

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«Петрозаводский государственный университет»  
Физико-технический институт  
Кафедра информационно-измерительных систем и физической электроники

Отчет по командному заданию  
Дисциплина: «Технология программирования»

Выполнили студенты: 3 курса  
физико-технического института,  
группы 21312  
Бутырев Никита Сергеевич  
Стахивич Елизавета Александровна

Научный руководитель:  
кандидат физико-математических наук,  
Бульба Артем Владимирович

Цель работы:

Разработать программу для заказчика (в нашем случае заказчиком является владелец студии маникюра), которая могла бы: выполнять ввод новых данных о записях, производить редактирование уже имеющихся данных и выводить списки(таблицы) с необходимой информацией.

Кратко о программной реализации:

Среда разработки: Microsoft Visual Studio 13.

Язык: C++.

Перечисление всех созданных заголовочных файлов и единиц компиляции:

Заголовочные файлы:

Record.h -Содержит информацию о записи клиента

RecordList.h – Содержит список записей

GlobalFunc.h -Содержит глобальные функций

Menu.h – Содержит в себе главное меню.

Исходные файлы:

GlobalFunc.cpp – Содержит в себе реализацию глобальных функций

Record.cpp – Содержит в себе методы класса Record

RecordList.cpp – Содержит в себе данные о всех записях, так же методы для поиска записей по номеру, удаления и добавления новых, их вывода

Menu.cpp – Реализация методов главного меню, с помощью которых происходит взаимодействие с пользователем

Source.cpp – Содержит в себе главную функцию main

## 1. Краткое словесное описание сюжета

К нам поступил заказ от владельца студии маникюра написать программу, которая упростила бы запись клиентов на сеанс. Таблица записей должны содержать следующие данные: Дату, время, ФИО, номер телефона клиента, имя мастера, вид услуги и стоимость. Программа должна иметь возможность ввода, вывода данных, редактирования и удаления записей.

## 2. Описание имеющихся у заказчика материалов (данных), с которыми ему приходится работать

Таблица 1. Общий вид таблицы

Дата (00.00.0000)	Время(00:00)	Фамилия	Имя	Отчество	Контактный телефон	Мастер	Вид услуги	Стоимость услуги
14.12.2021	14:00	Осипова	Марина	Анатольевна	89004594344	Екатерина	дизайн	1000
15.12.2021	16:30	Есимова	Алена	Вячеславовна	89534423111	Елизавета	дизайн	1000

## 3. Необходимо перечислить действующих субъектов (actors) и их функциональные обязанности

Администратор производит запись клиентов, в его обязанности входят внесение записей и их редактирование.

4. Представить список вариантов использования, которые могут возникнуть при работе программы

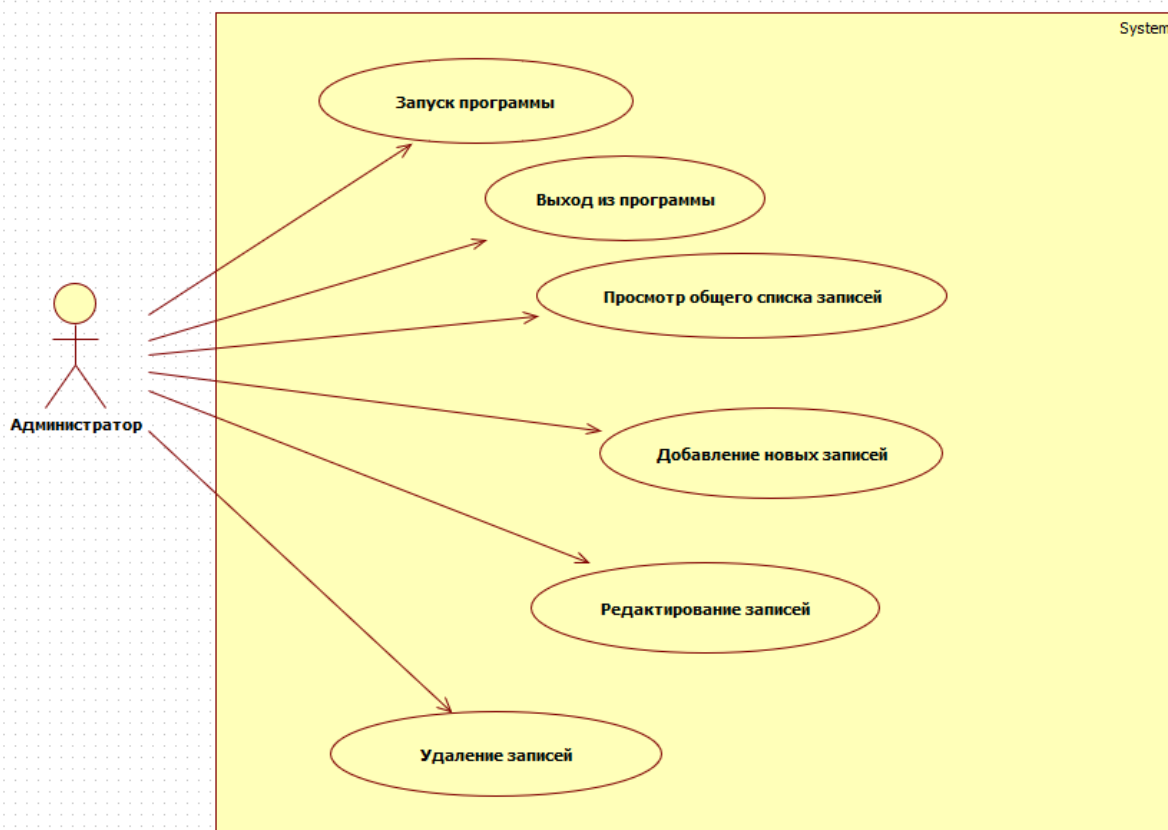


Рис.1 – Список вариантов использования, которые могут выполняться в проекте

5. Каждый вариант использования должен быть отдельно описан текстом сценария

Вариант использования – Запустить программу

Пользователь инициализирует запуск программы и переходит в меню.

Вариант использования – Просмотр общего списка записей

Пользователь получает от системы таблицу данных, содержащую всю информацию о введенных записях.

Вариант использования – Добавление данных в таблицу.

Пользователь вносит в систему данные о записи, а именно: Дату, Время, ФИО, телефон, имя мастера, вид услуги, стоимость, после чего система запоминает и хранит данные.

Вариант использования – Редактирование данных:

Пользователь имеет возможность изменить данные. Для этого ему необходимо ввести соответствующий id. После это предоставляется возможность изменить все данные, указанные в вышеупомянутом варианте использования.

Вариант использования – Удаление записи.

Пользователь имеет возможность удалить запись. Для этого он вводит соответствующий id, после чего система удаляет запись.

Вариант использования – Выход.

Пользователь по нажатию на клавишу завершает работу программы и выходит из нее.

6. Не менее двух вариантов использования опишите диаграммами действий и приложите диаграммы к отчету

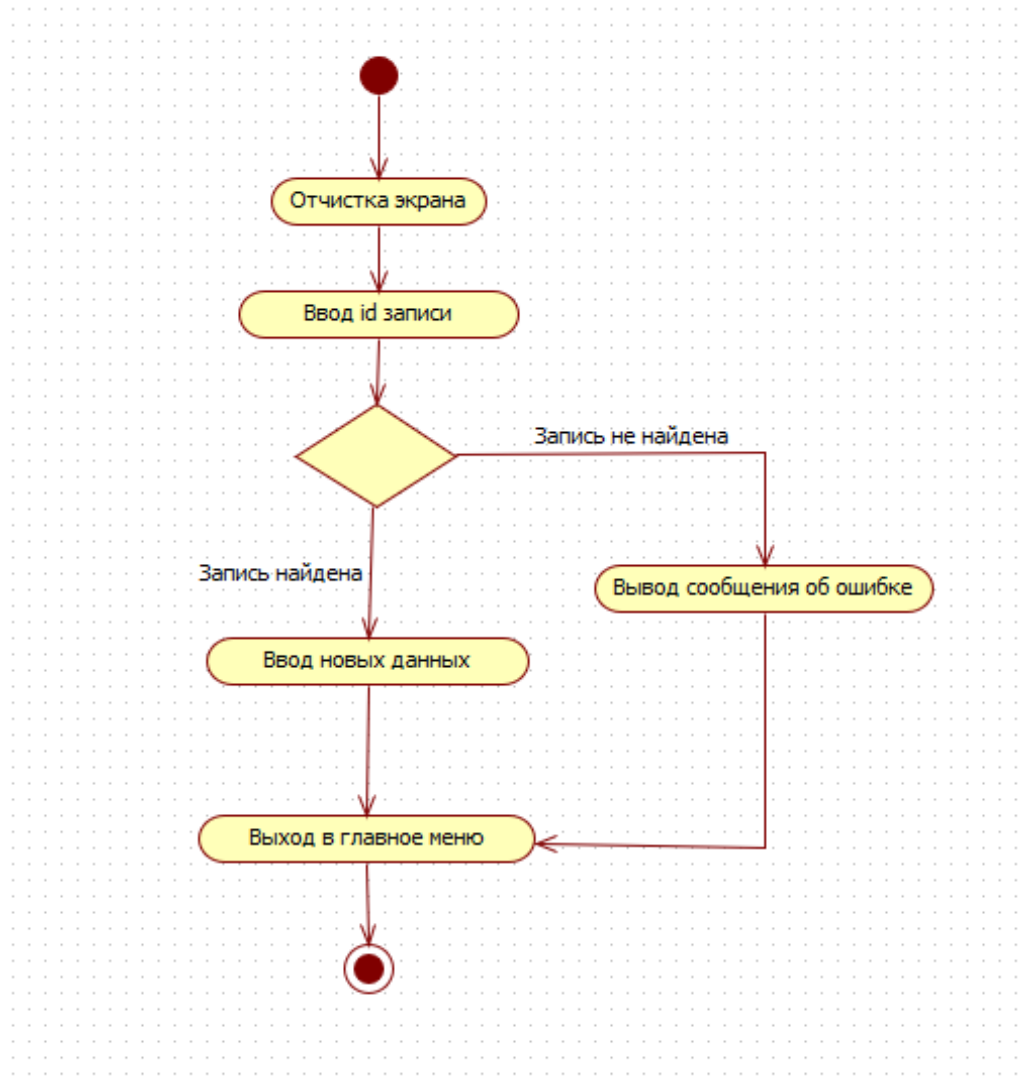


Рис. 2 – Диаграмма действия: редактирование

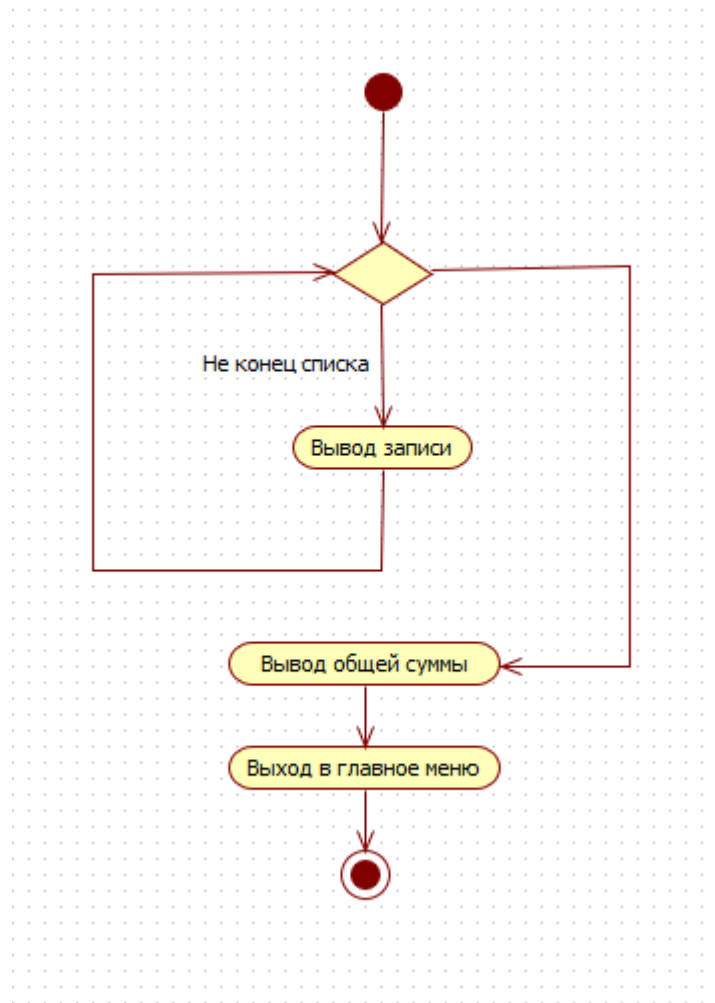


Рис.3 – Диаграмма действия: вывод таблицы нарушителей

7. Из описаний вариантов использования необходимо составить предварительный список существительных, которые, возможно, станут классами программы.

- 1) Главное меню
- 2) Экран ввода данных
- 3) Фамилия
- 4) Имя
- 5) Отчество
- 6) Дата
- 7) Время
- 8) Имя мастера
- 9) Вид услуги
- 10) Стоимость

- 11) Экран редактирования данных
- 12) id
- 13) Список записей
- 14) Выход из программы
- 15) Запись
- 16) Администратор
- 17) Экран удаления записи

## 8. Список классов

- 1) Record (Запись)
- 2) RecordList (Список записей)
- 3) Menu (Основной интерфейс)

## 9. По каждому классу составьте список атрибутов с пояснением предназначения каждого. Список атрибутов приложите к отчёту

### 1) Record:

id; //айди/номер  
day; //день  
month; //месяц  
year; //год  
hour; //час  
min; //минута  
lname; //фамилия  
fname; //имя  
patron; //отчество  
number; //номер телефона  
master; //имя мастера  
type; //вид услуги  
price; //цена

### 2) RecordList:

list<Record\*> ptrRecord; //контейнер с указателями на экземпляры класса запись

list<Record\*>::iterator iter; //итератор

### 3) Menu:

RecordList\* ptrRecordList; // указатель на список записей



10. Проанализируйте описания вариантов использования для выяснения того, какими сообщениями будут обмениваться классы.

1) Record:

GetId(); получение id

GetDay(); получение дня

GetMonth(); получение месяца

GetYear(); получение года

GetHour(); получение часа

GetMin(); получение минут

GetLname(); получение фамилии

GetFname(); получение имени

GetPatron(); получение отчества

GetNumber(); получение номера телефона

GetMaster(); получение имени мастера

GetType(); получение типа услуги

GetPrice(); получение стоимости

SetId(int sid); установка id

SetDay(int sday); установка дня

SetMonth(int smonth); установка месяца

SetYear(int syear); установка года

SetHour(int shour); установка часа

SetMin(int smin); установка минут

SetLname(string slname); установка фамилии

SetFname(string sfname); установка имени

SetPatron(string spatron); установка отчества

SetNumber(long long snumber); установка номера телефона

SetMaster(string smaster); установка имени мастера

SetType(string stype); установка типа услуг

SetPrice(int sprice); установка стоимости

Print();//вывод полей всех полей

2) RecordList:

FindId(int sid); метод нахождения элемента в списке по id

InsertRecord(Record\*) Введение нового элемента в список

Display() вывод списка

Delete(int sid) удаление элемента с id = аргументу

3) Menu:

Menu() выделение памяти для элементов программы

~Menu() освобождение памяти

InputRecord()	интерфейс ввода новой записи
EditRecord()	интерфейс редактирование записи
DeleteRecord()	интерфейс удаление записи
DisplayList()	вывод списка
Update()	обновление интерфейса

## 11. Построить диаграмму классов

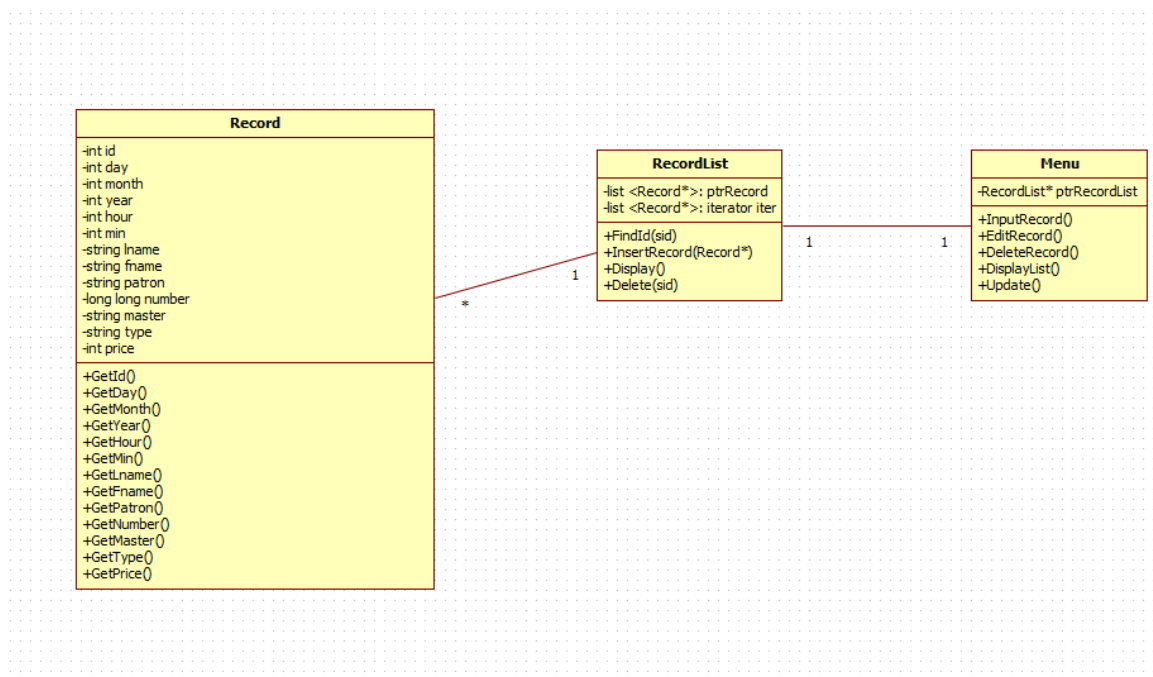


Рис. 5 – Диаграмма классов проекта

## 12. Разработать диаграмму последовательностей UML, на которой отображается, каким образом события разворачиваются во времени.

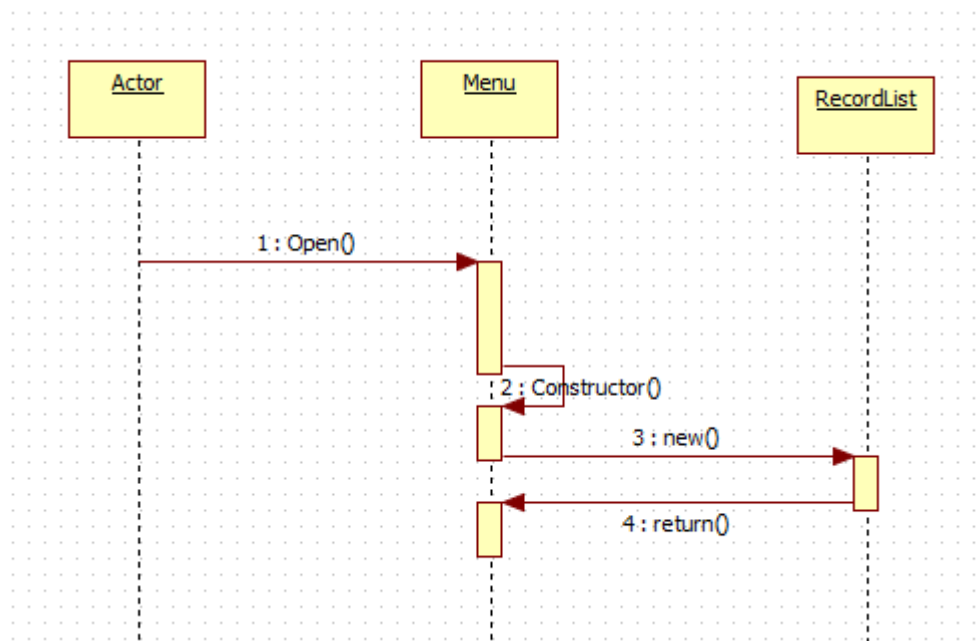


Рис. 6 – Вариант использования: запустить программу

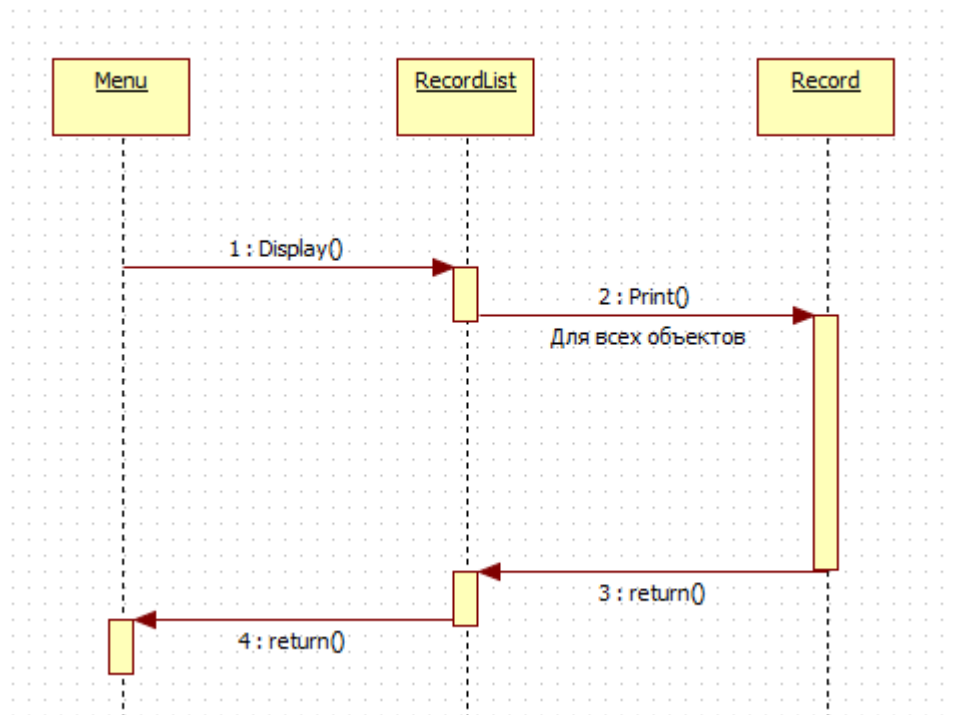


Рис. 7 - Вариант использования: просмотр таблицы

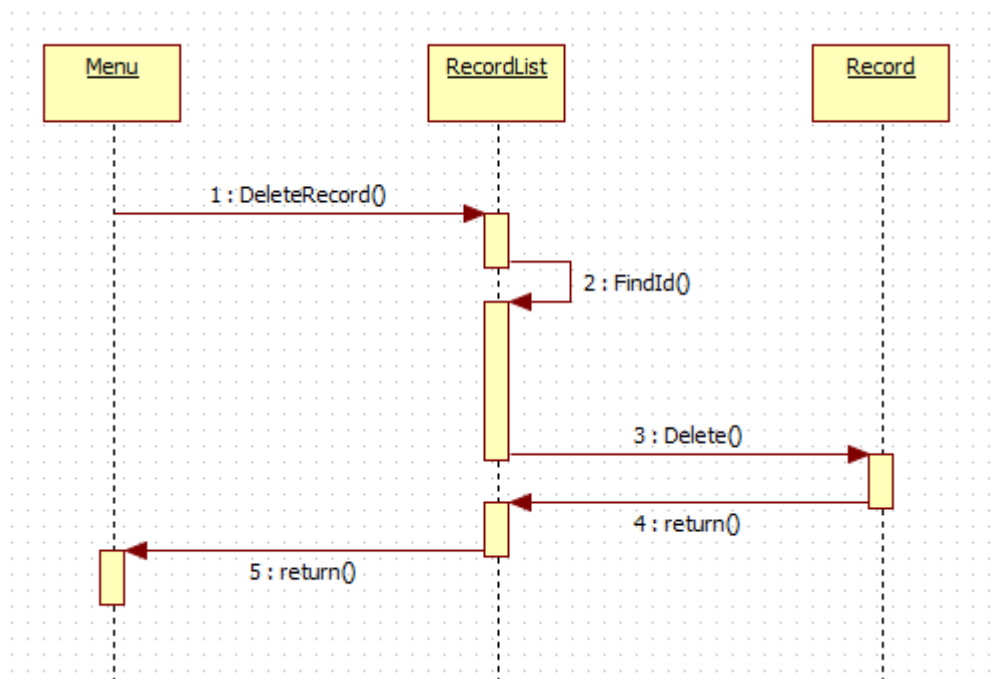


Рис. 8 – Вариант использования: удаление записи

### 13. Написание кода программы. (.H) файлы

#### 1) Заголовочный файл Record.h

```
#ifndef __RECORD_H__
#define __RECORD_H__
#include <iostream>
using namespace std;
class Record{
private:
    int id; //айди/номер
    int day; //день
    int month; //месяц
    int year; //год
    int hour; //час
    int min; //минута
    string lname; //фамилия
    string fname; //имя
    string patron; //отчество
    long long number; //номер телефона
    string master; //имя мастера
    string type; //вид услуги
    int price; //цена
public:
    Record(int sid, int sday, int smonth, int syear, int shour, int smin,
//конструктор
        string slname, string sfname, string spatron, long long snumber, string
smaster,
        string stype, int sprice);
    ~Record(); //деструктор
    //Геттеры
    int GetId();
    int GetDay();
    int GetMonth();
    int GetYear();
    int GetHour();
    int GetMin();
    string GetLname();
    string GetFname();
    string GetPatron();
    long long GetNumber();
    string GetMaster();
    string GetType();
```

```

int GetPrice();
//Сеттеры
void SetId(int sid);
void SetDay(int sday);
void SetMonth(int smonth);
void SetYear(int syear);
void SetHour(int shour);
void SetMin(int smin);
void SetLname(string slname);
void SetFname(string sfname);
void SetPatron(string spatron);
void SetNumber(long long snumber);
void SetMaster(string smaster);
void SetType(string stype);
void SetPrice(int sprice);
void Print();//выводит поля экземпляра класса
};
#endif

```

## 2) Заголовочный файл Record.h

```

#ifndef __RECORDLIST_H__
#define __RECORDLIST_H__
#include <list>
#include "Record.h"
using namespace std;
class RecordList{
private:
    list<Record*> ptrRecord;//контейнер с указателями на экземпляры
    класса запись
    list<Record*>::iterator iter;    //итератор
public:
    ~RecordList();    //деструктор
    Record* FindId(int sid); //метод нахождения элемента в списке по id
    void InsertRecord(Record*);    //Введение нового элемента в список
    void Display();    //вывод списка
    void Delete(int sid);    //удаление элемента с id = аргументу
};

#endif

```

## 3) Заголовочный файл GlobalFunc.h

```

#ifndef __GLOBALFUNC_H__
#define __GLOBALFUNC_H__
void printHeader();      //вывод заголовка таблицы
void printFooter(); //вывод линий(костей) таблицы
int checkNumber(long long snumber); //проверка диапазона номера телефона
int checkDay(int sday); //проверка диапазона ввода дня
int checkMonth(int smonth); //проверка диапазона ввода месяца
int checkYear(int syear); //проверка диапазона ввода года
int checkHour(int shour); //проверка диапазона ввода часов
int checkMin(int smin); //проверка диапазона ввода минут
int checkNegative(int num); //проверка на отрицательное число
void enterInt(int* i); //функция ввода целого числа
void enterLongLong(long long* l); //функция ввода long long числа, для номера
телефона
#endif

```

#### 4) Заголовочный файл Menu.h

```

#ifndef __MENU_H__
#define __MENU_H__

#include "RecordList.h"

using namespace std;
class Menu{
private:
    RecordList* ptrRecordList; // указатель на список записей
public:
    Menu(); //конструктор
    ~Menu(); //деструктор
    void InputRecord(); //ввод записи в список
    void EditRecord(); //редактирование записи
    void DeleteRecord(); //удаление записи
    void DisplayList(); //вывод списка
    void Update(); //обновление меню
};

#endif

```

#### 14. Написание исходных файлов. (.cpp) файлы

##### 1) Исходный файл GlobalFunc.cpp

```
#include <iostream>
#include <iomanip>
#include "GlobalFunc.h"

using namespace std;

void printHeader(){
    cout << endl
        << setw(3) << "id" << "|"
        << setw(10) << "Дата" << "|"
        << setw(6) << "Время" << "|"
        << setw(12) << "Фамилия" << " "
        << setw(12) << "Имя" << " "
        << setw(12) << "Отчество" << "|"
        << setw(12) << "Телефон" << "|"
        << setw(12) << "Мастер" << "|"
        << setw(15) << "Вид услуги" << "|"
        << setw(4) << "Цена" << endl;
}

void printFooter(){
    cout << setw(3) << "---" << "+"
        << setw(10) << "-----" << "+"
        << setw(6) << "-----" << "+"
        << setw(12) << "-----"
        << setw(12) << "-----"
        << setw(12) << "-----" << "+"
        << setw(12) << "-----" << "+"
        << setw(12) << "-----" << "+"
        << setw(15) << "-----" << "+"
        << setw(4) << "----";
}

int checkNegative(int num){//проверка на отрицательное число
    if (num < 0){
        cout << "Введите неотрицательное число" << endl;
        return 0;
    }
    return 1;
}
```



```

}

int checkNumber(long long snumber){//проверка диапазона ввода номера
    if (snumber < 0){
        cout << "Введите неотрицательное число" << endl;
        return 0;
    }
    if (snumber < 800000000000 || snumber > 9000000000000){
        cout << "Ошибка ввода номера телефона" << endl;
        return 0;
    }
    return 1;
}

int checkDay(int sday){//проверка диапазона ввода дня
    if (checkNegative(sday)){
        if (sday > 31 || sday == 0){
            cout << "Ошибка ввода дня" << endl;
            return 0;
        }
    }
    return 1;
}

int checkMonth(int smonth){//проверка диапазона ввода месяца
    if (checkNegative(smonth)){
        if (smmonth > 12 || smmonth == 0){
            cout << "Ошибка ввода месяца" << endl;
            return 0;
        }
    }
    return 1;
}

int checkYear(int syear){//проверка диапазона ввода года
    if (syearch < 2020 || syearch > 2030){
        cout << "Ошибка ввода года" << endl;
        return 0;
    }
    return 1;
}

int checkHour(int shour){//проверка диапазона ввода часов

```

```

        if (checkNegative(shour)){
            if (shour > 23){
                cout << "Ошибка ввода часов" << endl;
                return 0;
            }
        }
        return 1;
    }

int checkMin(int smin){//проверка диапазона ввода минут
    if (checkNegative(smin)){
        if (smin > 60){
            cout << "Ошибка ввода минут" << endl;
            return 0;
        }
    }
    return 1;
}

void enterInt(int *i){ //ввод целого числа
    while (!(cin >> *i)){ //пока ввод некорректен
        cin.clear(); //очищаем флаги ошибок
        cin.ignore(std::numeric_limits<std::streamsize>::max(),
'n');//пропускаем все символы
        cout << "Неверный ввод." << endl;
    }
    cin.ignore(std::numeric_limits<std::streamsize>::max(), 'n'); //еще раз
пропускаем все символы
    //на случай, если остались \0 \n и др.
}

void enterLongLong(long long *l){
    while (!(cin >> *l)){//пока ввод некорректен
        cin.clear();//очищаем флаги ошибок
        cin.ignore(std::numeric_limits<std::streamsize>::max(),
'n');//пропускаем все символы
        cout << "Неверный ввод." << endl;
    }
    cin.ignore(std::numeric_limits<std::streamsize>::max(), 'n'); //еще раз
пропускаем все символы
}

```

## 2) Исходный файл Record.cpp

```
#include <iostream>
#include "Record.h"
#include <string>
#include <iomanip>

Record::Record(int sid, int sday, int smonth, int syear, int shour, int
smin, //конструктор
               string slname, string sfname, string spatron, long long snumber, string smaster,
               string stype, int sprice) :id(sid), day(sday), month(smonth), year(syear),
               hour(shour), min(smin), lname(slname), fname(sfname), patron(spatron),
               number(snumber), master(smaster), type(stype), price(sprice){
}
Record::~Record() { //деструктор

}
//Геттеры
int Record::GetId(){
    return id;
}
int Record::GetDay(){
    return day;
}
int Record::GetMonth(){
    return month;
}
int Record::GetYear(){
    return year;
}
int Record::GetHour(){
    return hour;
}
int Record::GetMin(){
    return min;
}
string Record::GetLname(){
    return lname;
}
string Record::GetFname(){
    return fname;
}
string Record::GetPatron(){
```

```

        return patron;
    }
    long long Record::GetNumber(){
        return number;
    }
    string Record::GetMaster(){
        return master;
    }
    string Record::GetType(){
        return type;
    }
    int Record::GetPrice(){
        return price;
    }

    //Сеттеры
    void Record::SetId(int sid){
        id = sid;
        return;
    }
    void Record::SetDay(int sday){
        day = sday;
        return;
    }
    void Record::SetMonth(int smonth){
        month = smonth;
        return;
    }
    void Record::SetYear(int syear){
        year = syear;
        return;
    }
    void Record::SetHour(int shour){
        hour = shour;
        return;
    }
    void Record::SetMin(int smin){
        min = smin;
        return;
    }
    void Record::SetLname(string slname){
        lname = slname;
        return;
    }

```

```

}
void Record::SetFname(string sfname){
    fname = sfname;
    return;
}
void Record::SetPatron(string spatron){
    patron = spatron;
    return;
}
void Record::SetNumber(long long snumber){
    number = snumber;
    return;
}
void Record::SetMaster(string smaster){
    master = smaster;
    return;
}
void Record::SetType(string stype){
    type = stype;
    return;
}
void Record::SetPrice(int sprice){
    price = sprice;
    return;
}
void Record::Print(){    //метод вывода полей класса запись
    cout << endl
        << setw(3) << GetId() << "|"
        << setw(2) << GetDay() << "."
        << setw(2) << GetMonth() << "."
        << setw(4) << GetYear() << "|"
        << setw(3) << GetHour() << ":"
        << setw(2) << GetMin() << "|"
        << setw(12) << GetLname() << " "
        << setw(12) << GetFname() << " "
        << setw(12) << GetPatron() << "|"
        << setw(12) << GetNumber() << "|"
        << setw(12) << GetMaster() << "|"
        << setw(15) << GetType() << "|"
        << setw(4) << GetPrice() << endl;
}

```

### 3) Исходный файл RecordList.cpp

```
#include <iostream>
#include <string>
#include <iomanip>
#include "RecordList.h"
#include "GlobalFunc.h"

using namespace std;

RecordList::~RecordList() { //удаление списка записей
    while (!ptrRecord.empty()) { //пока не пустой
        iter = ptrRecord.begin(); //итератор на начало
        delete *iter; //очищаем выделенную память
        ptrRecord.erase(iter); //удаляем элемент
    }
}

Record* RecordList::FindId(int sid) {
    if (ptrRecord.empty()) { //если пустой
        return NULL; //возвращаем пустой указатель
    }
    else { //иначе
        iter = ptrRecord.begin(); //ставим итератор в начало списка
        while (iter != ptrRecord.end()) { //пока не конец
            if ((*iter)->GetId() == sid) //сравниваем аргумент с id
                return *iter; //если нашли =, то возвращаем указатель на элемент
            *iter++; //если не нашли, то переходим к след. элементу
        }
        return NULL; //если ничего не нашли, возвращаем пустой указатель
    }
}

void RecordList::InsertRecord(Record* ptrR) {
    ptrRecord.push_back(ptrR); //вставляем в конец новую запись
}

void RecordList::Display() {
    if (ptrRecord.empty()) //если список пустой
        cout << "\n\nСписок записей пуст, добавьте записи." << endl;
    //пишем, что пустой
}
```

```

else //иначе
{
    printHeader();    //выводим заголовок таблицы
    printFooter();
    iter = ptrRecord.begin(); //итератор в начало списка
    while (iter != ptrRecord.end()){ //если итератор не в конце
        (*iter)->Print();    //то выводим элемент по указателю при
помощи метода
        printFooter();
        *iter++;    //к следующему элементу
    }
}
return;
}

void RecordList::Delete(int sid){    //удаляем запись с соотв. id
    if (*iter = FindId(sid)){ //если метод нашел запись с таким id, то
        delete *iter; //очищаем выделенную память
        ptrRecord.erase(iter); //удаляем указатель
        cout << "\n\nЗапись удалена." << endl;
    }
    else
        cout << "Запись не найдена" << endl;
}

```

#### 4) Исходный файл Menu.cpp

```

#include <iostream>
#include <string>
#include <iomanip>
#include <conio.h>
#include "GlobalFunc.h"
#include "Menu.h"

Menu::Menu(){
    ptrRecordList = new RecordList; //выделяем память под список
}

Menu::~Menu(){
    delete ptrRecordList;    //очищаем зарезервированную память
}

```

```

}
void Menu::DisplayList() {    //выводим список
    ptrRecordList->Display();    //метод класса списка записей
}

void Menu::DeleteRecord() {    //Удаление записи из списка
    int sid;
    do {
        cout << "\n\nВведите id записи для удаления:";
        enterInt(&sid);    //вводим id
    } while (!checkNegative(sid)); //пока отрицательное
    ptrRecordList->Delete(sid);    //вызываем метод удаления класса списка
записей
}

void Menu::Update() {    //обновление меню
    while (1) {    //бесконечный, пока не нажат 0
        system("cls");
        cout << "1. Просмотр списка записей\n"
            << "2. Добавить новую запись\n"
            << "3. Редактировать существующую запись\n"
            << "4. Удалить запись\n"
            << "0. Выйти из программы" << endl;
        switch (_getch()) { //ждем нажатия клавиши
            case 49: {    //1
                system("cls");
                DisplayList();
                break;
            }
            case 50: {    //2
                system("cls");
                InputRecord();
                break;
            }
            case 51: {    //3
                system("cls");
                EditRecord();
                break;
            }
            case 52: {    //4
                system("cls");
                DeleteRecord();
                break;
            }
        }
    }
}

```



```

    }
    case 48:{ //0
        return;
        break;
    }
    default: break;
}
cout << "\nНажмите любую клавишу для продолжения...";
_getch();//читаем символ, что бы не переходило к следующей
итерации
}
}

void Menu::InputRecord(){ //ввод записи
    int sid; //ид
    int sday; //день
    int smonth; //месяц
    int syear; //год
    int shour; //час
    int smin; //минуты
    string slname; //фамилия
    string sfname; //имя
    string spatron; //отчество
    long long snumber; //номер
    string smaster; //имя мастера
    string stype; //тип
    int sprice; //стоимость
    do{
        cout << "\n\nВведите id записи:";
        enterInt(&sid);//функция ввода целого числа
        if (sid > 900)
            cout << "Слишком большое id" << endl;
        if (ptrRecordList->FindId(sid) != NULL)
            cout << "Запись с таким id уже существует, повторите ввод."
<< endl;
    } while ((!checkNegative(sid)) || (sid > 900) || (ptrRecordList->FindId(sid) !=
NULL)); //условие выполнения, не отриц,
//не больше 900, нет такого ид
    do{
        cout << "Введите день:";
        enterInt(&sday);
    } while (!checkDay(sday));
    ///////////////////////////////////

```

```

do{
    cout << "Введите месяц:";
    enterInt(&smoth);
} while (!checkMonth(smoth));
//////////

cout << "Введите год:";
do{
    enterInt(&syar);
} while (!checkYear(syar));
//////////

do{
    cout << "Введите часы:";
    enterInt(&shour);
} while (!checkHour(shour));
//////////

do{
    cout << "Введите минуты:";
    enterInt(&smin);
} while (!checkMin(smin));
//////////

do{
    cout << "Введите Имя:";
    sfname.clear();
    getline(cin, sfname);
    if (sfname.size() > 12)
        cout << "Слишком длинная строка" << endl;
} while (sfname.size() > 12);
//////////

do{
    cout << "Введите Фамилию:";
    slname.clear();
    getline(cin, slname);
    if (slname.size() > 12)
        cout << "Слишком длинная строка" << endl;
} while (slname.size() > 12);
//////////

do{
    cout << "Введите Отчество:";
    spatron.clear();
    getline(cin, spatron);
    if (spatron.size() > 12)
        cout << "Слишком длинная строка" << endl;
} while (spatron.size() > 12);

```

```

    } while (spatron.size() > 12);
    //////////////////////////////////
    do{
        cout << "Введите номер телефона(В формате 890000000000):";
        enterLongLong(&snumber);
    } while (!checkNumber(snumber));
    //////////////////////////////////
    do{
        cout << "Введите имя мастера:";
        smaster.clear();
        getline(cin, smaster);
        if (smaster.size() > 12)
            cout << "Слишком длинная строка" << endl;
    } while (smaster.size() > 12);
    //////////////////////////////////
    do{
        cout << "Введите вид услуги:";
        stype.clear();
        getline(cin, stype);
        if (stype.size() > 15)
            cout << "Слишком длинная строка" << endl;
    } while (stype.size() > 15);
    //////////////////////////////////
    do{
        cout << "Введите стоимость:";
        enterInt(&sprice);
    } while (!checkNegative(sprice));
    Record *ptrNewRec = new Record(sid, sday, smonth, syear, shour, smin,
    sfname, slname, spatron, snumber, smaster, stype, sprice);
    ptrRecordList->InsertRecord(ptrNewRec);
}

```

```

void Menu::EditRecord(){
    int sid;
    int sday;
    int smonth;
    int syear;
    int shour;
    int smin;
    string slname;
    string sfname;
    string spatron;
    long long snumber;

```

```

string smaster;
string stype;
int sprice;
Record* eRec;
do{
    cout << "\n\nВведите id записи для редактирования:";
    enterInt(&sid);
} while (!checkNegative(sid));
if (eRec = ptrRecordList->FindId(sid)){
    printHeader();
    printFooter();
    eRec->Print();
    printFooter();
    cout << endl;
    do{
        cout << "Введите день:";
        enterInt(&sday);
    } while (!checkDay(sday));
    eRec->SetDay(sday);
    //////////////////////////////////
    do{
        cout << "Введите месяц:";
        enterInt(&smonth);
    } while (!checkMonth(smonth));
    eRec->SetMonth(smonth);
    //////////////////////////////////
    do{
        cout << "Введите год(В формате 0000):";
        enterInt(&syyear);
    } while (!checkYear(syyear));
    eRec->SetYear(syyear);
    //////////////////////////////////
    do{
        cout << "Введите часы:";
        enterInt(&shour);
    } while (!checkHour(shour));
    eRec->SetHour(shour);
    //////////////////////////////////
    do{
        cout << "Введите минуты:";
        enterInt(&smin);
    } while (!checkMin(smin));
    eRec->SetMin(smin);
}

```

```

//////////
do{
    cout << "Введите Имя:";
    sfname.clear();
    getline(cin, sfname);
    if (sfname.size() > 12)
        cout << "Слишком длинная строка" << endl;
} while (sfname.size() > 12);
eRec->SetFname(sfname);
//////////
do{
    cout << "Введите Фамилию:";
    slname.clear();
    getline(cin, slname);
    if (slname.size() > 12)
        cout << "Слишком длинная строка" << endl;
} while (slname.size() > 12);
eRec->SetLname(slname);
//////////
do{
    cout << "Введите Отчество:";
    spatron.clear();
    getline(cin, spatron);
    if (spatron.size() > 12)
        cout << "Слишком длинная строка" << endl;
} while (spatron.size() > 12);
eRec->SetPatron(spatron);
//////////
do{
    cout << "Введите номер телефона(В формате 890000000000):";
    enterLongLong(&snumber);
} while (!checkNumber(snumber));
eRec->SetNumber(snumber);
//////////
do{
    cout << "Введите имя мастера:";
    smaster.clear();
    getline(cin, smaster);
    if (smaster.size() > 12)
        cout << "Слишком длинная строка" << endl;
} while (smaster.size() > 12);
eRec->SetMaster(smaster);
//////////

```

```

do{
    cout << "Введите вид услуги:";
    stype.clear();
    getline(cin, stype);
    if (stype.size() > 15)
        cout << "Слишком длинная строка" << endl;
} while (stype.size() > 15);
eRec->SetType(stype);
//////////
do{
    cout << "Введите стоимость:";
    enterInt(&sprice);
} while (!checkNegative(sprice));
eRec->SetPrice(sprice);
}
else
    cout << "Запись не найдена" << endl;
}

```

##### 5) Исходный файл Source.cpp

```

#include <iostream>
#include <conio.h>
#include "Menu.h"

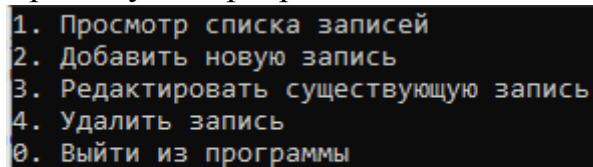
using namespace std;
int main(){
    Menu *menu = new Menu(); //выделяем память под объект класса меню
    setlocale(LC_ALL, "Russian"); // задаём русский текст
    system("chcp 1251"); // настраиваем кодировку консоли
    menu->Update(); //вызываем метод update
    delete menu; //по выходу из метода очищаем память из под объекта
    return 1;
}

```

## 15. Руководство пользователя

Наша программа предназначена для заполнения записи клиентов маникюрной студии. Система запоминает введенные Вами данные, а после выводит их.

При запуске программы Вы попадаете в главное меню.

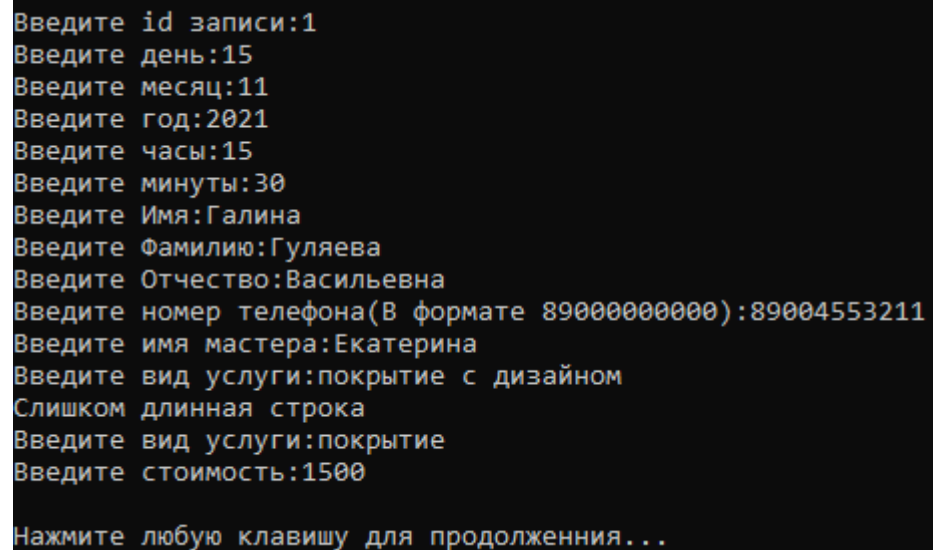


```
1. Просмотр списка записей
2. Добавить новую запись
3. Редактировать существующую запись
4. Удалить запись
0. Выйти из программы
```

Рис.9 – Вид на главное меню программы

Нажатие соответствующих клавиш инициирует определенную операцию.

Для заполнения таблицы необходимо нажать клавишу 2(Добавить новую запись).



```
Введите id записи:1
Введите день:15
Введите месяц:11
Введите год:2021
Введите часы:15
Введите минуты:30
Введите Имя:Галина
Введите Фамилию:Васильевна
Введите номер телефона(В формате 89000000000):89004553211
Введите имя мастера:Екатерина
Введите вид услуги:покрытие с дизайном
Слишком длинная строка
Введите вид услуги:покрытие
Введите стоимость:1500
Нажмите любую клавишу для продолжения...
```

Рис. 10 – Режим заполнения таблицы (при нажатии 1)

```

Введите id записи:-1
Введите неотрицательное число

Введите id записи:9129213
Слишком большое id

Введите id записи:2
Введите день:42
Ошибка ввода дня
Введите день:1
Введите месяц:32
Ошибка ввода месяца
Введите месяц:31
Ошибка ввода месяца
Введите месяц:12
Введите год:2021
Введите часы:51
Ошибка ввода часов
Введите часы:14
Введите минуты:61
Ошибка ввода минут
Введите минуты:60
Введите Имя:Yasdadsadasdsdasddadassddadasdasdasdasdd
Слишком длинная строка
Введите Имя:Никита
Введите Фамилию:Бутырев
Введите Отчество:Сергеевич
Введите номер телефона(В формате 89000000000):213121521512512515
Ошибка ввода номера телефона
Введите номер телефона(В формате 89000000000):-1
Введите неотрицательное число
Введите номер телефона(В формате 89000000000):4505050505050505
Ошибка ввода номера телефона
Введите номер телефона(В формате 89000000000):8000000000
Ошибка ввода номера телефона
Введите номер телефона(В формате 89000000000):89004502020
Введите имя мастера:Лиза
Введите вид услуги:o
Введите стоимость:130_

```

Рис. 11 – Пример проверки на не корректность ввода

Добавляя записи можно их посмотреть нажав клавишу 1 в главном меню.

id	Дата	Время	Фамилия	Имя	Отчество	Телефон	Мастер	Вид услуги	Цена
1	15.11.2021	15:30	Галина	Гуляева	Васильевна	89004553211	Екатерина	покрытие	1500
2	1.12.2021	14:60	Никита	Бутырев	Сергеевич	89004502020	Лиза	o	130

Нажмите любую клавишу для продолжения...

Рис. 12 – Пример вида главного меню при заполненной таблице



При необходимости редактирования записи вы можете нажать на клавишу 3, ввести id существующей записи, которую вам необходимо изменить.

```

Введите id записи для редактирования:1

id|    Дата| Время|    Фамилия    Имя    Отчество|    Телефон|    Мастер|    Вид услуги|Цена
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
1|15.11.2021| 15:30|    Галина    Гуляева    Васильевна| 89004553211|    Екатерина|    покрытие |1500
+-----+-----+-----+-----+-----+-----+-----+-----+
Введите день:13
Введите месяц:12
Введите год(В формате 0000):2021
Введите часы:15
Введите минуты:20
Введите Имя:Галина
Введите Фамилию:Гуляева
Введите Отчество:Васильевна
Введите номер телефона(В формате 89000000000):89004553211
Введите имя мастера:Елизавета
Введите вид услуги:дизайн
Введите стоимость:2000

Нажмите любую клавишу для продолжения...

```

Рис. 13 – Режим редактирования записи.

```

id|    Дата| Время|    Фамилия    Имя    Отчество|    Телефон|    Мастер|    Вид услуги|Цена
+-----+-----+-----+-----+-----+-----+-----+-----+
1|13.12.2021| 15:20|    Гуляева    Галина    Васильевна| 89004553211|    Елизавета|    дизайн |2000
+-----+-----+-----+-----+-----+-----+-----+-----+
2| 1.12.2021| 14:60|    Никита    Бутырев    Сергеевич| 89004502020|    Лиза|    о | 130
+-----+-----+-----+-----+-----+-----+-----+-----+
Нажмите любую клавишу для продолжения...

```

Рис. 14 – Вывод таблицы с измененной записью.

Если вы хотите удалить запись, то необходимо нажать клавишу 4, ввести id записи, если такая запись существует, то она будет удалена.

```

Введите id записи для удаления:
2

Запись удалена.

Нажмите любую клавишу для продолжения...

```

Рис. 15 - Окно удаления записи.

```

id|    Дата| Время|    Фамилия    Имя    Отчество|    Телефон|    Мастер|    Вид услуги|Цена
+-----+-----+-----+-----+-----+-----+-----+-----+
1|13.12.2021| 15:20|    Гуляева    Галина    Васильевна| 89004553211|    Елизавета|    дизайн |2000
+-----+-----+-----+-----+-----+-----+-----+-----+
Нажмите любую клавишу для продолжения...

```

Рис. 16 - Вывод таблицы с удаленной записью.

## История проекта на GitHub

Проект находится по ссылке: <https://github.com/morkein/comproject>

```
Author: Butyrev Nikita <bggniklol@gmail.com> 2021-11-16 11:23:38
Committer: Butyrev Nikita <bggniklol@gmail.com> 2021-11-16 11:23:38
Tags: v0.0
Child: b1c48a0af5ac370d8257d072997f84ea038bb627 (Merge branch 'develop')
Child: 950197ad04b46a0cdaaba94bbfa824062246d7d8 (0.1 добавила Record.h и Record.cpp, пока пустые)
Branches: develop, master, remotes/origin/develop, remotes/origin/master
Follows:
Precedes: v1.0

Initial project, source.cpp

----- .gitignore -----
new file mode 100644
index 0000000..5b72a85
@@ -0,0 +1,159 @@
+## Ignore Visual Studio temporary files, build results, and
+## files generated by popular Visual Studio add-ons.
+
+## User-specific files
+*.suo
+*.user
+*.sln.docstates
+
+
+## Build results
```

Рис.16 – Бутырев. Первый коммит. Создал проект, добавил .gitignore

```
Diff Old version New version Lines of context: 3 Ignore space change Line diff
Author: Liza Stahievich <liza.12321@mail.ru> 2021-11-16 12:32:52
Committer: Liza Stahievich <liza.12321@mail.ru> 2021-11-16 12:32:52
Parent: 8cd3c20e604d3ddf9c05b2fe2a003886075ef31d (Initial project, source.cpp)
Child: 33477a2e970a43f0dble78f9a4e2e7f39fad4ff3 (Merge branch 'myfeature1' into develop)
Child: d03fee95e82dc0584cbf5e4ff855d9146bac06d2 (Добавила класс Record, описала конструктор, деструктор, геттеры)
Branches: develop, master, remotes/origin/develop, remotes/origin/master
Follows: v0.0
Precedes: v1.0

0.1 добавила Record.h и Record.cpp, пока пустые

----- comproj/Record.cpp -----
new file mode 100644
index 0000000..e69de29

----- comproj/Record.h -----
new file mode 100644
index 0000000..fa7c7ef
@@ -0,0 +1,5 @@
+#ifndef __RECORD_H__
+#define __RECORD_H__
+#include <iostream>
+using namespace std;
+#endif
\ No newline at end of file
```

Рис.17 – Стахивевич. Второй коммит. Добавление Record.h, Record.cpp

```
Author: Liza Stahievich <liza.12321@mail.ru> 2021-11-16 12:46:41
Committer: Liza Stahievich <liza.12321@mail.ru> 2021-11-16 12:46:41
Parent: 950197ad04b46a0cdaaba94bbfa824062246d7d8 (0.1 добавила Record.h и Record.cpp, пока пустые)
Child: fd06bade276cd898c96516c112674252d1345cc9 (Добавила сеттеры, метод Print)
Branches: develop, master, remotes/origin/develop, remotes/origin/master
Follows: v0.0
Precedes: v1.0

Добавила класс Record, описала конструктор, деструктор, геттеры

----- comproj/Record.cpp -----
index e69de29..c569ee5 100644
@@ -0,0 +1,53 @@
+#include <iostream>
+#include "Record.h"
+#include <string>
+
+Record::Record(int sid, int sday, int smonth, int syear, int shour, int smin, //конструктор
+               string slname, string sfname, string spatron, long long snumber, string smaster,
+               string stype, int sprice) :id(sid), day(sday), month(smonth), year(syear),
+               hour(shour), min(smin), lname(slname), fname(sfname), patron(spatron),
+               number(snumber), master(smaster), type(stype), price(sprice){
+}
+Record::~Record() {} //деструктор
```

Рис.18 – Стахивич. Третий коммит. Описала атрибуты класса Record, конструктор и геттеры.

```
Author: Liza Stahievich <liza.12321@mail.ru> 2021-11-16 12:50:06
Committer: Liza Stahievich <liza.12321@mail.ru> 2021-11-16 12:50:06
Parent: d03fee95e82dc0584cbf5e4ff855d9146bac06d2 (Добавила класс Record, описала конструктор, деструктор, геттеры)
Child: 33477a2e970a43f0db1e78f9a4e2e7f39fad4ff3 (Merge branch 'myfeature1' into develop)
Branches: develop, master, remotes/origin/develop, remotes/origin/master
Follows: v0.0
Precedes: v1.0

Добавила сеттеры, метод Print

----- comproj/Record.cpp -----
index c569ee5..2d79951 100644
@@ -1,6 +1,7 @@
#include <iostream>
#include "Record.h"
#include <string>
+#include <iomanip>

Record::Record(int sid, int sday, int smonth, int syear, int shour, int smin, //конструктор
               string slname, string sfname, string spatron, long long snumber, string smaster,
               string stype, int sprice) :id(sid), day(sday), month(smonth), year(syear),
               hour(shour), min(smin), lname(slname), fname(sfname), patron(spatron),
               number(snumber), master(smaster), type(stype), price(sprice){
+   Record::SetType(stype);
+}
+Record::SetType(string stype) {
+   type = stype;
+}
+Record::Print() {
+   cout << "Record: " << id << " " << day << " " << month << " " << year << " " << hour << " " << min << " " << lname << " " << fname << " " << patron << " " << number << " " << master << " " << type << " " << price << endl;
+}
+Record::~Record() {}
```

Рис.19 – Стахивич. Четвертый коммит. Описала сеттеры и метод Print() класса Record

```
Author: Liza Stahievich <liza.12321@mail.ru> 2021-11-16 12:51:00
Committer: Liza Stahievich <liza.12321@mail.ru> 2021-11-16 12:51:00
Parent: 950197ad04b46a0cdaaba94bbfa824062246d7d8 (0.1 добавила Record.h и Record.cpp, пока пустые)
Parent: fd06bade276cd898c96516c112674252d1345cc9 (Добавила сеттеры, метод Print)
Child: 7898611ab1a71bc65722eebe795ae8719c0f854a (Merge branch 'recordlist' into develop)
Child: a1058271c6da4a75cd819b310f3158edd87bd86b (Описал класс RecordList)
Child: b1c48a0af5ac370d8257d072997f84ea038bb627 (Merge branch 'develop')
Branches: develop, master, remotes/origin/develop, remotes/origin/master
Follows: v0.0
Precedes: v1.0

Merge branch 'myfeature1' into develop
```

Рис.20 – Стахивич. Пятый коммит. Слияние ветки myfeature1 в ветку develop

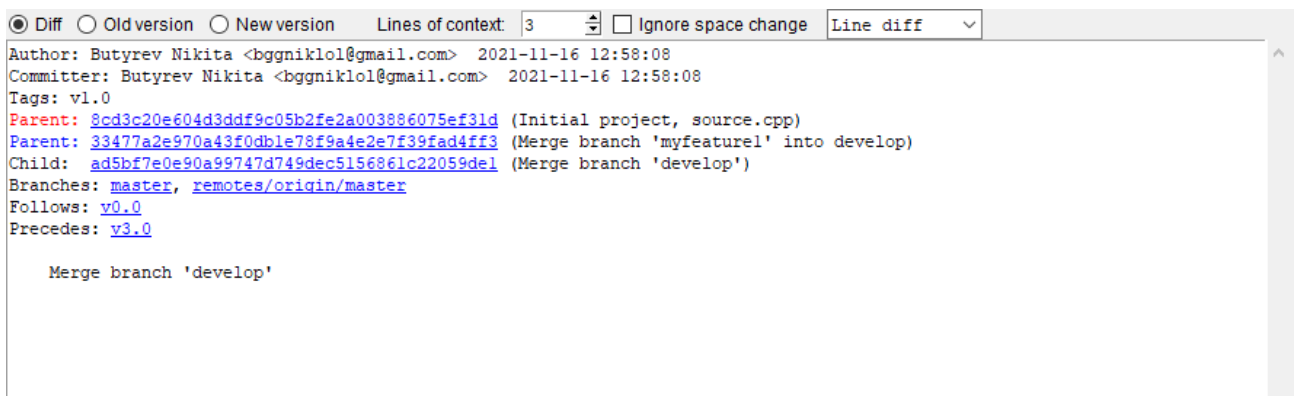


Рис.21 – Бутырев. Шестой коммит. merge ветки develop в master



Рис.22 – Бутырев. Седьмой коммит. Описал класс Record List

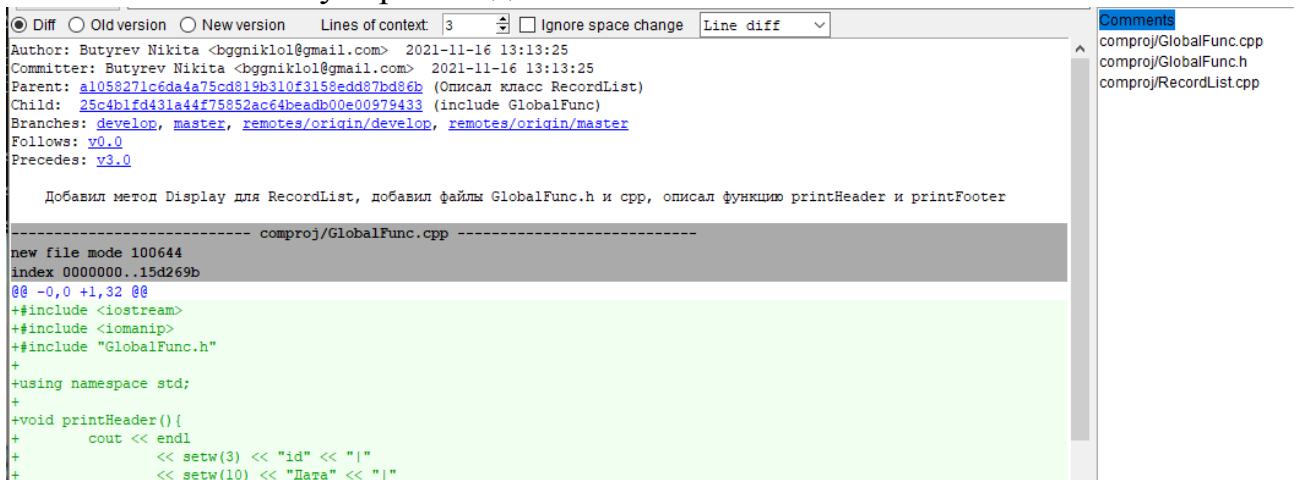


Рис.23 – Бутырев. Восьмой коммит. Добавил метод Display для RecordList, добавил файлы GlobalFunc.h и .cpp, описал функцию printHeader и printFooter

```
Search
Diff Old version New version Lines of context: 3 Ignore space change Line diff
Author: Butyrev Nikita <bognik101@gmail.com> 2021-11-16 13:14:24
Committer: Butyrev Nikita <bognik101@gmail.com> 2021-11-16 13:14:24
Parent: 74b324ae72a8a2262a334c65d36621d1dc7bfa7a (Добавил метод Display для RecordList, добавил файлы GlobalFunc.h и cpp, о
Child: 7898611ab1a71bc65722eebe795ae8719c0f854a (Merge branch 'recordlist' into develop)
Branches: develop, master, remotes/origin/develop, remotes/origin/master
Follows: v0.0
Precedes: v3.0

include GlobalFunc

----- compproj/RecordList.cpp -----
index 3e0ddf2..f5f36a5 100644
@@ -2,6 +2,7 @@
#include <string>
#include <iomanip>
#include <RecordList.h>
+ #include <GlobalFunc.h>

using namespace std;
```

Рис.24 – Бутырев. Девятый коммит. Забыл добавить #include  
Десятый и одиннадцатый коммиты - слияние с ветками develop и master.

```
Diff Old version New version Lines of context: 3 Ignore space change Line diff
Author: Lisa Stahievich <lisa.12321@mail.ru> 2021-11-16 13:26:55
Committer: Lisa Stahievich <lisa.12321@mail.ru> 2021-11-16 13:26:55
Parent: 7898611ab1a71bc65722eebe795ae8719c0f854a (Merge branch 'recordlist' into develop)
Child: 8f15b514f42da1751b9936663a804a793259bb96 (Merge branch 'menu' into develop)
Child: 7da41600da20ce440635a365c7d1827f1af2ac72 (Добавлены файлы класса menu)
Branches: develop, master, remotes/origin/develop, remotes/origin/master
Follows: v0.0
Precedes: v3.0

Добавлены функции ввода значений в переменные и проверки диапазона значений

----- compproj/GlobalFunc.cpp -----
index 15d269b..41a9498 100644
@@ -29,4 +29,91 @@ void printFooter(){
    << setw(12) << "-----" << "+"
    << setw(15) << "-----" << "+"
    << setw(4) << "----";
-}
+ \ No newline at end of file
+
+int checkNegative(int num){//проверка на отрицательное число
+    if (num < 0){
+        cout << "Введите неотрицательное число" << endl;
+        return 0;
+    }
+    return 1;
+}
```

Рис.25 – Стахивич. Двенадцатый коммит. Добавление описание глобальных  
функций.

Diff Old version New version Lines of context: 3 Ignore space change Line diff

Author: Butyrev Nikita <bgniklol@gmail.com> 2021-11-16 13:31:50  
Committer: Butyrev Nikita <bgniklol@gmail.com> 2021-11-16 13:31:50  
Parent: [a5ed718cc7fe8200864a84310aa092de93bd55ff](#) (Добавлены функции ввода значений в переменные и проверки диапазона значений)  
Child: [48202ee3451411a41f2ae004f5893537cec299aa](#) (добавлены конструктор, деструктор, методы удаления записи, обновления меню)  
Branches: [develop](#), [master](#), [remotes/origin/develop](#), [remotes/origin/master](#)  
Follows: [v0.0](#)  
Precedes: [v3.0](#)

Добавлены файлы класса menu

----- comproj/Menu.cpp -----  
new file mode 100644  
index 0000000..e69de29

----- comproj/Menu.h -----  
new file mode 100644  
index 0000000..1fda835  
@@ -0,0 +1,20 @@  
+ #ifndef \_\_MENU\_H\_\_  
+ #define \_\_MENU\_H\_\_  
+  
+ #include "RecordList.h"  
+  
+ using namespace std;  
+ class Menu{  
+ private:  
+ RecordList\* ptrRecordList; // указатель на список записей  
+ public:  
+ Menu(); // конструктор  
+ ~Menu(); // деструктор  
+ void InputRecord(); // ввод записи в список  
+ void EditRecord(); // редактирование записи  
+ void DeleteRecord(); // удаление записи  
+ void DisplayList(); // вывод списка  
+ void Update(); // обновление меню  
+ };  
+  
+ #endif  
\\ No newline at end of file

Comments  
comproj/Menu.cpp  
comproj/Menu.h

Рис.26 – Бутырев. Тринадцатый коммит. Добавление файлов menu.cpp, menu.h

Author: Butyrev Nikita <bogniklol@gmail.com> 2021-11-16 13:34:32  
 Committer: Butyrev Nikita <bogniklol@gmail.com> 2021-11-16 13:34:32  
 Parent: [7da41600da20ce440635a365c7d1827f1af2ac72](#) (Добавлены файлы класса menu)  
 Child: [7058f631a5c563d5b1c3ee014c48944bdd017f16](#) (добавлены методы ввода новых данных, редактирования существующих)  
 Branches: [develop](#), [master](#), [remotes/origin/develop](#), [remotes/origin/master](#)  
 Follows: [v0.0](#)  
 Precedes: [v3.0](#)

добавлены конструктор, деструктор, методы удаления записи, обновления меню

```
----- comproj/Menu.cpp -----
index e69de29..3398695 100644
@@ -0,0 +1,66 @@
+#include <iostream>
+#include <string>
+#include <iomanip>
+#include <conio.h>
+#include "GlobalFunc.h"
+#include "Menu.h"
+
+Menu::Menu() {
+    ptrRecordList = new RecordList; //выделяем память под список
+}
+
+Menu::~Menu() {
+    delete ptrRecordList; //очищаем зарезервированную память
+}
+
+void Menu::DisplayList() { //выводим список
+    ptrRecordList->Display(); //метод класса списка записей
+}
+
+void Menu::DeleteRecord() { //Удаление записи из списка
+    int sid;
+    do {
+        cout << "\n\nВведите id записи для удаления:";
+        enterInt(&sid); //вводим id
+    } while (!checkNegative(sid)); //пока отрицательное
+    ptrRecordList->Delete(sid); //вызываем метод удаления класса списка записей
+}
+
+void Menu::Update() { //обновление меню
+    while (1) { //бесконечный, пока не нажат 0
+        svsystem("cls");
```

Рис.27 –Бутырев. Пятнадцатый коммит. Добавление конструктора, деструктора, методов удаления записи, обновления меню, класса Menu

```

Diff Old version New version Lines of context: 3 Ignore space change Line diff
Author: Butyrev Nikita <bggniklol@gmail.com> 2021-11-16 13:37:49
Committer: Butyrev Nikita <bggniklol@gmail.com> 2021-11-16 13:37:49
Parent: 48202ee345141a41f2ae004f5893537cec299aa (добавлены конструктор, деструктор, методы удаления записи, обновления мен
Child: 8f15b514f42da1751b9936663a804a793259bb96 (Merge branch 'menu' into develop)
Branches: develop, master, remotes/origin/develop, remotes/origin/master
Follows: v0.0
Precedes: v3.0

добавлены методы ввода новых данных, редактирования существующих

----- comproj/Menu.cpp -----
index 3398695..aa3f904 100644
@@ -63,4 +63,222 @@ void Menu::Update() { //
    cout << "\nНажмите любую клавишу для продолжения...";
    _getch(); //читаем символ, что бы не переходило к следующей итерации
}

+}
+
+void Menu::InputRecord() { //ввод записи
+    int sid; //ид
+    int sday; //день
+    int smonth; //месяц
+    int syear; //год
+    int shour; //час
+    int smin; //минуты
+    string sname; //фамилия
+    string sfname; //имя
+    string spatron; //отчество
+    long long snumber; //номер
+    string smaster; //имя мастера
+    string stype; //тип
+    int sprice; //стоимость
+    do{
+        cout << "\nВведите id записи:";
+        enterInt(&sid); //функция ввода целого числа
+        if (sid > 900)
+            cout << "Слишком большое id" << endl;
+        if (ptrRecordList->FindId(sid) != NULL)
+            cout << "Запись с таким id уже существует, повторите ввод." << endl;
+    } while (!(!checkNegative(sid) || (sid > 900) || (ptrRecordList->FindId(sid) != NULL))); //условие выполнения, не о
+    //не больше 900, нет такого id
+    do{
+        cout << "Введите день:";

```

Рис.28 - Бутырев. Шестнадцатый коммит. Добавление методов ввода новых данных, редактирования существующих класса Menu

Семнадцатый и восемнадцатый коммиты - слияние с ветками develop и master.

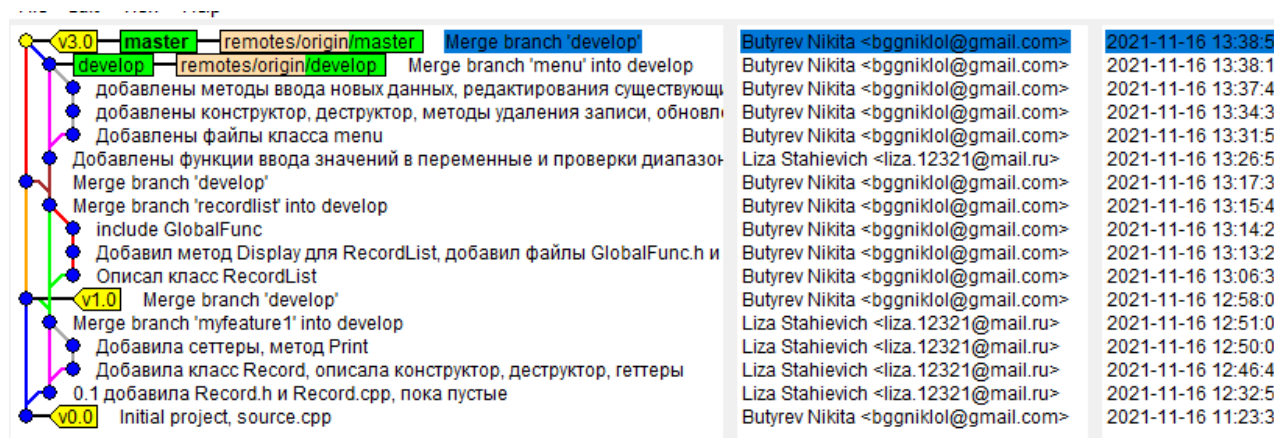


Рис.29 История с gitk



## Заключение

Наш проект был разработан в среде Microsoft Visual Studio 13 на языке C++. Нами была применена система контроля версий Git. В программе не наблюдается никаких зависаний и сбоев. Также в проекте используется система раздельной компиляции. Очистка памяти реализована корректно с помощью ряда деструкторов. Все переменные используются. Все конструкции являются необходимыми. В отчете нами были приложены диаграммы классов, диаграммы вариантов использования и диаграммы действий. Программа соответствует поставленной задаче и выполняет операции корректно.