|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ **ИУ**

КАФЕДРА **КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)**

**ОТЧЕТ ПО УЧЕБНОЙ ПРАКТИКЕ**

Студент **Валуев Илья Евгеньевич Вариант 4**

*фамилия, имя, отчество*

Группа: ИУ6 24Б

Тип практики: Практика по получению первичных профессиональных умений и навыков (Программирование)

Название предприятия МГТУ им. Н.Э. Баумана

Студент **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Валуев И. Е.**

*подпись, дата фамилия, и.о.*

Руководитель практики **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Минитаева А.М.**

*подпись, дата фамилия, и.о.*

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*2019 г.*

Contents

[**Создание программной системы в интегрированной среде Qt creator.** 3](#_Toc9189842)

[**Условие задания:** 3](#_Toc9189843)

[**Задачи практикума:** 3](#_Toc9189844)

[**Код основного модуля «Меню – записная книжка» :** 4](#_Toc9189845)

[**Код модуля «Ввод записи»:** 4](#_Toc9189846)

[**Код модуля «Построение графика по записям»:** 8](#_Toc9189847)

[**Схемы работы программы** 10](#_Toc9189848)

[Рис 1.1 Диаграмма классов 11](#_Toc9189849)

[Рисунок 1.2 Диаграмма последовательностей 13](#_Toc9189850)

[Рисунок 1.3 Диаграмма объектов программы 14](#_Toc9189851)

[Рисунок 1.4 Окно приложения. Главное меню 15](#_Toc9189852)

[Рисунок 1.5. Создать или добавить информацию о поступлении товара на склад 15](#_Toc9189853)

[Рисунок 1.6 Форма графика поступления товаров 16](#_Toc9189854)

[Рисунок 1.7 Файл записей 16](#_Toc9189855)

[Рисунок 1.8 График поступления товаров по месяцам 17](#_Toc9189856)

[Рисунок 1.9 Диаграмма состояний интерфейса приложения 17](#_Toc9189857)

[**Вывод:** 18](#_Toc9189858)

# **Создание программной системы в интегрированной среде Qt creator.**

## **Условие задания:**

Выполнить объектную декомпозицию, разработать формы интерфейса, диаграмму состояний интерфейса, диаграммы классов интерфейсной и предметной областей, диаграмму последовательностей одной из реализуемых операций. Разработать, протестировать и отладить программу в интегрированной среде кроссплатформенной разработки Qt creator.

Дан файл, содержащий сведения о том, сколько изделий и каких видов содержится на складе и месяц поступления товара на склад. Программа должна в интерактивном режиме формировать исходный файл, а также формировать вспомогательный файл, который содержит информацию об изменении количества изделий на складе. Затем пользователю предоставляется две возможности: обновить все записи первого файла, сведения об изменении которых содержатся во втором файле, или указать конкретные записи, подлежащие обновлению.

Построить график поступления товаров на склад по месяцам.

**Целью практикума** является получение навыков создания небольших программных систем с оконными и консольными интерфейсами.

**Задачи практикума:**

- Овладение методикой и получение практических навыков проектирования небольших программных систем при структурном и объектном подходах;

- более глубокое изучение средств реализации проектов программ на одном из изучаемых универсальных языках программирования высокого уровня;

- воспитание внимания, аккуратности, систематичности, а также формирование интереса к изучаемой профессиональной деятельности.

# **Код основного модуля «Меню – записная книжка» :**

#include "mainwindow.h"

#include "ui\_mainwindow.h"

#include<QProcess>

#include<QDesktopServices>

MainWindow::**MainWindow**()

{

ui->setupUi(this); //работает не трогай

Vnesti=new QPushButton("Input to Database", this); //создал кнопку);

Vnesti -> setGeometry(50, 50, 200, 30); // расположение и длина кнопки

Information=new QPushButton("Information about exchanhing of products", this);

Information -> setGeometry(50, 100, 200, 30); // расположение и длина кнопки

Clear=new QPushButton("Clear Database", this);

Clear -> setGeometry(50, 150, 200, 30); // расположение и длина кнопки

Built = new QPushButton("build a graph", this);

Built -> setGeometry(50, 200, 200, 30); // расположение и длина кнопки

Exit=new QPushButton("Exit", this);

Exit -> setGeometry(50, 250, 200, 30); // расположение и длина кнопки

QObject :: connect(Exit, SIGNAL(clicked()), this, SLOT(close())); //закрытие

QObject::connect( Information, SIGNAL(clicked()), this, SLOT(ShowInfo()));

QObject::connect( Clear, SIGNAL(clicked()),this,SLOT(Clearing()));

}

MainWindow::~***MainWindow***()

{

delete ui;

}

void MainWindow::**ShowInfo**()

{

system("C:\\Users\\dguln\_000\\Documents\\Qt Projects\\gotovaia\\file3.txt");

}

void MainWindow::**Clearing**()

{

file->*open*(QIODevice::WriteOnly);

file->*close*();

}

# **Код модуля «Ввод записи»:**

#include "options.h"

#include <QPushButton>

#include <QString>

#include <QVector>

#include <QDataStream>

#include <QDebug>

#include<stdlib.h>

Options:: **Options**(QPushButton \*Vnesti)

{

QObject :: connect(Vnesti, SIGNAL(clicked()),this, SLOT(show())) ;

nameofproduct=new QLabel("name of product: ", this); //выделение памяти под таблички

nameofproduct -> setGeometry(300, 50, 600, 20); // расположение и длина таблички

amount\_of\_product=new QLabel("amount of product: ", this);

amount\_of\_product -> setGeometry(300, 80, 220, 20);

data\_of\_arrival=new QLabel("date of arrival: ", this);

data\_of\_arrival -> setGeometry(300, 110, 220, 20);

///////////////////////////////////////////////////////

Changezap=new QPushButton("Change string", this); //создал кнопку);

Changezap -> setGeometry(50, 50, 150, 30); // расположение и длина кнопки

ChangeAll=new QPushButton("Change All", this);

ChangeAll -> setGeometry(50, 90, 150, 30); // расположение и длина кнопки

EnterToDatabase=new QPushButton("Enter to Database", this);

EnterToDatabase -> setGeometry(300, 150, 120, 30); // расположение и длина кнопки

// EnterToDatabase -> setEnabled(false);

Exit=new QPushButton("Back", this);

Exit -> setGeometry(50, 150, 150, 30); // расположение и длина кнопки

CreateFile2 = new QPushButton("CreateFile2", this);

CreateFile2 -> setGeometry(430, 150, 120, 30); // расположение и длина кнопки

///////////////////////////////////////////////////////

polename=new QLineEdit(this); //выделение памяти под поле

polename-> setGeometry(430, 50, 100, 20); //расположение поля

polekolvo=new QLineEdit(this);

polekolvo-> setGeometry(430, 80, 100, 20);

poledata=new QLineEdit(this);

poledata-> setGeometry(430, 110, 100, 20);

//////////////////////////////////////////////////////////

file1=new QFile("C:\\Users\\dguln\_000\\Documents\\Qt Projects\\gotovaia\\file1.txt"); //выделяю память под файлы

file2=new QFile("C:\\Users\\dguln\_000\\Documents\\Qt Projects\\gotovaia\\file2.txt");

file3=new QFile("C:\\Users\\dguln\_000\\Documents\\Qt Projects\\gotovaia\\file3.txt");

file4=new QFile("C:\\Users\\dguln\_000\\Documents\\Qt Projects\\gotovaia\\file4.txt");

EnterToDatabase->setShortcut(Qt::Key\_Return); //

QObject :: connect(Exit, SIGNAL(clicked()), this, SLOT(close())); //закрытие

QObject :: connect(EnterToDatabase, SIGNAL(clicked()), this, SLOT(Enter()));

QObject :: connect(ChangeAll, SIGNAL(clicked()), this, SLOT(Change\_All()));

QObject :: connect(CreateFile2, SIGNAL(clicked()), this, SLOT(Change\_File2()));

QObject :: connect(Changezap,SIGNAL(clicked()),this,SLOT(ChangeOne()));

}

void Options:: **Enter**()

{

QVector <QString> Array; //вектор записей

Array << polename -> text() << polekolvo -> text() << poledata -> text();

file1 -> *open*(QIODevice:: Append);

QDataStream stream(file1);

stream << Array;

file1 -> *close*();

polename -> clear();

polekolvo -> clear();

poledata -> clear();

k++;

}

void Options :: **Change\_File2**()

{

QVector<QString> vec={};

QVector<QString> vec2={};

QVector<QString> vec3={};

bool flag=false;

QDataStream stream(file1);

QDataStream stream2(file2);

QDataStream stream3(file3);

file1->*open*(QIODevice::ReadOnly);

file2->*open*(QIODevice::WriteOnly);

while(!file1->*atEnd*())

{

stream>>vec;

stream2<<vec;

}

file1->*close*();

file2->*close*();

file1->*open*(QIODevice::ReadOnly);

file3->*open*(QIODevice::WriteOnly);

file3->*close*();

////////////////////////////////////////////////////////////

while(!file1->*atEnd*())

{

file3->*open*(QIODevice::Append);

stream>>vec;

file2->*open*(QIODevice::ReadOnly);

while(!file2->*atEnd*())

{

stream2>>vec2;

if(vec[0]==vec2[0])

{

if(vec[2].toInt()>vec2[2].toInt())

{

flag=true;

vec3=vec;

}

}

}

if(flag)

{

stream3<<vec3;

flag=false;

}

file2->*close*();

file3->*close*();

}

file1->*close*();

}

void Options::**Change\_All**()

{

QVector<QString> vec={};

QVector<QString> vec2={};

QVector<QString> vec3={};

QString mas[200];

QDataStream stream(file1);

bool flag=false;

QDataStream stream2(file2);

QDataStream stream3(file3);

file3->*open*(QIODevice::ReadOnly);

file1->*open*(QIODevice::WriteOnly);

while(!file3->*atEnd*())

{

stream3 >> vec;

stream << vec;

}

int i=0;

file1->*close*();

file3->*close*();

file1->*open*(QIODevice::ReadOnly);

while(!file1->*atEnd*())

{

stream>>vec;

mas[i]=vec[0];

i++;

}

file1->*close*();

file2->*open*(QIODevice::ReadOnly);

while(!file2->*atEnd*())

{

stream2>>vec;

for(int v=0;v<i;v++)

{

if(vec[0]==mas[v])

{

flag=true;

}

}

if(flag==false)

{

flag=true;

file1->*open*(QIODevice::Append);

stream<<vec;

file1->*close*();

}

flag=false;

}

file1->*close*();

file2->*close*();

}

void Options::**ChangeOne**()

{

QVector<QString> vec={};

QVector<QString> vec2={};

QString a= polename->text();

QDataStream stream(file1);

QDataStream stream3(file3);

QDataStream stream4(file4);

file1->*open*(QIODevice::ReadOnly);

file3->*open*(QIODevice::ReadOnly);

file4->*open*(QIODevice::WriteOnly);

while(!file1->*atEnd*())

{

stream>>vec;

if(vec[0]!=a)

stream4<<vec;

}

file1->*close*();

file4->*close*();

file1->*open*(QIODevice::WriteOnly);

file4->*open*(QIODevice::ReadOnly);

while(!file4->*atEnd*())

{

stream4>>vec;

stream<<vec;

}

file1->*close*();

file4->*close*();

file1->*open*(QIODevice::Append);

while(!file3->*atEnd*())

{

stream3>>vec;

if(vec[0]==a)

vec2=vec;

}

stream<<vec2;

file1->*close*();

file3->*close*();

polename->clear();

}

# **Код модуля «Построение графика по записям»:**

#include "graph.h"

#include<QtCharts/qlineseries.h>

#include<QtCharts/qchart.h>

#include<QtCharts/qchartview.h>

#include<QVector>

#include<QDataStream>

#include<iostream>

#include<QDebug>

#include<QLegend>

#include<QBarSet>

QT\_CHARTS\_USE\_NAMESPACE

graph::**graph**(QPushButton\* activate,QFile \*file)

{

this->file=file;

build = new QPushButton("Build",this);

build->setGeometry(150,50,100,30);

back = new QPushButton("Back",this);

back->setGeometry(150,90,100,30);

label = new QLabel("Enter name of product: ",this);

label->setGeometry(20,10,130,30);

edit = new QLineEdit(this);

edit->setGeometry(150,10,100,30);

// this->setGeometry(400,250,1400,700);

QObject::connect(activate, SIGNAL(clicked()),this,SLOT(show()));

QObject::connect(back, SIGNAL(clicked()),this,SLOT(close()));

QObject::connect(build,SIGNAL(clicked()),this,SLOT(building()));

}

void graph::**building**()

{

window.close();

QVector<QString> vec;

QVector<QString> vecsort;

QVector<QString> masvec[100];

QDataStream stream(file);

int i=0;

QLineSeries \*series = new QLineSeries;

file->*open*(QIODevice::ReadWrite);

while(!file->*atEnd*())

{

stream >> vecsort;

if (vecsort[0] == edit -> text())

{

masvec[i] = vecsort;

i++;

}

}

for (int j=0; j<i - 1; j++) //пузырёк

{

for (int var = 0; var < i - j - 1; ++var)

{

if (masvec[var][2].toInt() > masvec[var+1][2].toInt())

{

swap(*masvec[var][*0*]*, *masvec[var+*1*][*0*]*);

swap(*masvec[var][*1*]*, *masvec[var+*1*][*1*]*);

swap(*masvec[var][*2*]*, *masvec[var+*1*][*2*]*);

}

}

}

for (int j=0; j<i; j++)

{

series->append(masvec[j][2].toInt(), masvec[j][1].toInt());

}

QChart \*chart = new QChart();

chart->addSeries(series);

chart->createDefaultAxes();

//QBarSet \*leg = new QBarSet ("apple");

QChartView \*chartView = new QChartView(chart);

chartView->setRenderHint(QPainter::Antialiasing);

window.setCentralWidget(chartView);

window.setGeometry(500, 300, 800, 600);

window.setWindowTitle("Graphic"); //название окна

chart -> legend()->setVisible(true);

QLabel \*leg = new QLabel(edit -> text(), &window);

// QLabel leg("Apple", &window);

leg->setGeometry(350, 30, 50, 30);

edit -> clear();

window.show();

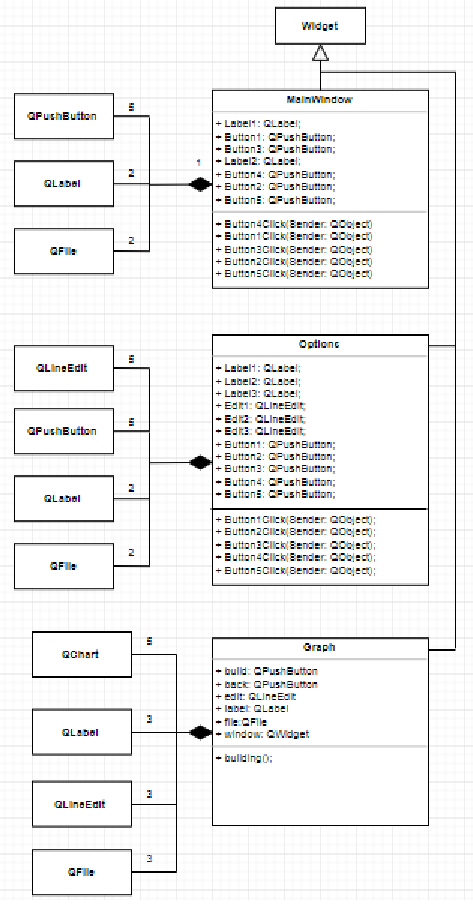
file->*close*();

}

# **Схемы работы программы**

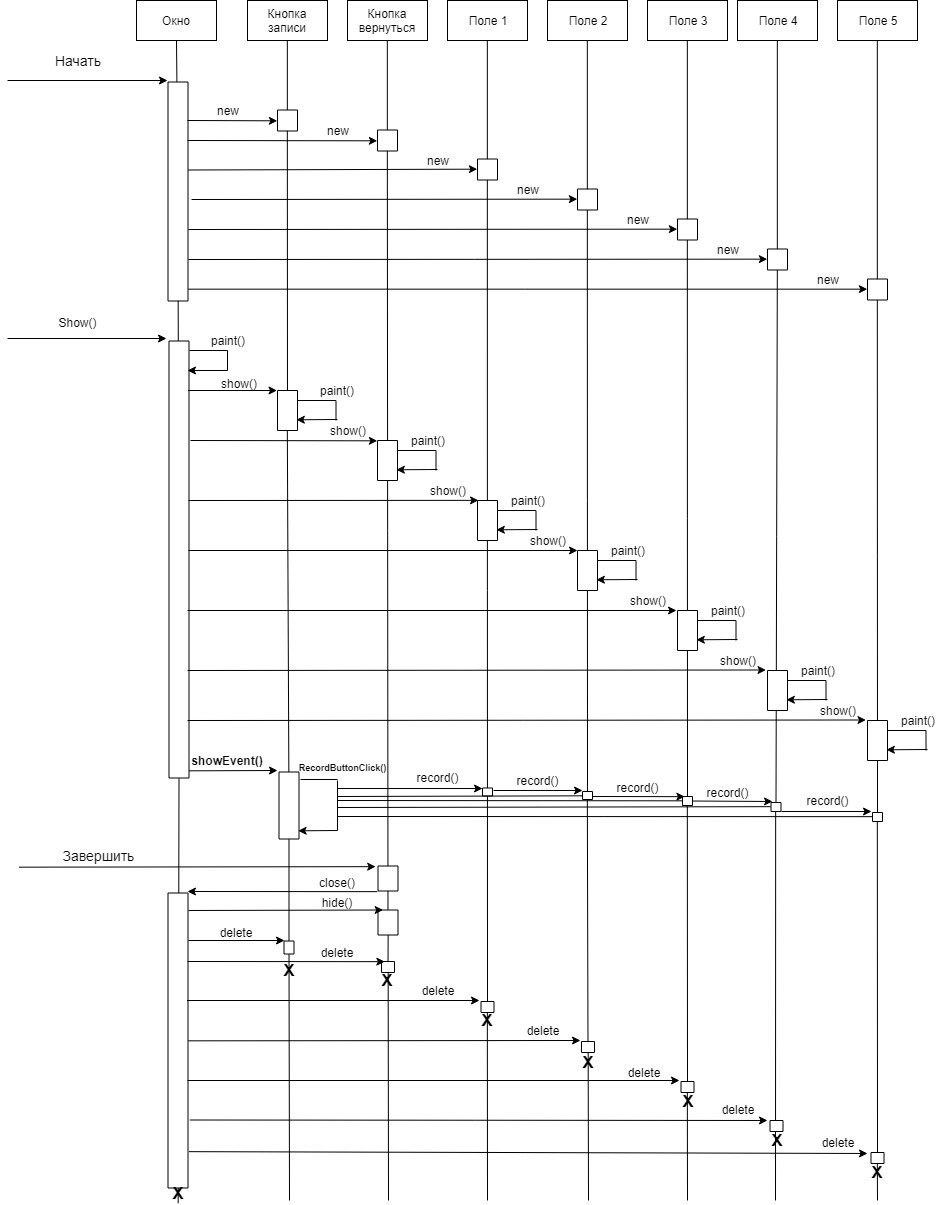
Далее представлены все схемы, объясняющие работу ПО

Диаграмма классов **«Запись товаров склада»** представлена на рисунке 1.1

****

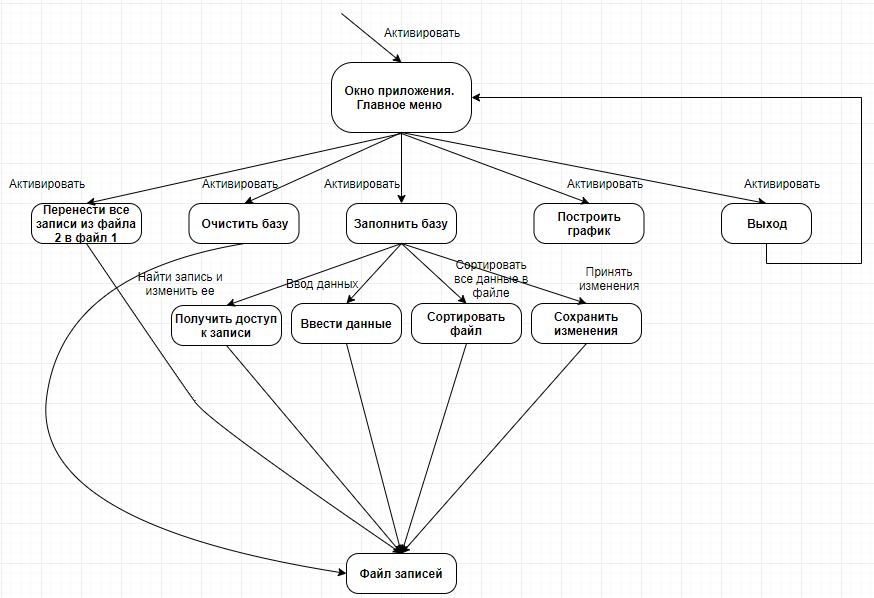
# Рис 1.1 Диаграмма классов

Диаграмма последовательностей программы «**Запись товаров в склад**» (дата поступления{месяц}, тип товара, кол-ва товаров) формы «Ввод записей» представлена на рисунке 1.2



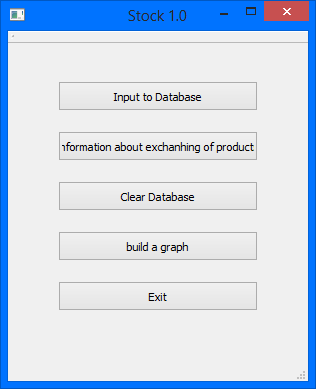
# Рисунок 1.2 Диаграмма последовательностей

Диаграмма объектов программы **«Запись товаров склада»** представлена на рисунке 1.3

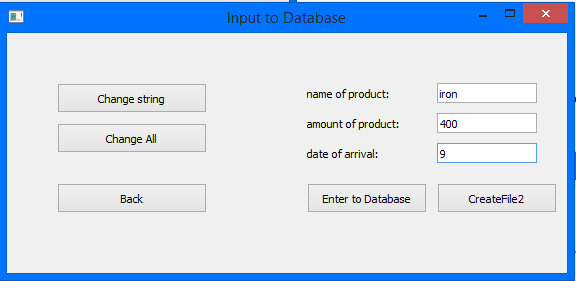


# Рисунок 1.3 Диаграмма объектов программы

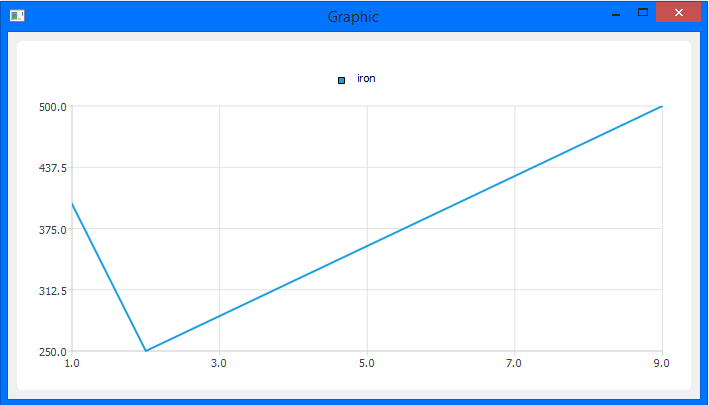
Формы приложения представлены на рисунке 1.4 – 1.6



# Рисунок 1.4 Окно приложения. Главное меню

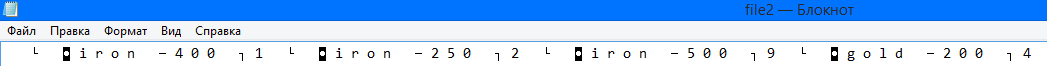


# Рисунок 1.5. Создать или добавить информацию о поступлении товара на склад

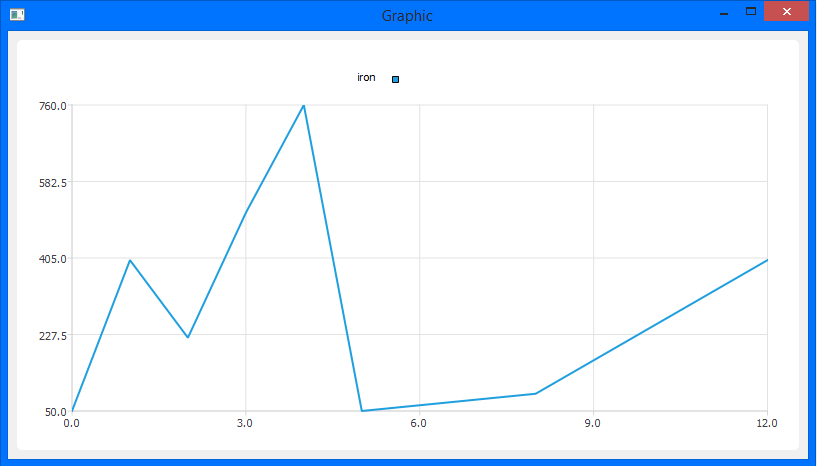


# Рисунок 1.6 Форма графика поступления товаров

Результаты работы программы представлены на рисунках 1.7 и 1.8

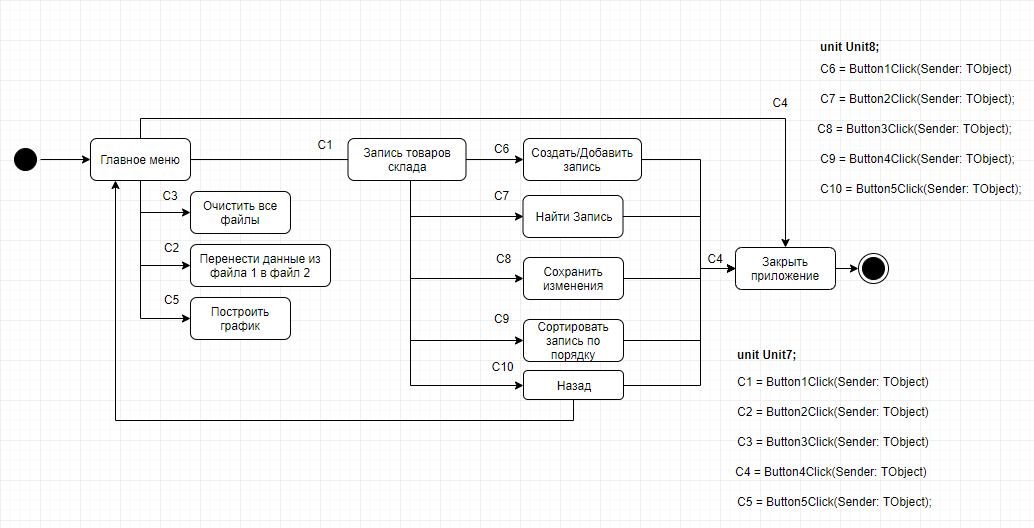


# Рисунок 1.7 Файл записей



# Рисунок 1.8 График поступления товаров по месяцам

Диаграмма состояний интерфейса приложения программы **«Запись товаров склада»** представлена на рисунке 1.9



# Рисунок 1.9 Диаграмма состояний интерфейса приложения

# **Вывод:**

научился разрабатывать небольшие приложения с оконными интерфейсами, укрепил знания многих важных аспектов в программировании, восстановил знания о многих процедурах и функциях, развил навык «интерфейсного программирования».