# Введение

Данный курсовой проект выполняется на тему «Разработка рабочего места парикмахера».

Информатизация в настоящее время затрагивает даже такие, казалось бы, далекие от ИТ сферы жизнедеятельности человека как оказание услуг парикмахерской, маникюрного салона, уход за животными и пр. Применение компьютера в таких сферах помогает сократить персонал, ускорить и упростить обработку данных рабочими, имеющими к сфере ИТ весьма посредственное отношение.

Целью курсового проекта является разработка программного продукта в среде Visual Studio на языке C#, позволяющее хранить данные клиентах парикмахерской, организовать учет рабочего времени и стоимости оказываемых услуг в базе данных, облегчить расчет зарплаты персонала и прибыли организации. Приложение помогает осуществлять поиск в таблицах базы данных, формировать и выводить отчеты о проделанной работе на экран компьютера, на печатающее устройство или сохранять на жесткий диск в различных форматах.

# 1 Постановка задачи

## 1.1 Описание предметной области

Программное средство разработано для облегчения учета клиентского потока на рабочем месте парикмахера, контроля получения денежных сумм, получения быстрой справочной информации по клиентам, их координатам, времени записи, формирования отчета о проделанной работе с указанием даты, времени и стоимости проделанной операции. Областью применения разрабатываемого приложения может являться парикмахерская работающая самостоятельно, либо как часть подразделения более крупной фирмы по оказанию услуг. Формат базы данных и интерфейса приложения позволит применять его так же в маникюрном салоне Автоматизация учета позволит упростить процесс контроля поступления денежных средств и учета рабочего времени работников.

## 1.2 Актуальность решаемой задачи

В настоящее в стране осуществляется интенсивное развитие рынка сферы услуг, применение вычислительной техники и в сфере услуг позволяет более эффективно использовать рабочее время сотрудников организации. Существуют программы, облегчающие учет денежных средств и количество оказываемых услуг, например такие как 1С и т.п., но они обладают высокой стоимостью при том, что большая часть такого программного средства будет невостребованной. За счет универсальности лицензионных программных средств работа с их помощью влечет за собой излишнюю нагрузку на пользователя. Применение программного средства, ориентированного на конкретную узкую задачу, позволяет как сэкономить на средствах на лицензионное отчисление, так и понизить нагрузку на оператора, позволяя ему не отвлекаться от своей основной деятельности.

## 1.3 Характеристика решаемой задачи

Полное наименование программного средства «Программное средство – рабочее место парикмахера».

Краткое наименование программного средства: «ПС рабочее место парикмахера».

Заказчик: Учреждение образования «Могилевский государственный политехнический колледж», город Могилев. От имени Заказчика – А.В.Карманов.

Разработчик программного обеспечения (в дальнейшем Исполнитель) – Ольга Евгеньевна Листопадова, учащаяся по специальности 2-40 01 01, «Программное обеспечение информационных технологий».

Основанием для проведения работ является задание на курсовое проектирование.

Информационная система для учета услуг будет выполнять следующие функции:

* хранение информации о клиентах;
* хранение информации типах оказываемых услуг;
* осуществление предварительной и основной записи клиентов;
* формирование отчета о клиентах.

Основными целями создания программного средства для учета являются:

* замещение ручного учета клиентов и работ, что уменьшит затраты времени;
* повышение качества выполняемой работы, что уменьшит риск ошибок при редактировании информации.

Система представляет собой базу данных, содержащую информацию о клиентах и их координатах, выполненных и планируемых работах, видах услуг в парикмахерской.

Объектом автоматизации является рабочее место парикмахера.

Видом автоматизируемой деятельности являются задачи учета произведенных и планируемых работ парикмахерского салона.

Критерием оценки достижения целей создания программного средства является упрощение ведения учета оказанных услуг.

Программное средство разрабатывается как однопользовательское программное обеспечение.

Наличие штата эксплуатационного и ремонтного персонала, ответственного за обеспечение функционирования информационных систем не требуется.

Структурой хранения данных будет являться собственная база данных. Клиентский интерфейс прикладной программы будет разрабатываться в Visual Studio 2019, поддерживающий интерфейс операционной системы Windows. Взаимодействие с пользователем будет осуществляться через диалоговые окна.

Система должна поддерживать целостность и сохранность базы данных, а также корректность вводимых данных и их наличие.

Требования к функциям выполняемой системы. Информационная система должна реализовывать следующие функции:

* добавлять данные о клиентах и услугах;
* производить редактирование информации о клиентах;
* формировать отчет о проделанной работе.

# 2 Проектирование программного продукта

## 2.1 Разработка модели данных

Функциональные требования к программному средству представлены на рисунке 1 в виде диаграммы вариантов использования.

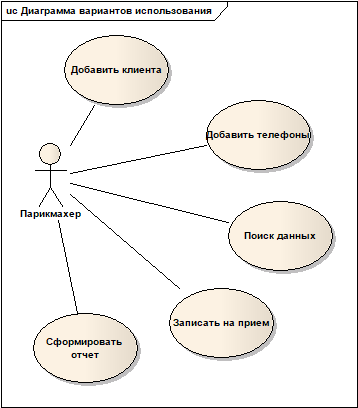


Рисунок 1 – Диаграмма вариантов использования

Описание вариантов использования представлено в таблице 1.

Таблица 1 – Реестр вариантов использования

|  |  |  |
| --- | --- | --- |
| Основной актер | Действие | Краткое описание |
| Парикмахер | Добавить телефоны | Ввод данных о телефонах клиента |
| Парикмахер | Добавить процедуру | Ввод данных о процедуре |
| Парикмахер | Записать на прием | Записать клиента на прием |
| Парикмахер | Искать данные в базе | Искать записи по датам или клиента по ФИО |
| Парикмахер | Вывести отчет по работам | Сформировать отчет по текущим работам |
| Парикмахер | Добавить клиента | Добавить данные о клиенте в базу данных |

Диаграмма классов форм приложения представлена на рисунке 2.

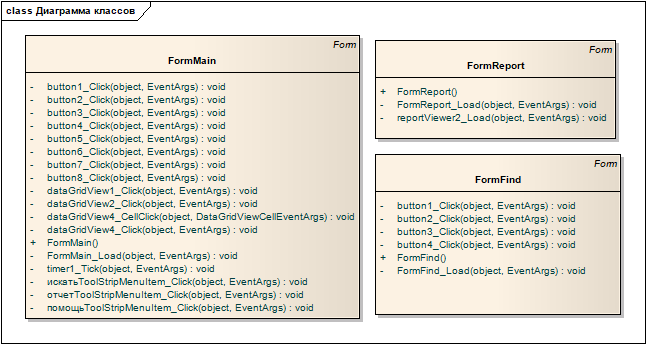


Рисунок 2 Диаграмма классов форм приложения

Диаграмма состояния приложения представлена на рисунке 3.

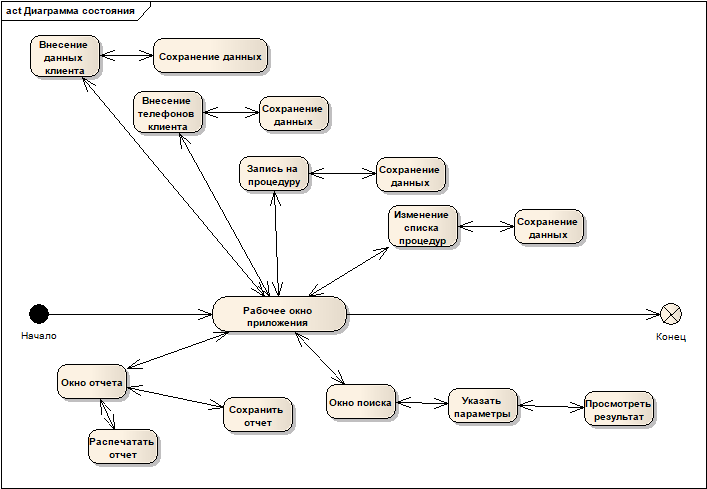


Рисунок 3. Диаграмма состояния приложения.

Диаграмма последовательности «Добавить клиента» представлена на рисунке 4.

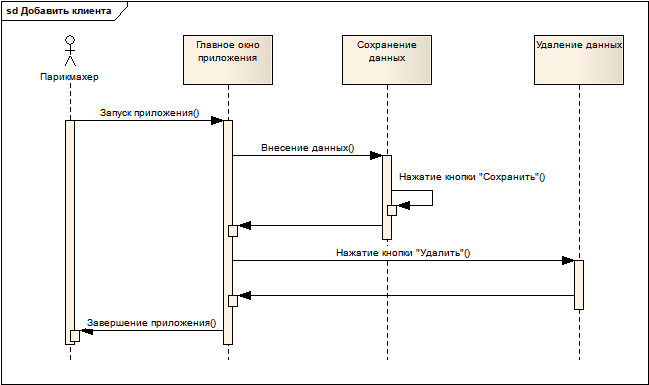


Рисунок 4. Диаграмма последовательности «Добавить клиента».

Таблица 2 - Описание процесса выполнения варианта использования

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Пользователь | Система | Экранная форма | Условие:  последующий шаг |
| 1 | Запуск программы | Запускается программа | Главная форма приложения | Ожидание действия |
| 2 | Внесение данных об клиенте | Ожидание действия | Главная форма приложения | Ожидание действия |
| 4 | Нажатие кнопки «Сохранить клиента» | Осуществляет сохранение данных в базе данных | Главная форма приложения | Ожидание действия |
| 5 | Нажатие кнопки «Удалить клиента» | Удаляет выделенную в табличной форме запись | Главная форма приложения | Ожидание действия |
| 6 | Нажатие кнопки закрытия главной формы программы или выбор пункта «Выход» главного меню | Завершает работу приложения |  |  |

Диаграмма последовательности «Добавить телефоны» представлена на рисунке 5.

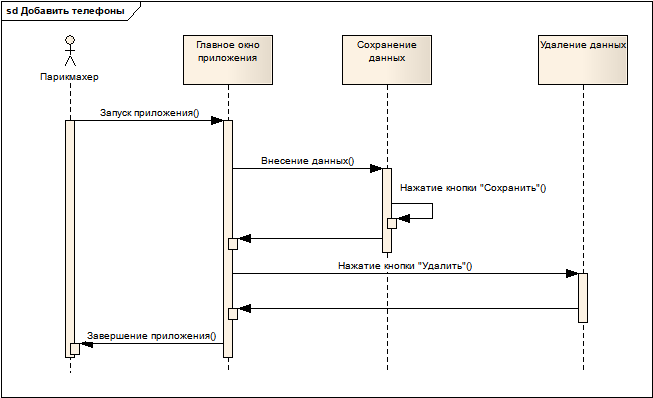


Рисунок 5. Диаграмма последовательности «Добавить телефоны».

Таблица 3 - Описание процесса выполнения варианта использования

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Пользователь | Система | Экранная форма | Условие:  последующий шаг |
| 1 | Запуск программы | Запускается программа | Главная форма приложения | Ожидание действия |
| 3 | Внесение данных о телефонах | Ожидание действия | Главная форма приложения | Ожидание действия |
| 4 | Нажатие кнопки «Добавить телефоны» | Осуществляет сохранение данных в базе данных | Главная форма приложения | Ожидание действия |
| 5 | Нажатие кнопки закрытия главной формы программы или выбор пункта «Выход» главного меню | Завершает работу приложения |  |  |

Диаграмма последовательности «Записать на прием» представлена на рисунке 6.

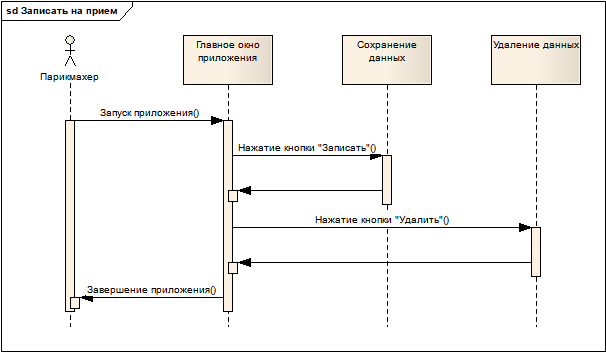


Рисунок 6. Диаграмма последовательности «Записать на прием».

Таблица 4 - Описание процесса выполнения варианта использования

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Пользователь | Система | Экранная форма | Условие:  последующий шаг |
| 1 | Запуск программы | Запускается программа | Главная форма приложения | Ожидание действия |
| 3 | Внесение данных о приеме | Ожидание действия | Главная форма приложения | Ожидание действия |
| 4 | Нажатие кнопки «Записать» | Осуществляет сохранение данных в базе данных | Главная форма приложения | Ожидание действия |
| 5 | Нажатие кнопки «Удалить» | Удаляет выделенную в табличной форме запись | Главная форма приложения | Ожидание действия |
| 6 | Нажатие кнопки закрытия главной формы программы или выбор пункта «Выход» главного меню | Завершает работу приложения |  |  |

Диаграмма последовательности «Поиск данных» представлена на рисунке 7.

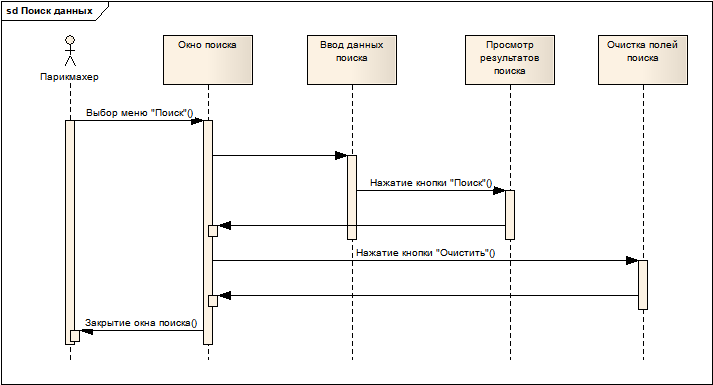


Рисунок 7. Диаграмма последовательности «Поиск данных».

Таблица 5 - Описание процесса выполнения варианта использования

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Пользователь | Система | Экранная форма | Условие:  последующий шаг |
| 1 | Запуск программы | Запускается программа | Главная форма приложения | Ожидание действия |
| 2 | Выбор пункта меню «Поиск» | Открывается форма «Поиск» | Форма приложения «Поиск» | Ожидание действия |
| 3 | Внесение данных о параметрах поиска | Ожидание действия | Форма приложения «Поиск» | Ожидание действия |
| 4 | Нажатие кнопки «Искать» | Осуществляет поиск в базе данных и отображает результаты в таблице | Форма приложения «Поиск» | Ожидание действия |
| 5 | Нажатие кнопки «Сбросить» | Очищает поля ввода экранной формы | Форма приложения «Поиск» | Ожидание действия |
| 6 | Нажатие кнопки закрытия формы | Закрывает форму «Поиск» | Главная форма приложения | Ожидание действия |
| 7 | Нажатие кнопки закрытия главной формы программы или выбор пункта «Выход» главного меню | Завершает работу приложения |  |  |

Диаграмма последовательности «Сформировать отчет» представлена на рисунке 8.

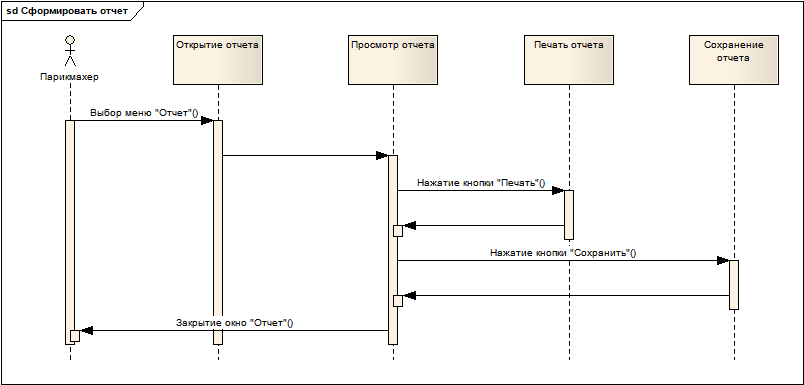


Рисунок 8. Диаграмма последовательности «Сформировать отчет».

Таблица 6 - Описание процесса выполнения варианта использования

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Пользователь | Система | Экранная форма | Условие:  последующий шаг |
| 1 | Запуск программы | Запускается программа | Главная форма приложения | Ожидание действия |
| 2 | Выбор пункта меню «Отчет» | Открывается форма «Отчет» | Форма приложения «Отчет» | Ожидание действия |
| 3 | Нажатие кнопки «Печать» | Осуществляет печать отчета на принтере | Форма приложения «Отчет» | Ожидание действия |
| 4 | Нажатие кнопки «Экспорт» | Осуществляет сохранение отчета на диск | Форма приложения «Отчет» | Ожидание действия |
| 5 | Нажатие кнопки закрытия формы | Закрывает форму «Отчет» | Главная форма приложения | Ожидание действия |
| 6 | Нажатие кнопки закрытия главной формы программы или выбор пункта «Закрыть» главного меню | Завершает работу приложения |  |  |

## 2.2 Выбор программного обеспечения

C# (произносится как "си шарп") — современный объектно-ориентированный и типобезопасный язык программирования. C# позволяет разработчикам создавать разные типы безопасных и надежных приложений, выполняющихся в .NET. C# относится к широко известному семейству языков C, и покажется хорошо знакомым любому, кто работал с C, C++, Java или JavaScript. Здесь представлен обзор основных компонентов языка C# 8 и более ранних версий. Если вы хотите изучить язык с помощью интерактивных примеров, рекомендуем поработать с вводными руководствами по C#.

C# — объектно-ориентированный, ориентированный на компоненты язык программирования. C# предоставляет языковые конструкции для непосредственной поддержки такой концепции работы. Благодаря этому C# подходит для создания и применения программных компонентов. С момента создания язык C# обогатился функциями для поддержки новых рабочих нагрузок и современными рекомендациями по разработке ПО. В основном C# — объектно-ориентированный язык. Вы определяете типы и их поведение.

Вот лишь несколько функций языка C#, которые позволяют создавать надежные и устойчивые приложения. Сборка мусора автоматически освобождает память, занятую недостижимыми неиспользуемыми объектами. Типы, допускающие значение null, обеспечивают защиту от переменных, которые не ссылаются на выделенные объекты. Обработка исключений предоставляет структурированный и расширяемый подход к обнаружению ошибок и восстановлению после них. Лямбда-выражения поддерживают приемы функционального программирования. Синтаксис LINQ создает общий шаблон для работы с данными из любого источника. Поддержка языков для асинхронных операций предоставляет синтаксис для создания распределенных систем. В C# имеется Единая система типов. Все типы C#, включая типы-примитивы, такие как int и double, наследуют от одного корневого типа object. Все типы используют общий набор операций, а значения любого типа можно хранить, передавать и обрабатывать схожим образом. Более того, C# поддерживает как определяемые пользователями ссылочные типы, так и типы значений. C# позволяет динамически выделять объекты и хранить упрощенные структуры в стеке. C# поддерживает универсальные методы и типы, обеспечивающие повышенную безопасность типов и производительность. C# предоставляет итераторы, которые позволяют разработчикам классов коллекций определять пользовательские варианты поведения для клиентского кода.

## 2.3 Определение требований к техническим средствам

Для разрабатываемой программы рекомендуется использовать аппаратно-программные средства со следующими характеристиками:

1) тип центрального процессора Intel Celeron (Pentium) или AMD Athlon:

- число ядер не менее 2;

2) системная плата с параметрами:

- форм-фактор - ATX;

- минимальный объем оперативной памяти - 4 Гб;

4) типы внешней памяти компьютера:

- минимальный объем жесткого диска: 100 ГБ;

6) параметры монитора:

- тип монитора - ЖК;

- ширина диагонали экрана – 17 и более дюймов;

- разрешение по горизонтали и вертикали – не менее 1024x768, желательно 1280х1024 и более;

Диаграмма компонентов представлена на рисунке 11.

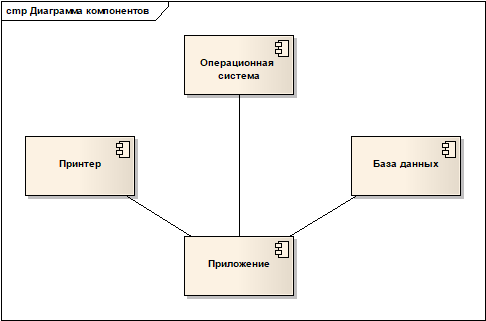


Рисунок 11 – Диаграмма компонентов

# 3 Реализация программного модуля

## 3.1 Определение формы представления входных и выходных данных

При разработке программного модуля была выбрана модель, при которой в главном окне приложения отображается сводная информация по клиентам и приемам, а выполнение дополнительных действий, таких как просмотр и печать отчетов, поиск в базе данных производятся в отдельных окнах приложения. Форма приложения разработана так, чтобы облегчить пользователю изучение и работу с приложением. На форме присутствуют поля ввода данных, поле отображения уже внесенной в таблицу информации и указания записи для удаления, и кнопки сохраняющие данные в базу, очищающие поля ввода и удаляющие запись, указанную в табличной части окна соответственно.

В качестве выходных форм, помимо сводной главной формы приложения выступают формы поиска и отчета. Форма поиска позволяет осуществить быстрый поиск в базе данных записей по указанным критериям, а форма отчета – сформировать отчет и вывести его, при необходимости, на печать. Вид форм и отчетов размещен в приложении Б.

## 3.2 Тестирование программного средства

Результаты тестирования приложения в различных режимах работы представлены в виде таблицы (таблица 7).

Таблица 7 – Результаты тестирования

|  |  |  |
| --- | --- | --- |
| Действие | Ожидаемый результат | Примечание |
| Главная форма приложения | | | |
| Запуск приложения | Отображение главной формы приложения с заполненными таблицами из базы данных | Выполнено |
| Выбор приема в таблице приемов | Отображение в связанных таблицах соответствующей информации о клиенте | Выполнено |
| Выбор пункта меню «Отчет» | Открытие формы отчета, содержащей сформированный отчет по путевкам | Выполнено |
| Выбор пункта меню «Поиск» | Открытие формы поиска | Выполнено |
| Выбор пункта меню «Справка» | Открытие справочной системы приложения | Выполнено |
| Работа с базой данных | | | |
| Нажатие кнопки «Сохранить клиента» | Внесение записи в базу данных, с проверкой пустых полей | Выполнено |
| Нажатие кнопки «Добавить телефоны» | Внесение записи в базу данных, с проверкой пустых полей | Выполнено |
| Нажатие кнопки «Удалить» | Удаление записи выделенной в табличной части формы | Выполнено |
| Нажатие кнопки закрытия формы | Закрытие главной формы приложения | Выполнено |
| Форма поиска | | | |
| Нажатие кнопки «Поиск» | Вывод результатов поиска в табличной части формы, в соответствии с указанными критериями | Выполнено |
| Нажатие кнопки «Стереть» | Очистка полей ввода поисковой строки | Выполнено |
| Нажатие кнопки закрытия формы | Закрытие текущей формы и возврат в главную форму приложения | Выполнено |
| Форма отчета | | | |
| Нажатие кнопки «Печать» | Вывод отчета на притер | Выполнено |
| Нажатие кнопки «Экспорт» | Сохранение отчета в указанном формате | Выполнено |
| Нажатие кнопки закрытия формы | Закрытие текущей формы и возврат в главную форму приложения | Выполнено |

## 3.3 Защита информации

Защита компьютеров от НСД является одной из основных проблем защиты информации, поэтому в большинство операционных систем и популярных пакетов программ встроены различные подсистемы защиты от НСД. Например, выполнение аутентификации пользователей при входе в операционные системы семейства Windows. Однако, не вызывает сомнений тот факт, что для серьезной защиты от НСД встроенных средств операционных систем недостаточно. К сожалению, реализация подсистем защиты большинства операционных систем достаточно часто вызывает нарекания из-за регулярно обнаруживаемых уязвимостей, позволяющих получить доступ к защищаемым объектам в обход правил разграничения доступа. Выпускаемые же производителями программного обеспечения пакеты обновлений и исправлений объективно несколько отстают от информации об обнаруживаемых уязвимостях.

Защита программы заключается в защите доступа к базе данных и осуществляется при необходимости соответствующей настройкой файла базы данных.

## 3.4 Разработка справочной системы

Справочная система - важная составная часть любой более-менее сложной (и даже простой) программы. Любое профессиональное приложение для операционной системы Windows должно содержать справочную систему стандартного вида, позволяющую пользователю получить дополнительную информацию по любому вопросу. Если внешний вид и принципы работы справочной системы будут отличаться от принятых в среде Windows, то это означает, что пользователь будет вынужден затратить дополнительное время на обучение.

Справочная система приложения подробно описывает пользователю основные принципы работы с программной, предоставляет ее описание и особенности. Справочная система приложения вызывается нажатием клавиши F1 на клавиатуре, либо выбором в главном меню приложения. При вызове справочной системы открывается справочное окно, состоящее из 2х частей – в левой можно выбрать интересующий раздел описания программы, в правой отображается информация об этом разделе, либо приветственное окно при первом запуске.

# 4Энерго- и ресурсосбережение

Стандарт управления энергопотреблением компьютеров, описывает три различных режима работы компьютера: ждущий, спящий, основной. Эти режимы отличаются потребляемой мощностью электроэнергии.

Ждущий режим позволяет экономить энергию, за счет отключения периферийных устройств. Спящий режим позволяет экономить электроэнергию на 100%, так как отключаются все устройства кроме ЦПУ. При работе в основном режиме экономии нет. Потребляемая мощность около 400 Ватт.

Для нахождения количества рабочих дней, в течение которых разрабатывался программный продукт, используем формулу

(1)

где Тпк – время работы компьютера, ч;

Тпк =201,4 ч;

tрп – суммарное время регламентированных перерывов, в течение

рабочего дня, ч;

tрп =50 мин=5/6 часа.

Для нахождения суммарной продолжительности регламентированных перерывов в течение всего времени разработки программного модуля

(2)

где Трп – суммарная продолжительность регламентированных перерывов в течение всего времени разработки программного модуля, ч.

Экономия электроэнергии рассчитывается по формуле

(3)

где Wпк – потребляемая мощность ПК, кВт;

Wсп – потребляемая мощность компьютера в ждущем режиме, кВт;

Wпк = 0,4 кВт;

Wсп = 0,16 кВт;

Сэ – стоимость 1 кВт электроэнергии, руб.

Таким образом сумма сэкономленной электроэнергии за время разработки программного обеспечения составила 1,73 рубля.

# Заключение

Результатом выполнения курсового проекта является разработанное программное средство для автоматизации учета клиентов и проделанной работы парикмахера, позволяющее хранить данные в базе и осуществлять управление, поиск информации и печать отчетов о проделанной работе.

Программа выполнена в соответствии с техническим заданием.

Достоинствами данной программы являются:

* простой и удобный интерфейс;
* возможность редактирования, добавления, удаления, изменения данных.

Разработанная программа нетребовательна к ресурсам компьютера, обладает интуитивно-понятным интерфейсом, проста и удобна в использовании.

# Список используемых источников

1. ГОСТ ИСО/ МЭК 2382-99. Информационные технологии. Словарь. Ч. Основные термины.
2. ГОСТ 34.602 – 89 Комплекс стандартов на автоматизированные системы. Техническое задание на создание автоматизированной системы
3. СТУ СМК 4.04-2011. Стандарт учреждения. Общие требования к оформлению текстовых документов».
4. Буч, Г. Язык UML. Руководство пользователя / Г. Буч, Д. Рамбо, И. Якобсон; Пер. с англ. Мухин Н. – 2-е изд. – М.: ДМК Пресс, 2007. – 496 с.
5. Рамбо, Дж. UML 2.0. Объектно-ориентированное моделирование и разработка / Рамбо Дж., М. Блаха. – 2-е изд. – СПб.: Питер, 2007. – 554 с. (Библиотека программиста)
6. Фролов, А. В. Язык С#. Самоучитель. / Г. В. Фролов, А. В. Фролов - М.: ДИАЛОГ-МИФИ, 2003. - 560 с.
7. Шилдт Г. С# 4.0: полное руководство / Г. Шилдт. – М.: ООО «И.Д. Вильямс», 2011. -1056 с.: ил.