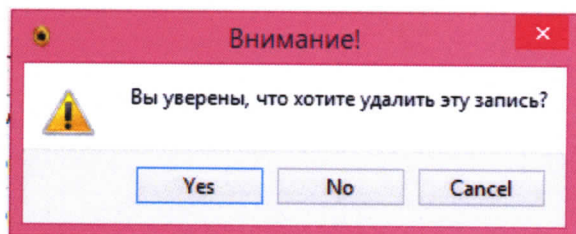


7.2.2. УДАЛЕНИЕ ЗАПИСИ

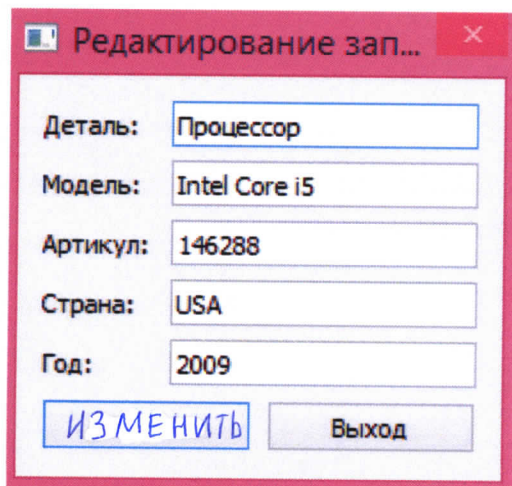
Чтобы удалить запись выделите запись, которую хотите удалить и выберите пункт меню [Запись] -> [Удалить] или воспользуйтесь панелью инструментов, выбрав значок  на панели. Появится сообщение, в котором вы должны подтвердить, что уверены в удалении записи, так как восстановить ее будет невозможно.



Запись будет удалена только при нажатии кнопки [Yes], в ином случае этого не произойдет.

7.2.3. РЕДАКТИРОВАНИЕ ЗАПИСИ

Для редактирования записи выделите запись, которую хотите отредактировать и выберите пункт меню [Запись] -> [Редактировать] либо воспользуйтесь панелью инструментов, выбрав значок . Перед вами появится диалоговое окно, в котором вы можете отредактировать поля выделенной строки. Изменения будут приняты после нажатия кнопки [OK].

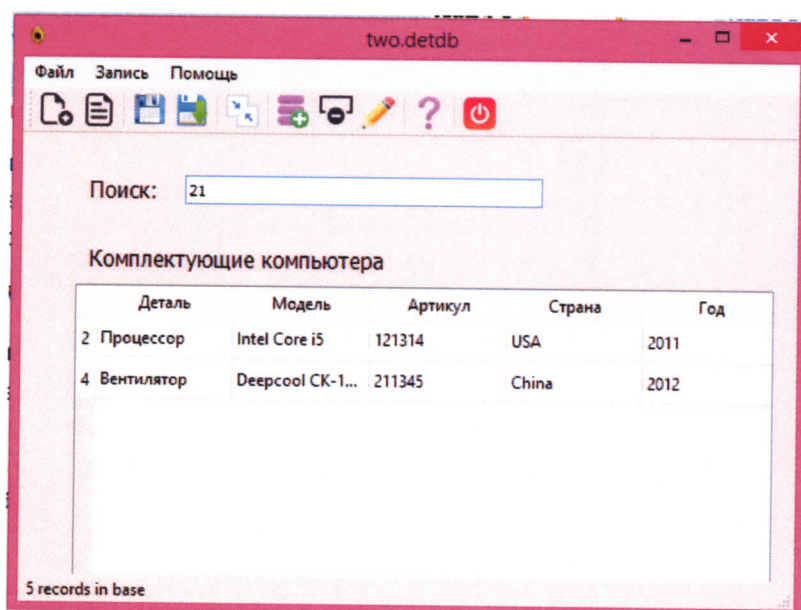


Альтернативный вариант редактирования записи: дважды кликните по ячейке таблицы, которую хотите отредактировать. Ячейка станет доступной для редактирования.

7.3. ДОПОЛНИТЕЛЬНЫЕ ВОЗМОЖНОСТИ

7.3.1. ПОИСК

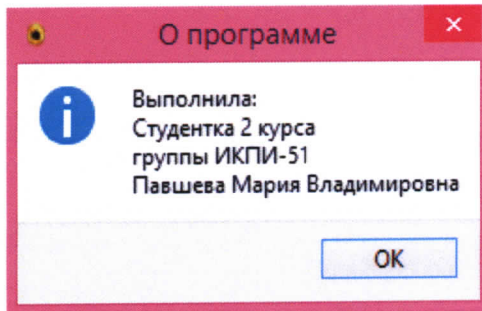
Для поиска по базе данных воспользуйтесь строкой поиска, расположенной сразу над таблицей. Введите в строку слово или последовательность символов, которую хотите найти. Программа немедленно отреагирует на ввод и сразу же отобразит результаты поиска в таблице, в которой располагалась исходная база данных.



Для скрытия результатов поиска просто очистите строку поиска.

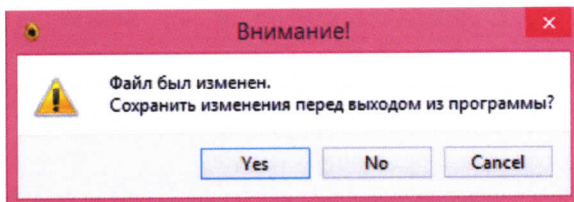
7.3.2. О ПРОГРАММЕ

Для просмотра информации о программе выберите пункт меню [Помощь] > [О программе]. Перед вами появится диалоговое окно:



7.3.3. ВЫХОД ИЗ ПРОГРАММЫ

Чтобы выйти из программы выберите пункт меню [Файл] -> [Выход] или просто нажмите крестик [X] в правом верхнем углу окна программы. Если в открытой базе были изменения, то перед вами появится сообщение, в котором вы должны выбрать, хотите ли вы сохранить базу. Так же вы можете закрыть программу без сохранения либо отменить закрытие редактора.



Если вы согласились на сохранение, то база будет сохранена в файл, из которого она была открыта либо откроется диалоговое окно выбора файла для сохранения, если база еще не сохранялась. То есть произойдет действие, аналогичное описанному в пункте 7.1.3.

8. ЗАКЛЮЧЕНИЕ

Поставленная задача выполнена успешно. Данная программа выполняет основные операции над базой данных. В ходе выполнения курсовой работы была освоена работа с основными возможностями языка C++ и библиотеками Qt, а также с редактором QtCreator.

9. СПИСОК ЛИТЕРАТУРЫ

1. С. В. Козин. Лекции по дисциплине Объектно-ориентированное программирование.
2. М. Шлее - Qt 5.3. Профессиональное программирование на C++. 2015
3. Справочные материалы Qt Creator.

10. ПРИЛОЖЕНИЕ

10.1. Файлы .cpp

main.cpp

```
#include "MainWindow.h"
#include <QApplication>

int main(int argc, char *argv[])
{
    QApplication a(argc, argv);
    MainWindow w;
    w.show();

    return a.exec();
}
```

Details.cpp

```
#include "Details.h"

Detail::Detail(): _comp(""), _model(""), _number(""), _country(""), _year(0)
{}

Detail::Detail(QString comp, QString model, QString number, QString country,
int year):
_comp(comp), _model(model), _number(number), _country(country), _year(year)
{}

```

EditWindow.cpp

```
#include "EditWindow.h"

EditWindow::EditWindow(const QString &comp, const QString &model, const
QString &number, const QString &country, int year, QWidget* parent)
: AddWindow(parent)
{
    setWindowTitle("Редактирование записи");
    compEdit->setText(comp);
    modelEdit->setText(model);
    numberEdit->setText(number);
    countryEdit->setText(country);
    yearEdit->setText(QString::number(year));
}

```

MainWindow.cpp

```
#include "MainWindow.h"
#include "ui_MainWindow.h"
#include "AddWindow.h"
#include "EditWindow.h"
#include <QFileDialog>
#include <QFile>
#include <QMessageBox>
#include <QTableWidgetItem>
#include <iostream>

MainWindow::MainWindow(QWidget *parent) : QMainWindow(parent), ui(new
Ui::MainWindow)
{
    ui->setupUi(this);
    this->setWindowIcon(QPixmap(":/images/Yo.png"));
}

```

```

        dataBase = new DataBase(this);
        connect(dataBase,
SIGNAL(dataChanged(QModelIndex,QModelIndex,QVector<int>)), this,
SLOT(changeWindowTitle()));
        connect(dataBase, SIGNAL(rowCountChanged()), this, SLOT(showRowCount()));
        connect(ui->statusBar, SIGNAL(messageChanged(QString)), this,
SLOT(showRowCount()));
        ui->tableView->setModel(dataBase);
        changeWindowTitle();
        showRowCount();
    }

MainWindow::~MainWindow() {
    delete ui;
}

void MainWindow::on_helpAbout_triggered() {
    QMessageBox::information(this,
        "О программе",
        "Выполнила:\n"
        "Студентка 2 курса\n"
        "группы ИКПИ-51\n"
        "Павшева Мария Владимировна",
        QMessageBox::Ok);
}

void MainWindow::closeEvent(QCloseEvent* event)
{
    if (dataBase->isModified()) {
        int r = showYesNoCancelWarning("Файл был изменен.\n"
            "Сохранить изменения перед "
            "выходом из программы?");

        // нажато Yes
        if (r == QMessageBox::Yes) {
            on_fileSave_triggered();
        } // нажато Cancel
        else if (r == QMessageBox::Cancel) {
            event->ignore();
            return;
        }
    }
    // нажато No или в базе не было изменений
    event->accept();
}

int MainWindow::showYesNoCancelWarning(const QString& message) {
    return QMessageBox::warning(this, tr("Внимание!"),
        message,
        QMessageBox::No,
        QMessageBox::Yes | QMessageBox::Default,
        QMessageBox::Cancel | QMessageBox::Escape);
}

void MainWindow::changeWindowTitle() {
    QString title = QFileInfo(dataBase->filename()).fileName();
    if (dataBase->isModified()) {
        title += "*";
    }
    setWindowTitle(title);
}

void MainWindow::showRowCount() {

```

```

void MainWindow::on_fileOpen_triggered()
{
    // сохранить базу перед открытием?
    if (dataBase->isModified()) {
        int r = showYesNoCancelWarning("Файл был изменен.\n"
                                       "Сохранить изменения перед "
                                       "открытием файла?");

        if (r == QMessageBox::Yes) {
            on_fileSave_triggered();
        } else if (r == QMessageBox::Cancel) {
            return;
        }
    }
    // открытие файла
    QFileDialog dialog(this);
    QStringList filter;
    filter << "Detail DataBase (*.detdb)"
           << "All (*.*)";
    dialog.setNameFilters(filter);
    dialog.setAcceptMode(QFileDialog::AcceptOpen);
    if (dialog.exec() == QDialog::Accepted) {
        dataBase->load(dialog.selectedFiles().at(0));
        // reload table
        ui->tableView->setModel(NULL);
        ui->tableView->setModel(dataBase);
    }
}

void MainWindow::on_fileNew_triggered() {
    if (dataBase->isModified()) {
        int r = showYesNoCancelWarning("Файл был изменен.\n"
                                       "Сохранить изменения перед "
                                       "созданием нового файла?");

        if (r == QMessageBox::Yes) {
            on_fileSave_triggered();
        } else if (r == QMessageBox::Cancel) {
            return;
        }
    }
    delete dataBase;
    dataBase = new DataBase(this);
    connect(dataBase,
            SIGNAL(dataChanged(QModelIndex,QModelIndex,QVector<int>)), this,
            SLOT(changeWindowTitle()));
    connect(dataBase, SIGNAL(rowCountChanged()), this, SLOT(showRowCount()));
    ui->tableView->setModel(dataBase);
    changeWindowTitle();
    showRowCount();
}

void MainWindow::on_fileSaveAs_triggered()
{
    QFileDialog dialog(this);
    QStringList filter;
    filter << "Detail DataBase (*.detdb)"
           << "All (*.*)";
    dialog.setNameFilters(filter);
    dialog.setAcceptMode(QFileDialog::AcceptSave);
    if (dialog.exec() == QDialog::Accepted) {
        dataBase->save(dialog.selectedFiles().at(0));
    }
}

```