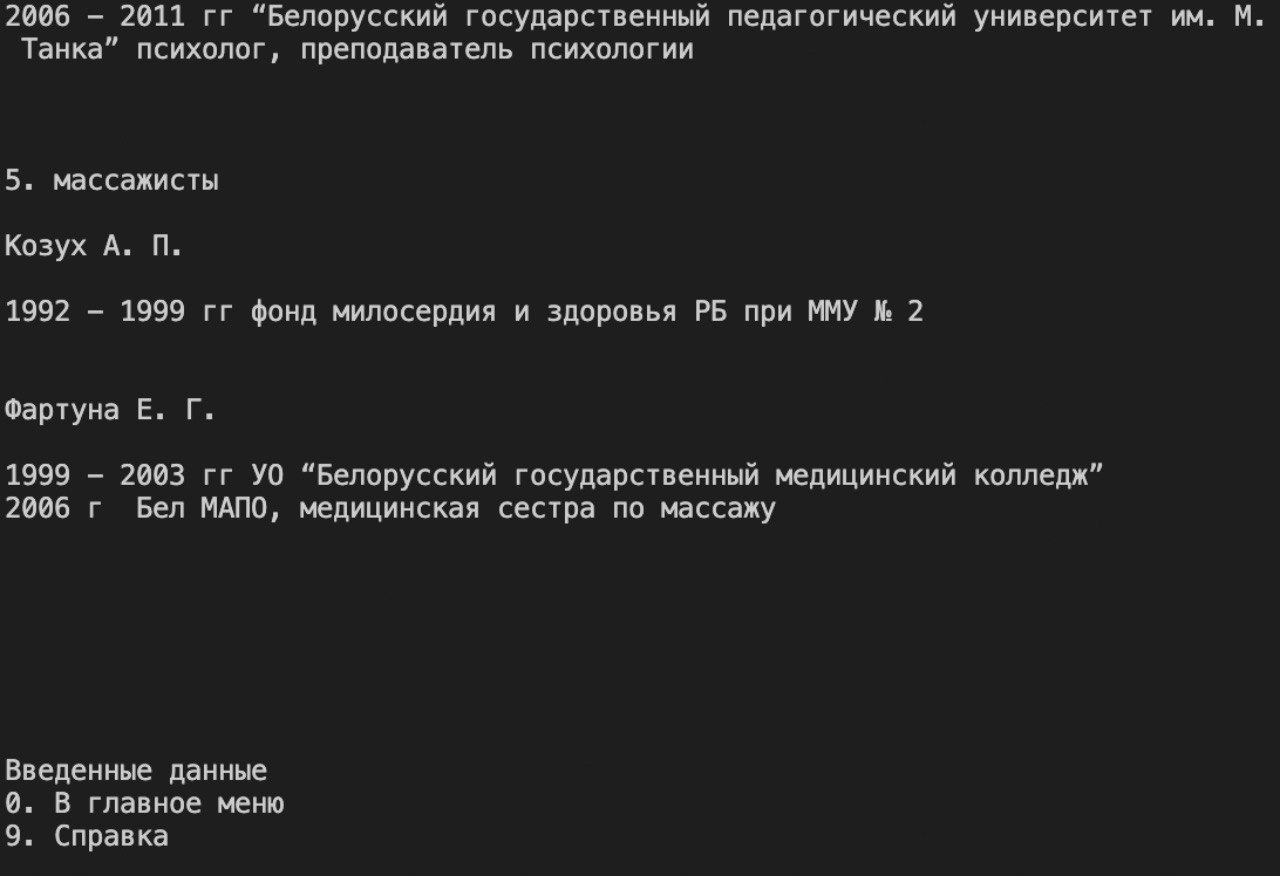


Рисунок 1.2 – Изменения

# 2 Вычислительная система

## 2.1 Используемые технические средства

Программное приложение не будет требовать инсталляции. Для того чтобы воспользоваться программой, необходимо скопировать каталог с программой на жесткий диск. Чтобы вынести ярлык программы на рабочий стол, необходимо нажать правой кнопкой мыши на «cw» и выбрать «Отправить» на «Рабочий стол (создать ярлык)».

Для нормальной работы программы будут необходимы следующие минимальные требования:

* процессор: Pentium IV и выше;
* объем оперативной памяти: 512 Мбайт и выше;
* свободного места на жестком диске: 100 Мбайт;
* операционная система: Windows 7 и выше;
* наличие .NET Framework 4.4 версии и выше;
* наличие манипулятора «мышь»;
* клавиатура IBM PC любой модификации.

## 2.2 Инструменты разработки

Данное программное приложение будет разрабатываться в операционной системе Microsoft Windows 10. Программа будет разрабатываться в среде VisualStudio Code и CLion.

Microsoft VisualStudio Code – линейка продуктов компании Microsoft, включающих интегрированную среду разработки программного обеспечения и ряд других инструментальных средств. Данные продукты позволяют разрабатывать как консольные приложения, так и приложения с графическим интерфейсом, в том числе с поддержкой технологии Windows Forms, а также веб-сайты, веб-приложения, веб-службы как в родном, так и в управляемом кодах для всех платформ, поддерживаемых Windows, Windows Mobile, Windows CE, .NET Framework, Xbox, Windows Phone .NET Compact Framework и Silverlight.

Visual Studio включает в себя редактор исходного кода с поддержкой технологии IntelliSense и возможностью простейшего рефакторинга кода. Встроенный отладчик может работать как отладчик уровня исходного кода, так и отладчик машинного уровня. Остальные встраиваемые инструменты включают в себя редактор форм для упрощения создания графического интерфейса приложения, веб-редактор, дизайнер классов и дизайнер схемы базы данных. Visual Studio позволяет создавать и подключать сторонние дополнения (плагины) для расширения функциональности практически на каждом уровне, включая добавление поддержки систем контроля версий исходного кода (как, например, Subversion и Visual SourceSafe), добавление новых наборов инструментов (например, для редактирования и визуального проектирования кода на предметно-ориентированных языках программирования) или инструментов для прочих аспектов процесса разработки программного обеспечения (например, клиент Team Explorer для работы с Team Foundation Server).

CLion — [интегрированная среда разработки](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D0%B3%D1%80%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D1%80%D0%B5%D0%B4%D0%B0_%D1%80%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B8) для [языков программирования](https://ru.wikipedia.org/wiki/%D0%AF%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F) C и [C++](https://ru.wikipedia.org/wiki/C%2B%2B), разрабатываемая компанией [JetBrains](https://ru.wikipedia.org/wiki/JetBrains). Состояние *CLion* — бесплатная пробная версия на 30 дней. Подходит для операционных систем [Windows](https://ru.wikipedia.org/wiki/Windows), [macOS](https://ru.wikipedia.org/wiki/MacOS), и [Linux](https://ru.wikipedia.org/wiki/Linux).

# 3 Проектирование задачи

## Организация данных

Исходя из исследования предметной области было принято решение о хранении данных в виде функций, ведь такой вид хранения предоставляет нам доступ к нужному полю структуры из любой точки кода. Были выделены следующие структуры данных:

Все данные будут хранится в виде базы данных как текстовый файл, запись в который будет производится при помощи одной из функций.

Чтение из файла производится специальной функцией, которая записывает строку в буфер и выводит ее на экран. Далее эта строка сохраняется как поле структуры, для дальнейшей обработки.

Доступ к произвольной записи в файле запрещен. Эта функция доступна только в случае авторизации админа.

Запись в файл-отчет производится автоматически вместе с просмотром результата в самой консоли.

## 3.2 Концептуальный прототип

Концептуальный прототип состоит из описания внешнего пользовательского интерфейса – системы меню, диалоговых окон и элементов управления. Программный продукт будет состоять из четырех модулей. Каждому модулю будет соответствовать свой пункт меню.

Меню:

* Расписание работы врачей.

Для начала пользователю предлагается выбрать специалиста, расписание которого необходимо просмотреть. После того, как сделан выбор, предоставляется возможность просмотреть расписание того или иного специалиста. А также возможность вернуться в главное меню и просмотреть справку. Пример выбора специалиста представлен на рисунке 3.1.

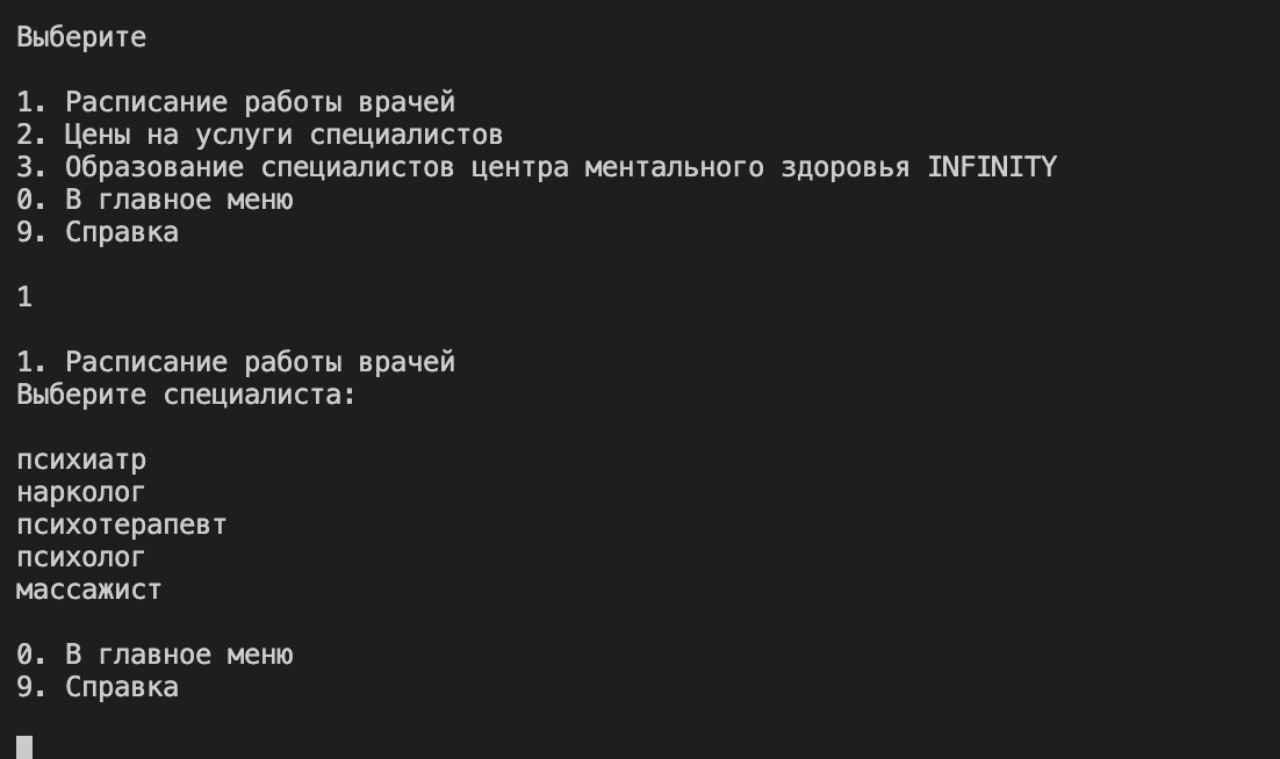


Рисунок 3.1 – Выбор специалиста.

* Цены на услуги специалистов

Для начала пользователю предлагается выбрать специалиста, услуги которого необходимо просмотреть. После того, как сделан выбор, предоставляется возможность просмотреть предоставляемые услуги того или иного специалиста. А также возможность вернуться в главное меню и просмотреть справку. Пример просмотра цен на услуги специалистов представлен на рисунке 3.2.

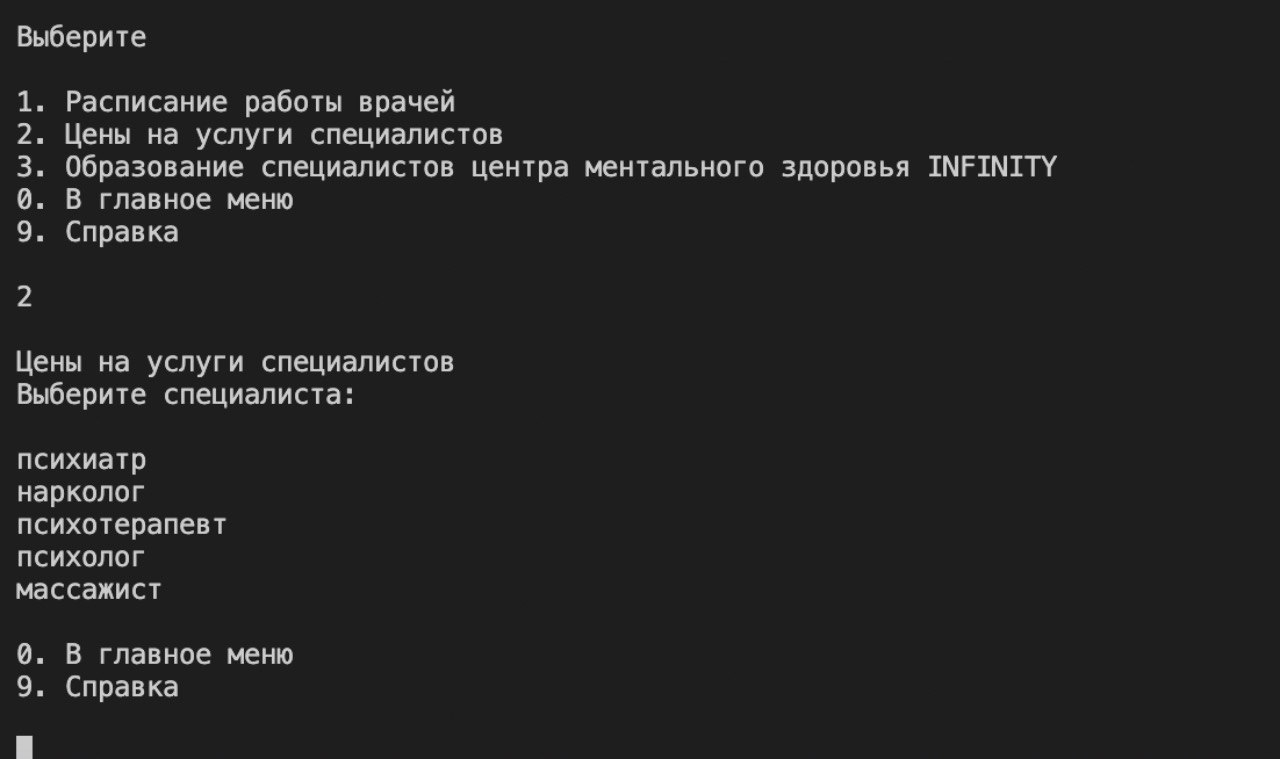


Рисунок 3.2 – Цены на услуги специалистов.

* Образование специалистов центра ментального здоровья INFINITY

Для начала пользователю предлагается выбрать специалиста, образование которого необходимо просмотреть. После того, как сделан выбор, предоставляется возможность просмотреть образование того или иного специалиста. А также возможность вернуться в главное меню и просмотреть справку. Пример выбора специалиста представлен на рисунке 3.1.

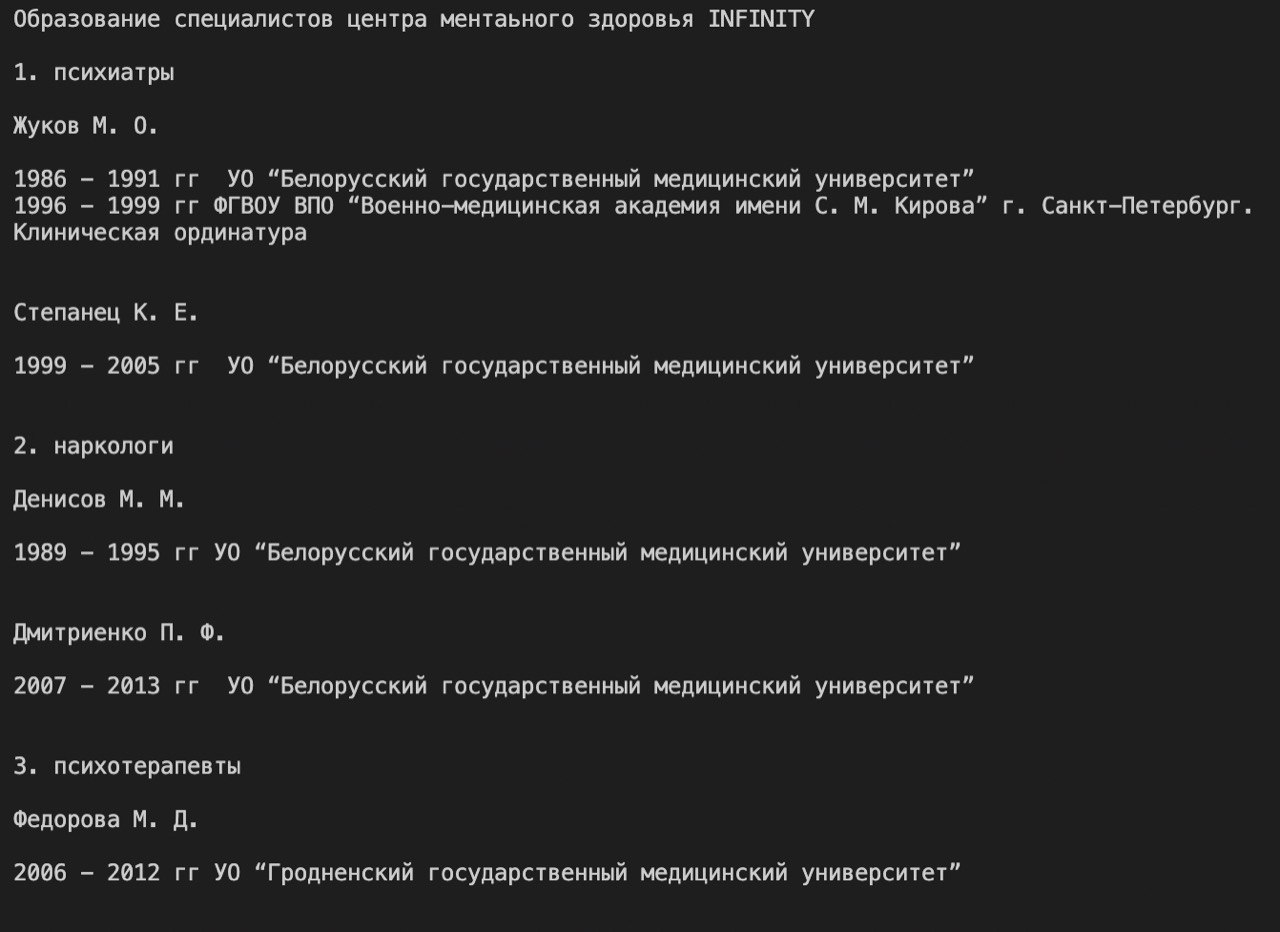


Рисунок 3.3 – Просмотр образования специалистов

* Добавление нового специалиста и его образования

Админу предлагается добавить новых специалистов и их образования в общий файл. Новая информация сохраняется и, при последующих просмотрах, видна пользователям. Пример выбора специалиста представлен на рисунке 3.4

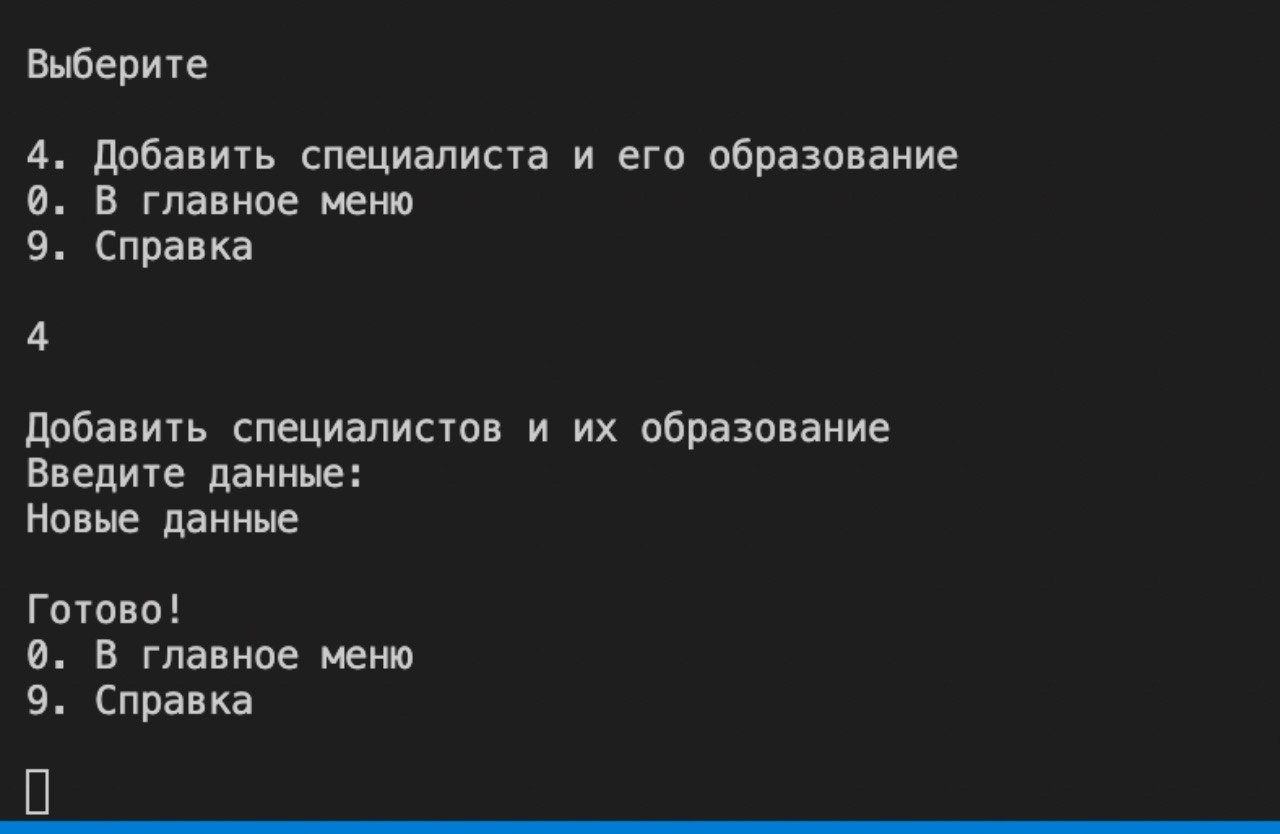


Рисунок 3.4 – Добавление нового специалиста и его образования

## 3.3 Функции: логическая и физическая организация

Функция выбора специалиста, расписание которого необходимо просмотреть закреплена за пунктом меню номер 1 «Расписание работы врачей». Код функции представлен ниже:

int performDoctorsTimetableAction(int x) {

switch (x) { //свич для файла с расписанием работы врачей

case 1:

cout << "психиатры\n";// здесь открывается файл с фамилиями и расписанием психиатров

readFile("/Users/coffin\_filling/Desktop/cw/файлы/2.txt");//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

showZeroOptionsText();

cin >> x;

break;

case 2:

cout << "наркологи\n";// здесь открывается файл с фамилиями и расписанием наркологов

readFile("/Users/coffin\_filling/Desktop/cw/файлы/3.txt");

showZeroOptionsText();

cin >> x;

break;

case 3:

cout << "психотерапевты\n";// здесь открывается файл с фамилиями и расписанием психотерапевтов

readFile("/Users/coffin\_filling/Desktop/cw/файлы/4.txt");

showZeroOptionsText();

cin >> x;

break;

case 4:

cout << "психологи\n";// здесь открывается файл с фамилиями и расписанием детских психологов

readFile("/Users/coffin\_filling/Desktop/cw/файлы/5.txt");

showZeroOptionsText();

cin >> x;

break;

case 5:

cout << "массажисты\n";// здесь открывается файл с фамилиями и расписанием массажистов

readFile("/Users/coffin\_filling/Desktop/cw/файлы/6.txt");

showZeroOptionsText();

cin >> x;

break;

}

return x;

}

Функция выбора специалиста, цены на услуги которого необходимо просмотреть закреплена за пунктом меню номер 2 «Цены на услуги специалистов». Код функции представлен ниже:

int performPricesAction(int x) {

switch (x) { // свич для файла с ценами на услуги специалистов

case 1:

cout << "Психиатры\n";

readFile("/Users/coffin\_filling/Desktop/cw/файлы/7.txt");

showZeroOptionsText();

cin >> x;

break;

case 2:

cout << "Наркологи\n";

readFile("/Users/coffin\_filling/Desktop/cw/файлы/8.txt");

showZeroOptionsText();

cin >> x;

break;

case 3:

cout << "Психотерапевты\n";

readFile("/Users/coffin\_filling/Desktop/cw/файлы/9.txt");

showZeroOptionsText();

cin >> x;

break;

case 4:

cout << "Психологи\n";

readFile("/Users/coffin\_filling/Desktop/cw/файлы/10.txt");

showZeroOptionsText();

cin >> x;

break;

case 5:

cout << "Массажисты\n";// для проверки админа услуги

readFile("/Users/coffin\_filling/Desktop/cw/файлы/11.txt");

showZeroOptionsText();

cin >> x;

break;

}

return x;

}

Функция просмотра образования специалистов закреплена за пунктом меню номер 3 «Образование специалистов центра ментального здоровья INFINITY». Код функции представлен ниже:

// выводим файл с образованием

int PerformDoctorsEducation(int x) {

readFile("/Users/coffin\_filling/Desktop/cw/файлы/12.txt");

цин >> х;

}

Полный код функций представлен в приложении А.

# 4 Описание программного средства

## 4.1 Общие сведения

Программное приложение на тему «Медицинские услуги» предназначена для облегчения процесса записи на прием.

Программа не требует инсталляции, необходимо ее только скопировать на компьютер.

Вызов и загрузка программы осуществляется после запуска исполняемого файла Main.cpp. Размер этого файла составляет 12 КБ.

Проект состоит из модулей, листинг которых представлен в Приложении А.

## 4.2 Функциональное назначение

Программное приложение предназначено для автоматизации ознакомления с услугами, предлагаемыми центром ментального здоровья INFINITY.

В программе реализованы функции просмотра расписания работы врачей, просмотра услуг, предоставляемых специалистами, просмотра образования специалистов, внесение информации об образовании новых специалистов.

Программа обладает интерфейсом понятным на интуитивном уровне как пользователю, не обладающему глубокими знаниями компьютерных технологий и навыками работы с персональным компьютером, так и опытному пользователю.

Для работы с программным приложением не требуется установка дополнительных компонентов. Это обусловлено тем, что данное программное приложение является простым в использовании.

Благодаря денной программе потенциальным клиентам медицинского центра ментального здоровья INFINITY.

## 4.3 Процедуры и функции

Для реализации поставленных задач было использовано большое количество процедур и функций. Основные функции, которые использовались при создании программного средства:

int chooseUserOrAdmin();

void showGreeting();

void showUserText();

int completeAdminAction();

void showZeroOptionsText();

int performDoctorsTimetableAction(int x);

int performPricesAction(int x);

int performDoctorsEducation(int x);

int performSpravka(int x);

void writeToFile(const std::string &fileName);

void readFile(const std::string &fileName);

## 4.4 Организация хранения данных

Все данные хранятся в одном файле, структуры (деревья) каталогов и файлов при создании данного ПС не использовались.

# 5 Методика испытаний

## 5.1 Технические требования

Для оптимальной работы программы необходимы следующие минимальные требования:

* процессор: Pentium IV и выше;
* объем оперативной памяти: 512 Мбайт и выше;
* свободного места на жестком диске: 100 Мбайт;
* операционная система: Windows 7 и выше;
* наличие .NET Framework 4.4 версии и выше;
* наличие манипулятора «мышь»;
* клавиатура IBM PC любой модификации.

## 5.2 Демонстрационный пример использования

Программное средство должно функционировать с определенными характеристиками для заданных областей данных. Отладкой называется процесс, позволяющий получить программу, функционирующую с требуемыми характеристиками в заданной области входных данных. В результате отладки программа должна соответствовать определенной фиксированной совокупности правил и показателей качества.

Для выявления логических ошибок необходимо провести полное тестирование программного приложения, всех его функций, реализованных в программе.

При запуске программы появляется первое окно с возможностью выбора пункта меню. Вид главного окна представлен на рисунке 5.1.

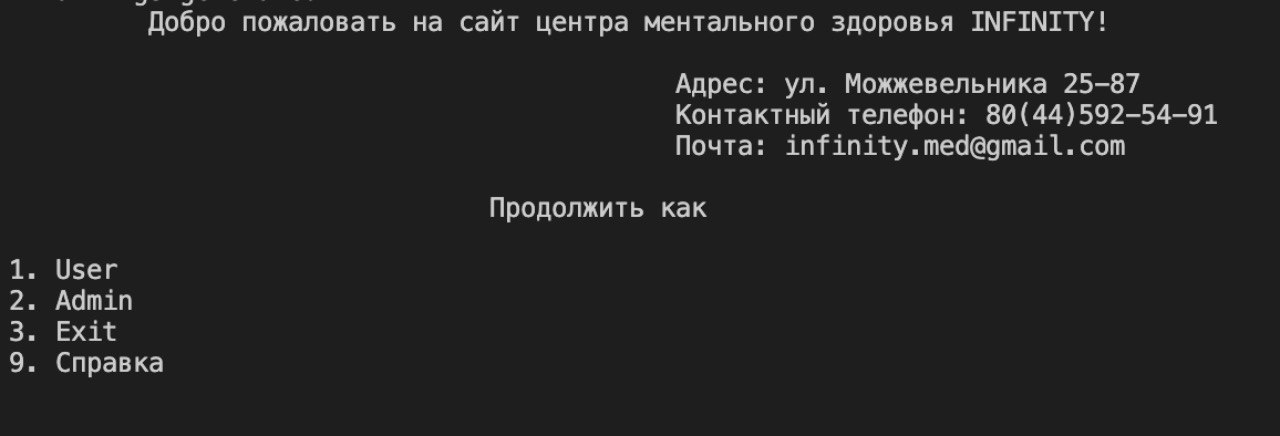


Рисунок 5.1 – Вид главного меню

Для того чтобы протестировать программное средство необходимо использовать один пример. При этом нужно пройти по всем пунктам меню.

Функции программы уже были протестированы и повторного тестирования не требуют.

# 6 Применение

## 6.1 Назначение программы

Разрабатываемое программное приложение предназначено для облегчения ознакомления пользователей с информацией, предоставляемой медицинским центром ментального здоровья INFINITY. Удобный и приятный интерфейс программы не вызовет затруднений в использовании даже у не опытного пользователя. Однако при необходимости предусмотрена справка, подробно описывающая основные функции программного приложения.

## 6.2 Условия применения

Для корректной работы программного средства требуются следующие минимальные требования к техническим средствам:

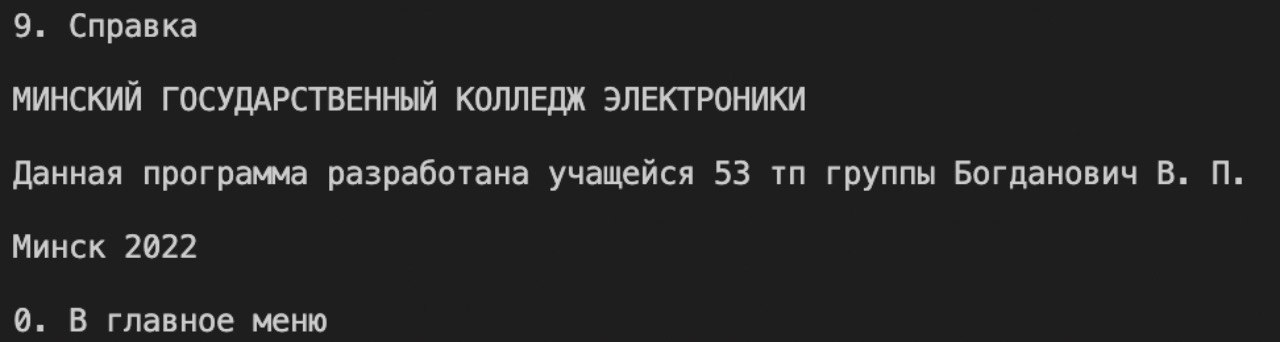
* процессор: Pentium IV и выше;
* объем оперативной памяти: 512 Мбайт и выше;
* свободного места на жестком диске: 100 Мбайт;
* операционная система: Windows 7 и выше;
* наличие .NET Framework 4.4 версии и выше;
* наличие манипулятора «мышь»;
* клавиатура IBM PC любой модификации.

Для того чтобы воспользоваться программой, необходимо скопировать каталог с программой на жесткий диск. Чтобы вынести ярлык программы на рабочий стол, необходимо нажать правой кнопкой мыши на файл «CW» и выбрать «Отправить» на «Рабочий стол (создать ярлык)».

## 6.3 Справочная система

Программное средство имеет интуитивно понятный и легкий в освоении интерфейс, а также несложную логическую организацию.

Так как есть возможность того, что пользователь не сможет понять, как нужно работать с программой в меню предусмотрен пункт «Справка». При нажатии этого пункта меню появиться справочная информация.



# ЗАКЛЮЧЕНИЕ

Задание на курсовое проектирование заключалось в написании программы на тему «Регистратура ­­- управление прайс-листом, занятость манипуляционных/операционных)». Реализовать поставленную задачу удалось при помощи среды программирования VisualStudio Code.

В процессе разработки было приложено максимум усилий для создания качественного и легкого в использовании программного продукта, который отвечал бы заданным требованиям.

Продукт содержит понятные и конкретные названия пунктов меню. Все это позволяет быстро и легко вникнуть в суть программы, разобраться в ее работе, а также быстро и эффективно выполнить необходимые действия. Это значит, что даже неопытный пользователь может применять в работе данное приложение.

Программа выполняет следующие функции:

* производит просмотр расписания специалистов;
* производит просмотр цен на услуги специалистов;

производит просмотр образования специалистов медицинского центра ментального здоровья INFINITY;

* позволяет вносить изменения в файл с информацией об образовании специалистов

Программа реализована полностью в соответствии с алгоритмом задачи, отлажена и протестирована.

# СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. ГОСТ 2.105-95. ЕСКД. Общие требования к текстовым документам
2. ГОСТ 19.401-2000. ЕСПД. Текст программы
3. ГОСТ 19.402-2000 ЕСПД. Описание программы
4. ГОСТ 19.301-2000 ЕСПД. Программа и методика испытаний

# ПРИЛОЖЕНИЕ А

(справочное)

Текст программы

Main:

int main() {

setlocale(LC\_ALL, "Russian");

showGreeting();

while (true) {

int x = chooseUserOrAdmin();

switch (x) { // свич для админ/юзер

case 1:

//user

showUserText();

cin >> x;

cout << "\n";

break;

case 2:

//admin

x = completeAdminAction();

break;

case 3:

exit(3);

} // конец свича для админ/юзер

switch (x) { // свич для админа и юзера: все возможные действия из первого списка

case 0:

do {

x = chooseUserOrAdmin();

switch (x) {

case 1:

showUserText();

cin >> x;

cout << "\n";

switch (x) {

case 0:

break;

case 1:

cout << "1. Расписание работы врачей\n";

showZeroOptionsText();

cin >> x;

x = performDoctorsTimetableAction(x);

break; // конец свича для файла с рассписанием работы врачей

//---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

case 2:

cout << "Цены на услуги специалистов\n"; // тут файл со списком специалистов

showZeroOptionsText();

cin >> x;

x = performPricesAction(x);

break;// конец свича для файла с ценами на услуги специалистов

case 3:

cout << "Образование специалистов центра ментального здоровья INFINITY\n"; // тут файл со списком специалистов

showZeroOptionsText();

cin >> x;

x = performDoctorsEducation(x);

break;// конец свича для файла с образованием специалистов центра ментального здоровья INFINITY

break;

}

break;

case 2:

x = completeAdminAction();

break;

case 3:

exit(3);

}

} while (x == 0);

break;

case 1:

cout << "1. Расписание работы врачей\n";

readFile("/Users/coffin\_filling/Desktop/cw/файлы/1.txt");

showZeroOptionsText();

cin >> x;

x = performDoctorsTimetableAction(x);

break; // конец свича для файла с рассписанием работы врачей

//---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

case 2:

cout << "Цены на услуги специалистов\n"; // тут файл со списком специалистов

readFile("/Users/coffin\_filling/Desktop/cw/файлы/1.txt");

showZeroOptionsText();

cin >> x;

x = performPricesAction(x);

break;// конец свича для файла с ценами на услуги специалистов

case 3:

cout << "Образование специалистов центра ментального здоровья INFINITY\n"; // тут файл со списком специалистов

readFile("/Users/coffin\_filling/Desktop/cw/файлы/12.txt");

showZeroOptionsText();

cin >> x;

x = performPricesAction(x);

break;

case 4:

cout << "Добавить специалистов и их образование\n";

writeToFile("/Users/coffin\_filling/Desktop/cw/файлы/12.txt");

showZeroOptionsText();

cin >> x;

break;

case 9:

cout << "9. Справка\n";

cout << "\n";

readFile("/Users/coffin\_filling/Desktop/cw/файлы/Spravka.txt");//read

cout << "\n";

showZeroOptionsText();

cin >> x;

x = performSpravka(x);

break;

break;

}//коннц свича для админа и юзера: все возможные действия из первого списка

}

}

Прототипы функций:

#include <iostream>

#include <string>

#include <fstream>

int chooseUserOrAdmin();

void showGreeting();

void showUserText();

int completeAdminAction();

void showZeroOptionsText();

int performDoctorsTimetableAction(int x);

int performPricesAction(int x);

int performDoctorsEducation(int x);

int performSpravka(int x);

void writeToFile(const std::string &fileName);

void readFile(const std::string &fileName);

Функции:

int completeAdminAction() {

int tmp;

cout << endl << " Введите пароль:\n ";

cin >> tmp;

cout << "\n";

if (tmp == 123) {

cout << "Выберите \n";

cout << "\n";

cout << "4. Добавить специалиста и его образование\n";

showZeroOptionsText();

cin >> tmp;

cout << "\n";

} else {

cout << "\n";

cout << "Неверный пароль\n";

cout << "\n";

showZeroOptionsText();

cin >> tmp;

}

return tmp;

}

void showGreeting() {

cout << " Добро пожаловать на сайт центра ментального здоровья INFINITY!\n";

cout << " \n";

cout << " Адрес: ул. Можжевельника 25-87\n";

cout << " Контактный телефон: 80(44)592-54-91\n";

cout << " Почта: infinity.med@gmail.com\n";

cout << "\n";

cout << " Продолжить как\n";

cout << "\n";

}

int chooseUserOrAdmin() {//шапка

int x = 0;

cout << "1. User\n";

cout << "2. Admin\n";

cout << "3. Exit\n";

cout << "9. Справка\n";

cout << "\n";

cin >> x;

cout << "\n";

return x;

}

void showUserText() {

cout << "Выберите\n";

cout << "\n";

cout << "1. Расписание работы врачей\n";

cout << "2. Цены на услуги специалистов\n";

cout << "3. Образование специалистов центра ментального здоровья INFINITY\n";

showZeroOptionsText();

}

void showZeroOptionsText() {

cout << "0. В главное меню\n";

cout << "9. Справка\n";

cout << "\n";

}

int performDoctorsTimetableAction(int x) {

switch (x) { //свич для файла с расписанием работы врачей

case 1:

cout << "психиатры\n";// здесь открывается файл с фамилиями и расписанием психиатров

readFile("/Users/coffin\_filling/Desktop/cw/файлы/2.txt");//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

showZeroOptionsText();

cin >> x;

break;

case 2:

cout << "наркологи\n";// здесь открывается файл с фамилиями и расписанием наркологов

readFile("/Users/coffin\_filling/Desktop/cw/файлы/3.txt");

showZeroOptionsText();

cin >> x;

break;

case 3:

cout << "психотерапевты\n";// здесь открывается файл с фамилиями и расписанием психотерапевтов

readFile("/Users/coffin\_filling/Desktop/cw/файлы/4.txt");

showZeroOptionsText();

cin >> x;

break;

case 4:

cout << "психологи\n";// здесь открывается файл с фамилиями и расписанием детских психологов

readFile("/Users/coffin\_filling/Desktop/cw/файлы/5.txt");

showZeroOptionsText();

cin >> x;

break;

case 5:

cout << "массажисты\n";// здесь открывается файл с фамилиями и расписанием массажистов

readFile("/Users/coffin\_filling/Desktop/cw/файлы/6.txt");

showZeroOptionsText();

cin >> x;

break;

}

return x;

}

int performPricesAction(int x) {

switch (x) { // свич для файла с ценами на услуги специалистов

case 1:

cout << "Психиатры\n";

readFile("/Users/coffin\_filling/Desktop/cw/файлы/7.txt");

showZeroOptionsText();

cin >> x;

break;

case 2:

cout << "Наркологи\n";

readFile("/Users/coffin\_filling/Desktop/cw/файлы/8.txt");

showZeroOptionsText();

cin >> x;

break;

case 3:

cout << "Психотерапевты\n";

readFile("/Users/coffin\_filling/Desktop/cw/файлы/9.txt");

showZeroOptionsText();

cin >> x;

break;

case 4:

cout << "Психологи\n";

readFile("/Users/coffin\_filling/Desktop/cw/файлы/10.txt");

showZeroOptionsText();

cin >> x;

break;

case 5:

cout << "Массажисты\n";// для проверки админа услуги

readFile("/Users/coffin\_filling/Desktop/cw/файлы/11.txt");

showZeroOptionsText();

cin >> x;

break;

}

return x;

}

// выводим файл с образованием

int performDoctorsEducation(int x) {

readFile("/Users/coffin\_filling/Desktop/cw/файлы/12.txt");

cin >> x;

}

int performSpravka(int x){

readFile("/Users/coffin\_filling/Desktop/cw/файлы/Spravka.txt");

showZeroOptionsText();

cin >> x;

}

void readFile(const std::string &fileName) {

string line;

ifstream myfile(fileName);

if (myfile.is\_open()) {

while (getline(myfile, line)) {

cout << line << '\n';

}

myfile.close();

}

}

void writeToFile(const std::string &fileName) {

ofstream foutput;

ifstream finput;

finput.open(fileName);

foutput.open(fileName, ios::app);

string str;

cout << "Введите данные: " << endl;

cin.ignore(1000, '\n');

std::getline(std::cin, str);

if (finput.is\_open()) {

foutput << "\n";

foutput << str;

}

cout << "\nГотово!\n";

finput.close();

foutput.close();

}