**Задание 3. Создание программной системы c Qt интерфейсом на С++.**

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 среде Visual Studio или QT Creator.

Файл содержит сведения о элементах: номер элемента по таблице Менделеева, название элемента, атомный вес, проводимость (проводник, изолятор, полупроводник). Программа должна в интерактивном режиме формировать файл, добавлять и удалять данные, а также воспринимать каждый из перечисленных запросов и давать на него ответ.

1. Найти названия и атомные веса всех полупроводников.

2. Вывести сведения о проводниках в порядке возрастания их атомных весов.

3. Определить названия всех изоляторов, атомные веса которых не превышают заданного значения.

4. Построить график зависимости атомного веса элемента от его номера.

**Текст программы:**

(addForm.h)

#ifndef ADDFORM\_H

#define ADDFORM\_H

#include <QWidget>

#include <QLabel>

#include <QLineEdit>

#include <QPushButton>

class addForm : public QWidget

{

Q\_OBJECT

QLabel \*number, \*name, \*weight, \*conduct;

QLineEdit \*numEdit, \*nameEdit,

\*weiEdit, \*condEdit;

QPushButton \*btnAct;

public:

addForm();

public slots:

void addElement();

};

#endif // ADDFORM\_H

(addForm.cpp)

#include "addForm.h"

#include "tableFile.h"

#include "mainForm.h"

#include <QFormLayout>

#include <QString>

addForm::addForm()

{

setWindowTitle("Add element");

QFormLayout \*layout = new QFormLayout;

number = new QLabel("Number", this);

name = new QLabel("Name", this);

weight = new QLabel("Atomic weight", this);

conduct = new QLabel("Conductivity", this);

numEdit = new QLineEdit("", this);

nameEdit = new QLineEdit("", this);

weiEdit = new QLineEdit("", this);

condEdit = new QLineEdit("", this);

btnAct = new QPushButton("Add", this);

layout -> addRow(number, numEdit);

layout -> addRow(name, nameEdit);

layout -> addRow(weight, weiEdit);

layout -> addRow(conduct, condEdit);

layout -> addRow(btnAct);

setLayout(layout);

setFixedSize(300, 180);

connect(btnAct, SIGNAL(clicked()), this, SLOT(addElement()));

}

void addForm::addElement()

{

tableFile table;

elem a;

a.number = numEdit -> text();

a.name = nameEdit -> text();

a.weight = weiEdit -> text();

a.conduct = condEdit -> text();

numEdit -> clear();

nameEdit -> clear();

weiEdit -> clear();

condEdit -> clear();

table.addElem(a);

}

(delForm.h)

#ifndef DELFORM\_H

#define DELFORM\_H

#include <QWidget>

#include <QLabel>

#include <QLineEdit>

#include <QPushButton>

#include "tableFile.h"

class delForm : public QWidget

{

Q\_OBJECT

QLabel \*name;

QLineEdit \*nameEdit;

QPushButton \*btnAct;

elem source[10];

int amount;

public:

delForm();

public slots:

void delElement();

};

#endif // DELFORM\_H

(delForm.cpp)

#include "delForm.h"

#include "mainForm.h"

#include <QFormLayout>

#include <QString>

delForm::delForm()

{

setWindowTitle("Delete element");

QFormLayout \*layout = new QFormLayout;

name = new QLabel("Name", this);

nameEdit = new QLineEdit("", this);

btnAct = new QPushButton("Delete", this);

layout -> addRow(name, nameEdit);

layout -> addRow(btnAct);

setLayout(layout);

setFixedSize(350, 100);

connect(btnAct, SIGNAL(clicked()), this, SLOT(delElement()));

}

void delForm::delElement()

{

if (nameEdit -> text() != "")

{

tableFile table;

int i = 0;

bool flag = false;

while (table.readElem())

{

if (table.buf.name != nameEdit -> text())

{

source[i] = table.buf;

i++;

}

else

flag = true;

}

amount = i;

if (flag)

{

table.delFile();

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

table.addElem(source[j]);

nameEdit -> clear();

}

}

}

(drawForm.h)

#ifndef DRAWFORM\_H

#define DRAWFORM\_H

#include <QWidget>

#include <QPainter>

#include <QPushButton>

#include <QLabel>

#include "tableFile.h"

class drawForm:public QWidget

{

Q\_OBJECT

public:

drawForm();

void getdata();

protected:

elem source[10];

int amount;

QPainter painter;

QPushButton \*btnAct;

QLabel \*zero, \*ord, \*abs;

private slots:

void *paintEvent*(QPaintEvent \*event);

};

#endif // DRAWFORM\_H

(drawForm.cpp)

drawForm::drawForm()

{

setWindowTitle("Graph");

setFixedSize(800, 655);

btnAct = new QPushButton("Draw", this);

btnAct -> setGeometry(5, 5, 80, 30);

//connect(btnAct, SIGNAL(clicked()), this, SLOT(getdata()));

connect(btnAct, SIGNAL(clicked()), this, SLOT(repaint()));

zero = new QLabel("0", this);

zero -> setGeometry(3, 636, 8, 12);

zero -> setStyleSheet("background-color: rgba(255, 255, 255, 0);");

zero -> setStyleSheet("font: 9pt;");

ord = new QLabel("Atomic weight", this);

ord -> setGeometry(25, 45, 90, 18);

ord -> setStyleSheet("background-color: rgba(255, 255, 255, 0);");

ord -> setStyleSheet("font: 9pt;");

abs = new QLabel("Number", this);

abs -> setGeometry(740, 610, 55, 18);

abs -> setStyleSheet("background-color: rgba(255, 255, 255, 0);");

abs -> setStyleSheet("font: 9pt;");

}

void drawForm::*paintEvent*(QPaintEvent \*event)

{

getdata();

Q\_UNUSED(event);

QPainter painter(this);

QPen pen1(Qt::black, 2, Qt::SolidLine);

painter.setPen(pen1);

painter.drawLine(15, 639, 15, 50);

painter.drawLine(15, 50, 20, 55);

painter.drawLine(15, 50, 10, 55);

painter.drawLine(15, 639, 784, 639);

painter.drawLine(784, 639, 779, 634);

painter.drawLine(784, 639, 779, 644);

QPen pen2(Qt::red, 2, Qt::SolidLine);

QPen pen3(Qt::red, 6, Qt::SolidLine);

if (amount > 1)

{

painter.setPen(pen3);

painter.drawPoint(15+6\*source[0].number.toDouble(),

639-2\*source[0].weight.toDouble());

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

{

painter.setPen(pen2);

painter.drawLine(15+6\*source[i].number.toDouble(),

639-2\*source[i].weight.toDouble(),

15+6\*source[i+1].number.toDouble(),

639-2\*source[i+1].weight.toDouble());

painter.setPen(pen3);

painter.drawPoint(15+6\*source[i+1].number.toDouble(),

639-2\*source[i+1].weight.toDouble());

}

}

}

void drawForm::getdata()

{

tableFile \*res = new tableFile;

int i = 0;

while (res -> readElem())

{

source[i] = res -> buf;

i++;

}

amount = i;

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

for (int k = 0; k < amount - 1 - j; k++)

if (source[k].number.toDouble() > source[k+1].number.toDouble())

{

elem \*temp = new elem;

\*temp = source[k];

source[k] = source[k+1];

source[k+1] = \*temp;

delete temp;

}

delete res;

}

(mainForm.h)

#ifndef MAINFORM\_H

#define MAINFORM\_H

//#include <QtGui>

#include <QWidget>

#include <QVBoxLayout>

#include <QHBoxLayout>

#include "addForm.h"

#include "delForm.h"

#include "showForm.h"

#include "drawForm.h"

class Window : public QWidget

{

Q\_OBJECT

QPushButton \*btnAdd, \*btnDel, \*btnShow, \*btnDraw;

addForm winAdd;

delForm winDel;

showForm winShow;

drawForm winDraw;

public:

Window();

public slots:

void showAdd();

void showDel();

void showShow();

void showDraw();

};

#endif // MAINFORM\_H

(mainForm.cpp)

#include "mainForm.h"

Window::Window()

{

setWindowTitle("Mendel table");

btnAdd = new QPushButton("Add", this);

btnDel = new QPushButton("Del", this);

btnShow = new QPushButton("Show", this);

btnDraw = new QPushButton("Draw", this);

QGridLayout \*layout = new QGridLayout;

layout -> addWidget(btnAdd, 0, 0);

layout -> addWidget(btnDel, 0, 1);

layout -> addWidget(btnShow, 1, 0, 1, 2);

layout -> addWidget(btnDraw, 2, 0, 1, 2);

setLayout(layout);

setFixedSize(400, 150);

connect(btnAdd, SIGNAL(clicked()),this,SLOT(showAdd()));

connect(btnShow, SIGNAL(clicked()), this, SLOT(showShow()));

connect(btnDel, SIGNAL(clicked()), this, SLOT(showDel()));

connect(btnDraw, SIGNAL(clicked()), this, SLOT(showDraw()));

}

void Window::showAdd()

{

winAdd.show();

}

void Window::showShow()

{

winShow.makeclear();

winShow.show();

}

void Window::showDel()

{

winDel.show();

}

void Window::showDraw()

{

winDraw.show();

winDraw.getdata();

}

(showForm.h)

#ifndef SHOWFORM\_H

#define SHOWFORM\_H

#include "tableFile.h"

#include <QWidget>

#include <QTableWidget>

#include <QPushButton>

#include <QLabel>

class showForm:public QWidget

{

Q\_OBJECT

QTableWidget \*table;

QPushButton \*btnShowAll, \*btnShowC, \*btnShowS, \*btnShowI;

QLineEdit \*lineI;

public:

showForm();

elem source[10];

int amount;

void makeclear();

void fulfill();

public slots:

void showAll();

void showSemiCond();

void showIns();

void showCond();

};

#endif // SHOWFORM\_H

(showForm.cpp)

#include "showForm.h"

#include "mainForm.h"

showForm::showForm()

{

setWindowTitle("Requested data");

btnShowAll = new QPushButton("Show info about all elements", this);

btnShowC = new QPushButton("Show info about all conductors \nin increasing order of atomic weight", this);

btnShowS = new QPushButton("Show names and numbers of all \nsemiconductors", this);

btnShowI = new QPushButton("Show names of all insulators \nwhose atomic weight does not \nexceed this value", this);

table = new QTableWidget(10, 4, this);

lineI = new QLineEdit("", this);

QStringList strlist;

strlist << "Number" << "Name" << "Atomic weight" << "Conductivity";

table -> setHorizontalHeaderLabels(strlist);

QGridLayout \*layout = new QGridLayout;

layout -> addWidget(btnShowAll, 0, 0);

layout -> addWidget(btnShowC, 1, 0);

layout -> addWidget(btnShowS, 2, 0);

layout -> addWidget(btnShowI, 3, 0);

layout -> addWidget(lineI, 4, 0);

layout -> addWidget(table, 0, 1, 5, 1);

setLayout(layout);

setFixedSize(1096,521);

connect(btnShowAll, SIGNAL(clicked()), this, SLOT(showAll()));

connect(btnShowS, SIGNAL(clicked()), this, SLOT(showSemiCond()));

connect(btnShowI, SIGNAL(clicked()), this, SLOT(showIns()));

connect(btnShowC, SIGNAL(clicked()), this, SLOT(showCond()));

}

void showForm::fulfill()

{

tableFile res;

int i = 0;

while (res.readElem())

{

source[i] = res.buf;

i++;

}

amount = i;

}

void showForm::makeclear()

{

table -> clearContents();

}

void showForm::showAll()

{

fulfill();

makeclear();

QTableWidgetItem \*item;

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

for (int j = 0; j < 4; j++)

{

item = new QTableWidgetItem();

item -> setFlags(Qt::NoItemFlags);

switch (j)

{

case 0:

item -> setText(source[i].number);

break;

case 1:

item -> setText(source[i].name);

break;

case 2:

item -> setText(source[i].weight);

break;

case 3:

item -> setText(source[i].conduct);

break;

}

table -> setItem(i, j, item);

}

}

void showForm::showSemiCond()

{

fulfill();

makeclear();

QTableWidgetItem \*item;

int row = 0;

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

if (source[i].conduct == "Semiconductor")

{

item = new QTableWidgetItem();

item -> setFlags(Qt::NoItemFlags);

item -> setText(source[i].name);

table -> setItem(row, 1, item);

item = new QTableWidgetItem();

item -> setFlags(Qt::NoItemFlags);

item -> setText(source[i].weight);

table -> setItem(row, 2, item);

row++;

}

}

void showForm::showIns()

{

if (lineI -> text() != "")

{

fulfill();

makeclear();

QTableWidgetItem \*item;

int row = 0;

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

if ((source[i].weight.toDouble() <= (lineI -> text()).toDouble())

&& (source[i].conduct == "Insulator"))

{

item = new QTableWidgetItem();

item -> setFlags(Qt::NoItemFlags);

item -> setText(source[i].name);

table -> setItem(row, 1, item);

row++;

}

}

}

void showForm::showCond()

{

fulfill();

makeclear();

elem conds[10];

int row = 0;

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

if (source[i].conduct == "Conductor")

{

conds[row] = source[i];

row++;

}

elem temp;

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

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

if (conds[j].weight.toDouble() > conds[j+1].weight.toDouble())

{

temp = conds[j];

conds[j] = conds[j+1];

conds[j+1] = temp;

}

QTableWidgetItem \*item;

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

for (int j = 0; j < 4; j++)

{

item = new QTableWidgetItem();

item -> setFlags(Qt::NoItemFlags);

switch (j)

{

case 0:

item -> setText(conds[i].number);

break;

case 1:

item -> setText(conds[i].name);

break;

case 2:

item -> setText(conds[i].weight);

break;

case 3:

item -> setText(conds[i].conduct);

break;

}

table -> setItem(i, j, item);

}

}

(tableFile.h)

#ifndef TABLEFILE\_H

#define TABLEFILE\_H

#include <QFile>

struct elem

{

QString name, conduct, number, weight;

};

class tableFile

{

QFile \*f;

public:

elem buf;

tableFile();

~tableFile();

void addElem(elem a);

bool readElem();

void delFile();

};

#endif // TABLEFILE\_H

(tableFile.cpp)

#include "tableFile.h"

#include "mainForm.h"

tableFile::tableFile()

{

f = new QFile ("table.txt");

f -> *open*(QFile::ReadWrite);

}

tableFile::~tableFile()

{

f -> *close*();

delete f;

}

void tableFile::addElem(elem a)

{

f -> *seek*(f -> *size*());

QDataStream out(f);

out << a.number << a.name << a.weight << a.conduct;

}

bool tableFile::readElem()

{

QDataStream in(f);

if (in.atEnd())

return false;

else

{

in >> buf.number >> buf.name >> buf.weight >> buf.conduct;

return true;

}

}

void tableFile::delFile()

{

f -> *resize*(0);

}

(start.cpp)

#include "mainForm.h"

#include <QApplication>

int main(int argc, char \*argv[])

{

QApplication app(*argc*,argv);// создать приложение

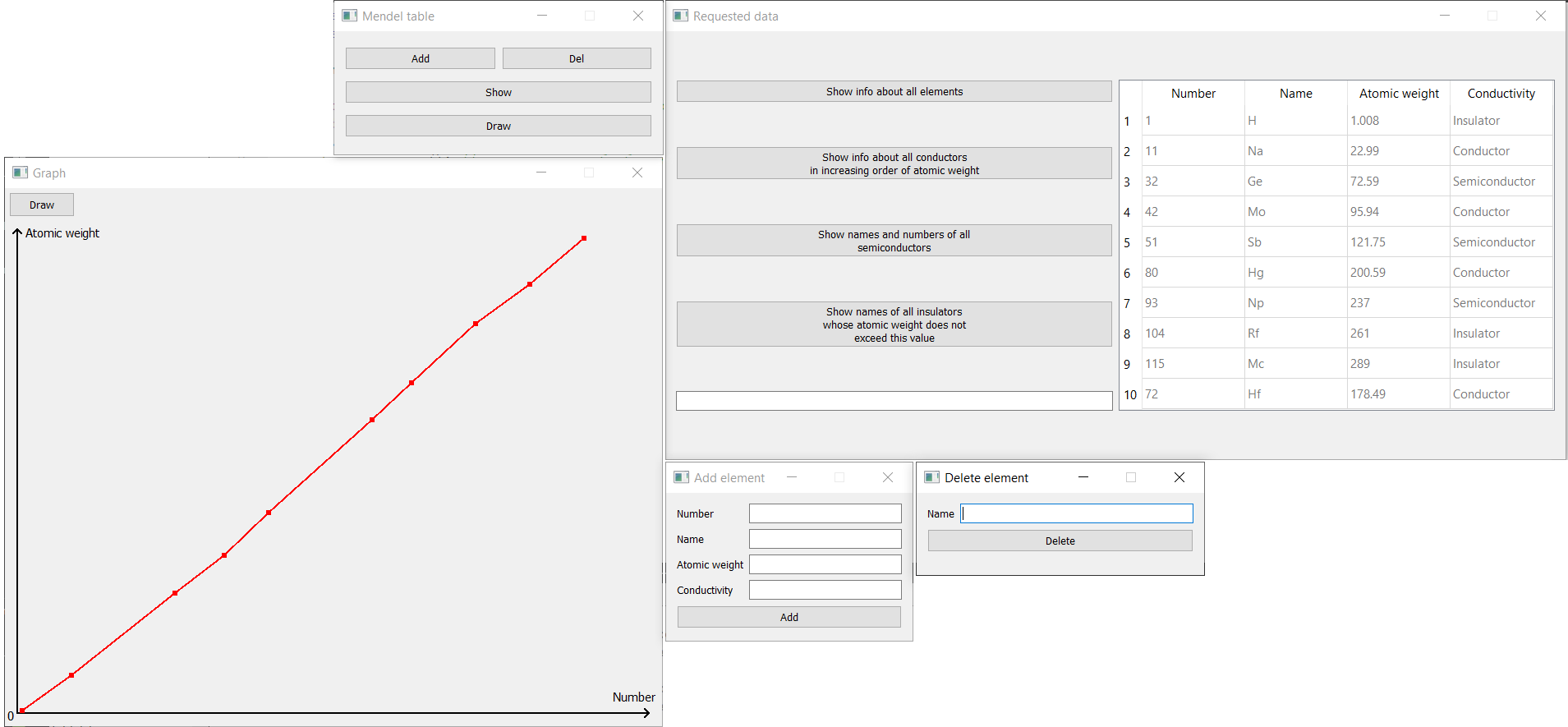
Window win; // создать Главное окно

win.show(); // показать на экране Главное окно

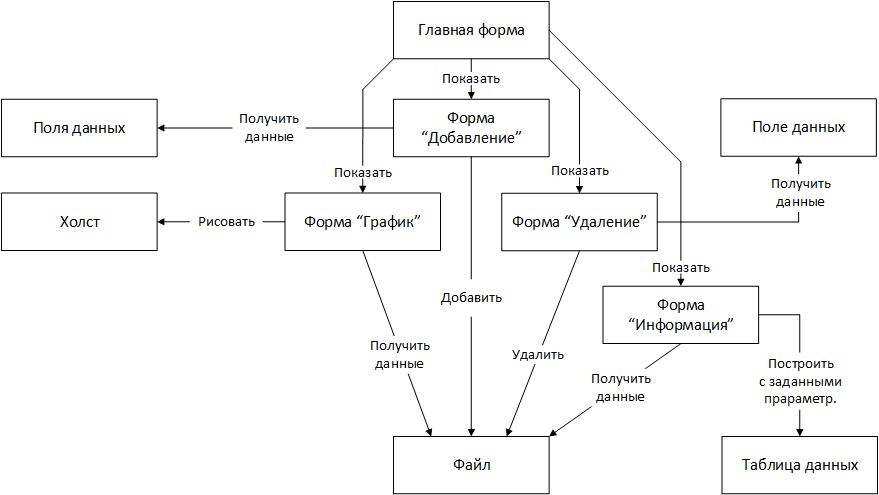
return app.exec(); // запуск цикла обработки сообщений

}

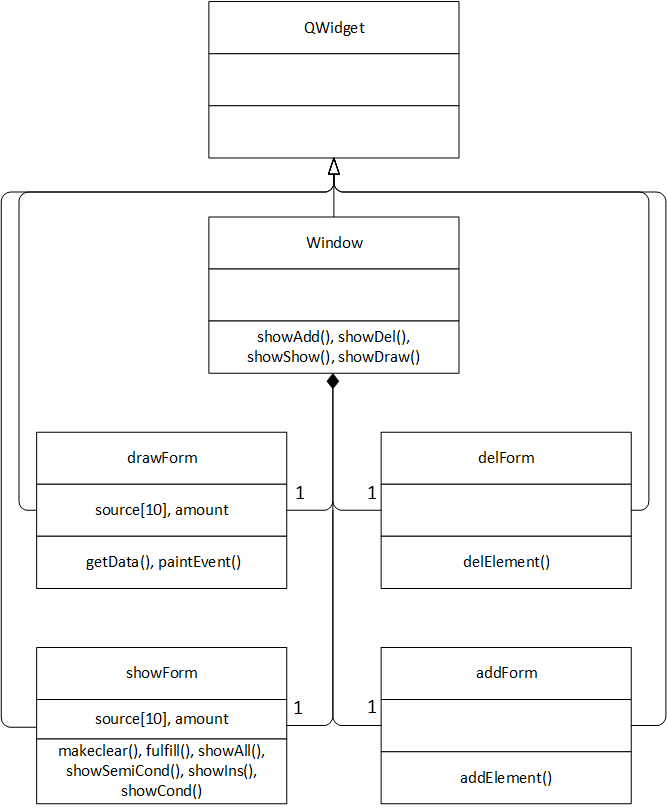
**Скриншоты:**

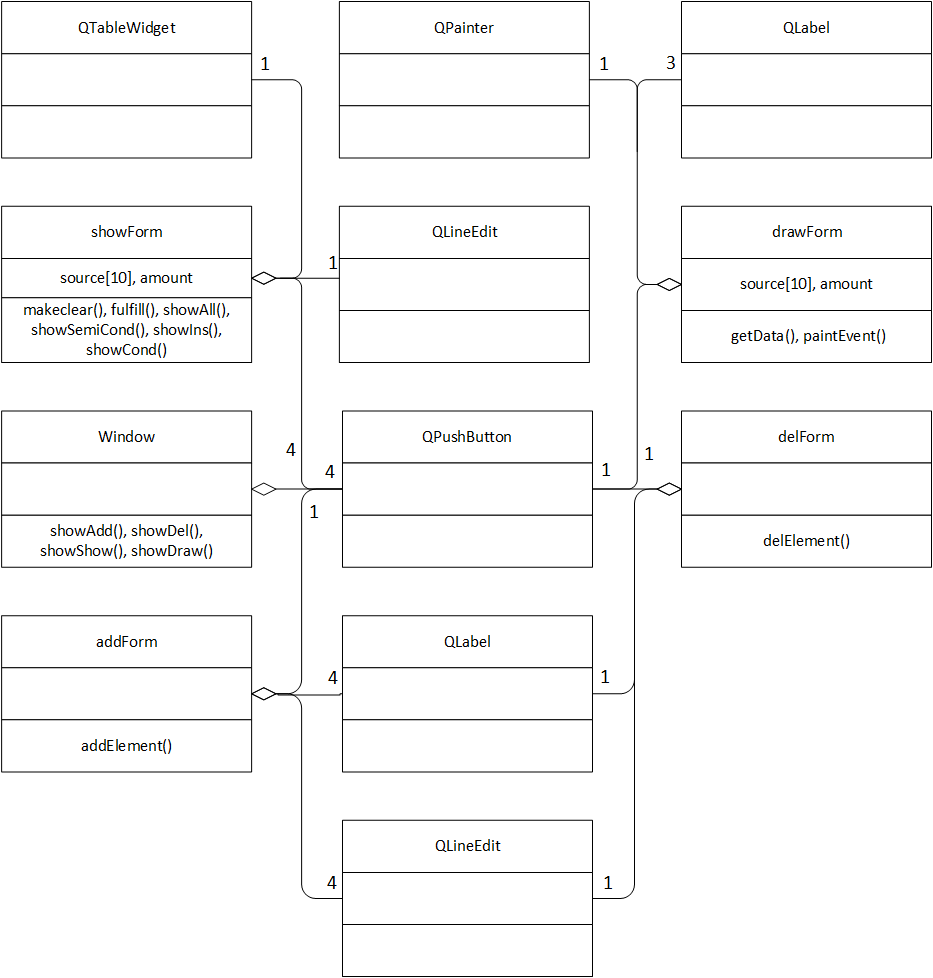
****

**Объектная декомпозиция:**

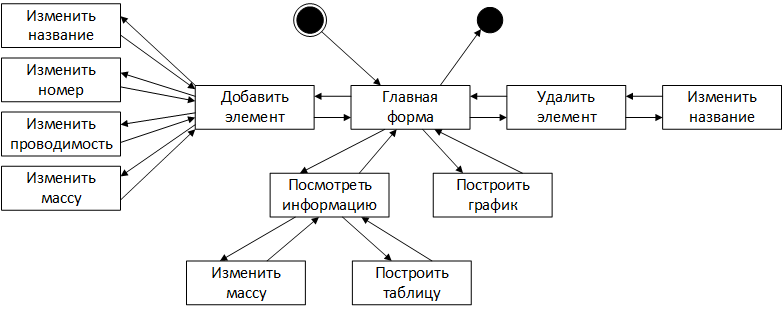
****

**Диаграмма классов:**

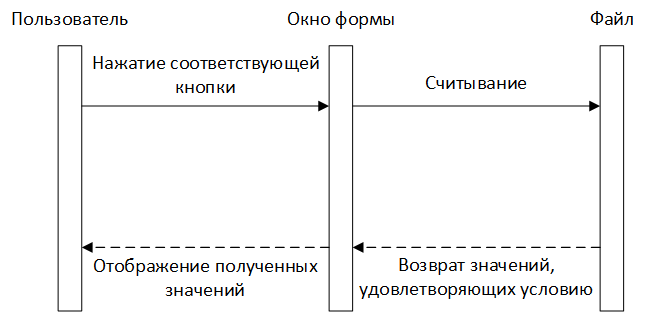
****

****

**Диаграмма состояний интерфейса:**

****

**Диаграмма последовательностей операции нахождения названий и атомных масс всех полупроводников:**

****

**Выводы:**

Изучены принципы работы с файлами Qt, а также получен опыт создания многооконных приложений и графических интерфейсов Qt.