**Задание**

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.

В сведениях о компьютерах представлены следующие характеристики: тип микропроцессора, объем памяти, объем винчестера, цена. Программа должна в интерактивном режиме формировать файл, добавлять и удалять данные, а также воспринимать каждый из перечисленных запросов и давать на него ответ.

1. Определить характеристики компьютеров, цена которых не превышает данную.

2. Определить типы микропроцессоров и цены всех компьютеров с памятью не менее заданного объема.

3. Определить цены всех компьютеров с данным типом микропроцессора, обладающих одновременно памятью и винчестерами не менее заданных объемов.

4. Построить график зависимости стоимости компьютера от объема памяти.

**Исходный код**

(файл front.cpp)

*#include <QApplication>*

*#include "back.h"*

*//#include "que.h"*

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

*{*

*QApplication app(argc, argv);*

*FormDialog \*dialog = new FormDialog();*

*dialog->show(); // отображаем окно*

*return app.exec(); // запускаем цикл обработки сообщений*

*}*

(файл back.h)

*#ifndef BACK\_H\_*

*#define BACK\_H\_*

*#include <QDialog>*

*#include <QLineEdit>*

*#include <QSignalMapper>*

*#include <QTextEdit>*

*#include <QString>*

*#include <QComboBox>*

*#include <QSpinBox>*

*#include <QTableWidget>*

*#include <QTableWidgetItem>*

*#include <QPainter>*

*#include <QWidget>*

*class CDrawer : public QWidget {*

*public:*

*CDrawer(QWidget \*parent = 0);*

*protected:*

*void paintEvent(QPaintEvent \*event);*

*void builder(QPainter \*qp);*

*};*

*class GraphDialog: public QWidget*

*{*

*Q\_OBJECT*

*public:*

*GraphDialog( QWidget \* parent = 0);*

*virtual ~GraphDialog(){};*

*protected:*

*int i1;*

*CDrawer \*drawer2;*

*private slots:*

*};*

*class TableDialog: public QWidget*

*{*

*Q\_OBJECT*

*public:*

*TableDialog( QWidget \* parent = 0);*

*virtual ~TableDialog(){};*

*protected:*

*bool checker(int k1,int k2,int k3,int k4);*

*QTableWidget \*table;*

*private slots:*

*};*

*class FormDialog: public QWidget*

*{*

*Q\_OBJECT*

*public:*

*GraphDialog \*dialog1;*

*TableDialog \*table1;*

*FormDialog( QWidget \* parent = 0);*

*virtual ~FormDialog(){};*

*protected:*

*int i1, conditionId;*

*CDrawer \*drawer1;*

*QTableWidget \*table;*

*QSpinBox \*spin1,\*spin2,\*spin3;*

*QComboBox \*combo1;*

*bool lower, isOut;*

*bool checker(int k1,int k2,int k3,int k4);*

*void outer();*

*private slots:*

*void outer0();*

*void outer1();*

*void outer2();*

*void outer3();*

*void adder();*

*void i1refr(int x);*

*void grapher();*

*void remover();*

*};*

*#endif*

(файл back.cpp)

*#include <QPushButton>*

*#include <QVBoxLayout>*

*#include <QHBoxLayout>*

*#include <QTextEdit>*

*#include <QLineEdit>*

*#include <QString>*

*#include <QLabel>*

*#include "back.h"*

*#include "fwork.h"*

*#define GWIDTH 720*

*#define GHEIGHT 480*

*#define GBORDER 45*

*#define GTB 12*

*using namespace std;*

*CBase oper;*

*int operId;*

*int f1,f2,f3,f4;*

*CDrawer::CDrawer(QWidget \*parent): QWidget(parent)*

*{*

*setFixedSize(QSize(GWIDTH,GHEIGHT));*

*}*

*void CDrawer::paintEvent(QPaintEvent \*e) {*

*Q\_UNUSED(e);*

*QPainter qpp(this);*

*builder(&qpp);*

*}*

*void CDrawer::builder(QPainter \*qp){*

*QPen pen(Qt::black, 2, Qt::SolidLine);*

*QRectF rect(0, 0, GWIDTH, GHEIGHT);*

*qp->eraseRect(rect);*

*qp->setPen(pen);*

*int mnx,mxx,mny,mxy;*

*oper.getmm(&mny,&mxy,&mnx,&mxx);*

*float kx,ky;*

*kx = 1.0 \* (GWIDTH-GBORDER\*2) / (mxx - mnx);*

*ky = 1.0 \* (GHEIGHT-GBORDER\*2) / (mxy - mny);*

*FILE\* f;*

*bool safety;*

*int x1,x2,y1,y2, k1,k2,k3,k4;*

*fopen\_s(&f,"base.dat", "r+b");*

*rewind(f);*

*safety = oper.gett(f,&k1,&k2,&x1,&y1);*

*qp->setFont(QFont("Arial", 14));*

*qp->drawText(int(GWIDTH/2.0) , GHEIGHT - (GBORDER/2.0) + GTB, "Disk space");*

*qp->drawText(2, int(GHEIGHT/2.0) , "Cost");*

*qp->setFont(QFont("Arial", 8));*

*while(safety){*

*safety = oper.gett(f,&k1,&k2,&x2,&y2);*

*pen.setStyle(Qt::SolidLine);*

*pen.setWidth(3);*

*pen.setColor(Qt::red);*

*qp->setPen(pen);*

*qp->drawLine(GBORDER + int(kx\*x1), GHEIGHT - int(ky\*y1) - GBORDER,GBORDER + int(kx\*x2),GHEIGHT - int(ky\*y2) - GBORDER);*

*pen.setStyle(Qt::DashLine);*

*pen.setWidth(1);*

*pen.setColor(Qt::black);*

*qp->setPen(pen);*

*qp->drawLine(GBORDER + int(kx\*x1), GHEIGHT - GBORDER,GBORDER + int(kx\*x1),GBORDER);*

*qp->drawLine(GBORDER + int(kx\*x2), GHEIGHT - GBORDER,GBORDER + int(kx\*x2),GBORDER);*

*qp->drawLine(GBORDER , GHEIGHT - int(ky\*y1) - GBORDER,GWIDTH - GBORDER,GHEIGHT - int(ky\*y1) - GBORDER);*

*qp->drawLine(GBORDER , GHEIGHT - int(ky\*y2) - GBORDER,GWIDTH - GBORDER,GHEIGHT - int(ky\*y2) - GBORDER);*

*qp->drawText(GBORDER + int(kx\*x1) - GTB, GHEIGHT - GBORDER - 3, QString::number(x1));*

*qp->drawText(GBORDER + int(kx\*x2) - GTB, GHEIGHT - GBORDER - 3, QString::number(x2));*

*if((y1!=0)&&(y2!=0)){*

*qp->drawText(GBORDER/2, GHEIGHT - int(ky\*y1) - GBORDER - GTB, QString::number(y1));*

*qp->drawText(GBORDER/2, GHEIGHT - int(ky\*y2) - GBORDER - GTB, QString::number(y2));*

*}*

*x1 = x2;*

*y1 = y2;*

*}*

*fclose(f);*

*}*

*GraphDialog::GraphDialog(QWidget \* parent){*

*this->setWindowTitle("Database manager");*

*drawer2 = new CDrawer(this);*

*QHBoxLayout \*mainLayout2 = new QHBoxLayout();*

*mainLayout2 ->addWidget(drawer2);*

*setLayout(mainLayout2);*

*}*

*bool FormDialog::checker(int k1, int k2,int k3,int k4){*

*bool res;*

*switch(conditionId){*

*case 0: res = true; break;*

*case 1: res =(k4 < spin1->value()); break;*

*case 2: res = (k2 > spin2->value()); break;*

*case 3: res = (k2 > spin2->value()&&(k3 > spin3->value())&&(k1==combo1->currentIndex())); break;*

*}*

*return res;*

*}*

*bool TableDialog::checker(int k1, int k2,int k3,int k4){*

*bool res;*

*switch(operId){*

*case 0: res = true; break;*

*case 1: res =(k4 < f4); break;*

*case 2: res = (k2 > f3); break;*

*case 3: res = ((k2 > f2)&&(k3 > f3)&&(k1==f1)); break;*

*}*

*res = (res) && (k3!=0) && (k4!=0);*

*return res;*

*}*

*TableDialog::TableDialog(QWidget \* parent){*

*oper.push(0,0,0,0);*

*this->setWindowTitle("Database manager");*

*table = new QTableWidget(0,4,this);*

*QStringList headers = { "proc type","ram", "disk space", "cost"};*

*table->setHorizontalHeaderLabels(headers);*

*QHBoxLayout \*mainLayout2 = new QHBoxLayout();*

*mainLayout2 ->addWidget(table);*

*setLayout(mainLayout2);*

*table->setRowCount(0);*

*table->setColumnCount(4);*

*FILE\* f;*

*bool safety;*

*int k1,k2,k3,k4;*

*fopen\_s(&f,"base.dat", "r+b");*

*rewind(f);*

*safety = oper.gett(f,&k1,&k2,&k3,&k4);*

*while(safety){*

*if(checker(k1,k2,k3,k4)){*

*table->setRowCount(table->rowCount() + 1);*

*QTableWidgetItem\* item = new QTableWidgetItem;*

*QString s;*

*switch(k1){*

*case 0:*

*s = "x32";*

*break;*

*case 1:*

*s = "x64";*

*break;*

*case 2:*

*s = "other";*

*break;*

*}*

*item->setText(s);*

*item->setTextAlignment(Qt::AlignCenter);*

*table->setItem(table->rowCount() - 1, 0, item);*

*item = new QTableWidgetItem;*

*s = QString::number(k2);*

*item->setText(s);*

*item->setTextAlignment(Qt::AlignCenter);*

*table->setItem(table->rowCount() - 1, 1, item);*

*item = new QTableWidgetItem;*

*s = QString::number(k3);*

*item->setText(s);*

*item->setTextAlignment(Qt::AlignCenter);*

*table->setItem(table->rowCount() - 1, 2, item);*

*item = new QTableWidgetItem;*

*s = QString::number(k4);*

*item->setText(s);*

*item->setTextAlignment(Qt::AlignCenter);*

*table->setItem(table->rowCount() - 1, 3, item);*

*}*

*safety = oper.gett(f,&k1,&k2,&k3,&k4);*

*}*

*fclose(f);*

*}*

*FormDialog::FormDialog(QWidget \* parent){*

*this->setWindowTitle("Database manager");*

*QHBoxLayout \*mainLayout = new QHBoxLayout();*

*QVBoxLayout \*layout1 = new QVBoxLayout();*

*QVBoxLayout \*layout2 = new QVBoxLayout();*

*QVBoxLayout \*layout3 = new QVBoxLayout();*

*QVBoxLayout \*layout4 = new QVBoxLayout();*

*QLabel \*l11 = new QLabel("Select by", this);*

*QLabel \*l12 = new QLabel("Uses all params", this);*

*QLabel \*l21 = new QLabel("Filter parametres", this);*

*QLabel \*l22 = new QLabel("(also used as input)", this);*

*QLabel \*l23 = new QLabel("Cost", this);*

*QLabel \*l24 = new QLabel("Proc type", this);*

*QLabel \*l25 = new QLabel("Ram", this);*

*QLabel \*l26 = new QLabel("Disk Space", this);*

*drawer1 = new CDrawer(this);*

*table = new QTableWidget(0,4,this);*

*QStringList headers = { "proc type","ram", "disk space", "cost"};*

*table->setHorizontalHeaderLabels(headers);*

*combo1 = new QComboBox(this);*

*combo1->addItem("x32");*

*combo1->addItem("x64");*

*combo1->addItem("other");*

*spin1 = new QSpinBox(this);*

*spin1->setMaximum(10000);*

*spin1->setMinimum(0);*

*spin2 = new QSpinBox(this);*

*spin2->setMaximum(10000);*

*spin2->setMinimum(0);*

*spin3 = new QSpinBox(this);*

*spin3->setMaximum(10000);*

*spin3->setMinimum(0);*

*QPushButton \*buttonS1 = new QPushButton("max cost");*

*QPushButton \*buttonS2 = new QPushButton("min ram");*

*QPushButton \*buttonS3 = new QPushButton("all params");*

*QPushButton \*buttonS4 = new QPushButton("output all");*

*QPushButton \*buttonS5 = new QPushButton("refresh graph");*

*QPushButton \*buttonM1 = new QPushButton("add");*

*QPushButton \*buttonM2 = new QPushButton("remove");*

*bool lower = true, isOut = false;*

*int i1;*

*i1 = spin1->value();*

*connect(buttonM1, SIGNAL(clicked()), this, SLOT(adder()));*

*connect(buttonS4, SIGNAL(clicked()), this, SLOT(outer0()));*

*connect(buttonS1, SIGNAL(clicked()), this, SLOT(outer1()));*

*connect(buttonS2, SIGNAL(clicked()), this, SLOT(outer2()));*

*connect(buttonS3, SIGNAL(clicked()), this, SLOT(outer3()));*

*connect(buttonS5, SIGNAL(clicked()), this, SLOT(grapher()));*

*connect(buttonM2, SIGNAL(clicked()), this, SLOT(remover()));*

*layout1->addWidget(l11);*

*layout1->addWidget(buttonS1);*

*layout1->addWidget(buttonS2);*

*layout1->addWidget(buttonS3);*

*layout1->addWidget(buttonS4);*

*layout1->addWidget(buttonS5);*

*layout1->addWidget(l12);*

*layout1->addWidget(buttonM1);*

*layout1->addWidget(buttonM2);*

*layout2->addWidget(l21);*

*layout2->addWidget(l22);*

*layout2->addWidget(l23);*

*layout2->addWidget(spin1);*

*layout2->addWidget(l24);*

*layout2->addWidget(combo1);*

*layout2->addWidget(l25);*

*layout2->addWidget(spin2);*

*layout2->addWidget(l26);*

*layout2->addWidget(spin3);*

*mainLayout->addLayout(layout1);*

*mainLayout->addLayout(layout2);*

*mainLayout->addWidget(table);*

*mainLayout->addWidget(drawer1);*

*table->hide();*

*drawer1->hide();*

*setLayout(mainLayout);*

*};*

*void FormDialog::i1refr(int x){*

*i1 = x;*

*spin2->setValue(x);*

*}*

*void FormDialog::adder(){*

*oper.push(combo1->currentIndex(),spin2->value(),spin3->value(),spin1->value());*

*}*

*void FormDialog::outer(){*

*table->setRowCount(0);*

*table->setColumnCount(4);*

*FILE\* f;*

*bool safety;*

*int k1,k2,k3,k4;*

*fopen\_s(&f,"base.dat", "r+b");*

*rewind(f);*

*safety = oper.gett(f,&k1,&k2,&k3,&k4);*

*while(safety){*

*if(checker(k1,k2,k3,k4)){*

*table->setRowCount(table->rowCount() + 1);*

*QTableWidgetItem\* item = new QTableWidgetItem;*

*QString s;*

*switch(k1){*

*case 0:*

*s = "x32";*

*break;*

*case 1:*

*s = "x64";*

*break;*

*case 2:*

*s = "other";*

*break;*

*}*

*item->setText(s);*

*item->setTextAlignment(Qt::AlignCenter);*

*table->setItem(table->rowCount() - 1, 0, item);*

*item = new QTableWidgetItem;*

*s = QString::number(k2);*

*item->setText(s);*

*item->setTextAlignment(Qt::AlignCenter);*

*table->setItem(table->rowCount() - 1, 1, item);*

*item = new QTableWidgetItem;*

*s = QString::number(k3);*

*item->setText(s);*

*item->setTextAlignment(Qt::AlignCenter);*

*table->setItem(table->rowCount() - 1, 2, item);*

*item = new QTableWidgetItem;*

*s = QString::number(k4);*

*item->setText(s);*

*item->setTextAlignment(Qt::AlignCenter);*

*table->setItem(table->rowCount() - 1, 3, item);*

*}*

*safety = oper.gett(f,&k1,&k2,&k3,&k4);*

*}*

*fclose(f);*

*}*

*void FormDialog::outer0(){*

*operId = 0;*

*f1*

*=combo1->currentIndex();*

*f2=spin2->value();*

*f3=spin3->value();*

*f4=spin1->value();*

*table1 = new TableDialog(this);*

*table1->show();*

*conditionId = 0;*

*outer();*

*}*

*void FormDialog::outer1(){*

*operId = 1;*

*f1 =combo1->currentIndex();*

*f2=spin2->value();*

*f3=spin3->value();*

*f4=spin1->value();*

*table1 = new TableDialog(this);*

*table1->show();*

*conditionId =1;*

*outer();*

*}*

*void FormDialog::outer2(){*

*operId = 2;*

*f1=combo1->currentIndex();*

*f2=spin2->value();*

*f3=spin3->value();*

*f4=spin1->value();*

*table1 = new TableDialog(this);*

*table1->show();*

*conditionId = 2;*

*outer();*

*}*

*void FormDialog::outer3(){*

*operId = 3;*

*f1 =combo1->currentIndex();*

*f2=spin2->value();*

*f3=spin3->value();*

*f4=spin1->value();*

*table1 = new TableDialog(this);*

*table1->show();*

*conditionId =3;*

*outer();*

*}*

*void FormDialog::grapher(){*

*oper.sort();*

*dialog1 = new GraphDialog(this);*

*dialog1->show();*

*this->drawer1->repaint();*

*}*

*void FormDialog::remover(){*

*oper.deletee(combo1->currentIndex(),spin2->value(),spin3->value(),spin1->value());*

*}*

(файл fwork.h)

*#ifndef FWORK\_H\_*

*#define FWORK\_H\_*

*#include <iostream>*

*#include <QDebug>*

*#include <QFile>*

*#include <QDataStream>*

*// struct comp {*

*// int ptype;*

*// int ram;*

*// int hdd;*

*// int cost;*

*// };*

*class CBase {*

*public:*

*//FILE\* f;*

*// void sort()*

*void push(int a1, int a2, int a3, int a4);*

*bool gett(FILE \*f1, int \*a1, int \*a2, int \*a3, int \*a4);*

*void getmm(int\* a1, int\* a2, int\* a3, int\* a4);*

*void deletee(int a1, int a2, int a3, int a4);*

*void sort();*

*};*

*#endif*

(файл fwork.cpp)

*#include "fwork.h"*

*struct comp {*

*int ptype;*

*int ram;*

*int hdd;*

*int cost;*

*};*

*void CBase::push(int a1, int a2, int a3, int a4) {*

*qDebug()<<2;*

*comp buf;*

*buf.ptype = a1;*

*buf.ram = a2;*

*buf.hdd = a3;*

*buf.cost = a4;*

*FILE\* f;*

*fopen\_s(&f,"base.dat", "a+b");*

*fwrite(&buf, sizeof(buf), 1, f);*

*fclose(f);*

*qDebug()<<3;*

*}*

*bool CBase::gett(FILE \*f1, int \*a1, int \*a2, int \*a3, int \*a4) {*

*comp buf;*

*if (!feof(f1)) {*

*fread(&buf, sizeof(buf), 1, f1);*

*}*

*if (!feof(f1)) {*

*\*a1 = buf.ptype;*

*\*a2 = buf.ram;*

*\*a3 = buf.hdd;*

*\*a4 = buf.cost;*

*}*

*return !(feof(f1));*

*}*

*void CBase::getmm(int\* a1, int\* a2, int\* a3, int\* a4) {*

*int minc, maxc, minh, maxh, p1, p2, p3, p4;*

*bool notall;*

*FILE\* f;*

*fopen\_s(&f,"base.dat", "r+b");*

*rewind(f);*

*notall = gett(f, &p1, &p2, &p3, &p4);*

*minc = p4;*

*maxc = p4;*

*minh = p3;*

*maxh = p3;*

*while (notall)*

*{*

*notall = gett(f, &p1, &p2, &p3, &p4);*

*if (notall) {*

*if (p4<minc) {*

*minc = p4;*

*}*

*if (p4>maxc) {*

*maxc = p4;*

*}*

*if (p3<minh) {*

*minh = p3;*

*}*

*if (p3>maxh) {*

*maxh = p3;*

*}*

*}*

*}*

*\*a1 = minc;*

*\*a2 = maxc;*

*\*a3 = minh;*

*\*a4 = maxh;*

*fclose(f);*

*}*

*void CBase::deletee(int a1, int a2, int a3, int a4) {*

*bool notall;*

*int p1, p2, p3, p4;*

*FILE\* f,\*fcopy;*

*fopen\_s(&f,"base.dat", "r+b");*

*fopen\_s(&fcopy,"basecopy.dat", "w+b");*

*rewind(f);*

*notall = gett(f, &p1, &p2, &p3, &p4);*

*comp buf;*

*while (notall)*

*{*

*if(!((p1==a1)&&(p2==a2)&&(p3==a3)&&(p4==a4))){*

*buf.ptype = p1;*

*buf.ram = p2;*

*buf.hdd = p3;*

*buf.cost = p4;*

*fwrite(&buf, sizeof(buf), 1, fcopy);*

*}*

*notall = gett(f, &p1, &p2, &p3, &p4);*

*}*

*fclose(f);*

*fopen\_s(&f,"base.dat", "w+b");*

*rewind(fcopy);*

*notall = gett(fcopy, &p1, &p2, &p3, &p4);*

*while (notall)*

*{*

*buf.ptype = p1;*

*buf.ram = p2;*

*buf.hdd = p3;*

*buf.cost = p4;*

*fwrite(&buf, sizeof(buf), 1, f);*

*notall = gett(fcopy, &p1, &p2, &p3, &p4);*

*}*

*fclose(f);*

*fclose(fcopy);*

*}*

*void CBase::sort(){*

*comp buf, buf0,buf1,buf2, basea[50];*

*int p1,p2,p3,p4, n;*

*bool notall, sorted;*

*FILE\* f, \*fcopy;*

*long target;*

*int crasher;*

*crasher = 0;*

*sorted = false;*

*n = 0;*

*fopen\_s(&f,"base.dat", "r+b");*

*rewind(f);*

*notall = gett(f, &p1, &p2, &p3, &p4);*

*while (notall)*

*{*

*buf.ptype = p1;*

*buf.ram = p2;*

*buf.hdd = p3;*

*buf.cost = p4;*

*basea[n] = buf;*

*n++;*

*notall = gett(f, &p1, &p2, &p3, &p4);*

*}*

*while(!sorted){*

*sorted = true;*

*for(int i=1; i<n; i++){*

*if(basea[i-1].hdd>basea[i].hdd){*

*buf = basea[i-1];*

*basea[i-1] = basea[i];*

*basea[i] = buf;*

*sorted = false;*

*}*

*}*

*}*

*fclose(f);*

*fopen\_s(&f,"base.dat", "w+b");*

*for(int i=0; i<n; i++){*

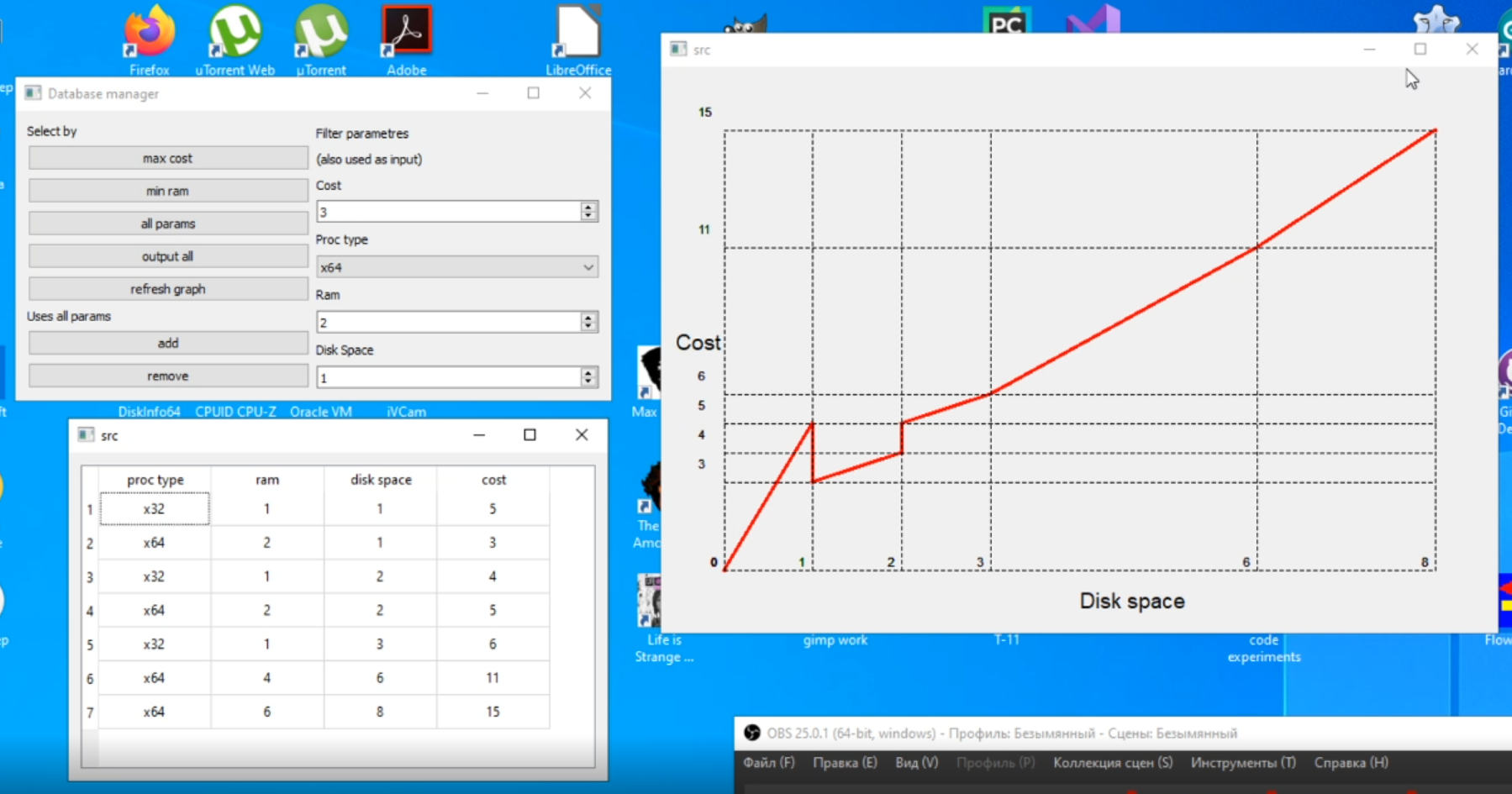
*fwrite(&basea[i], sizeof(buf), 1, f);*

*}*

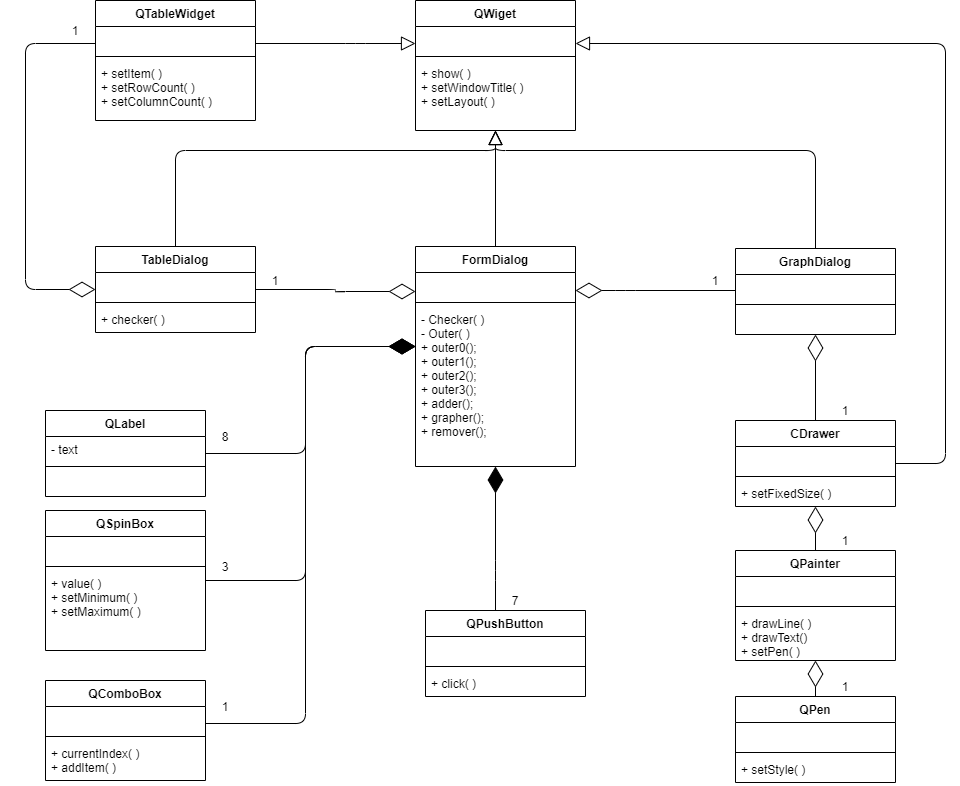
*fclose(f);*

*}*

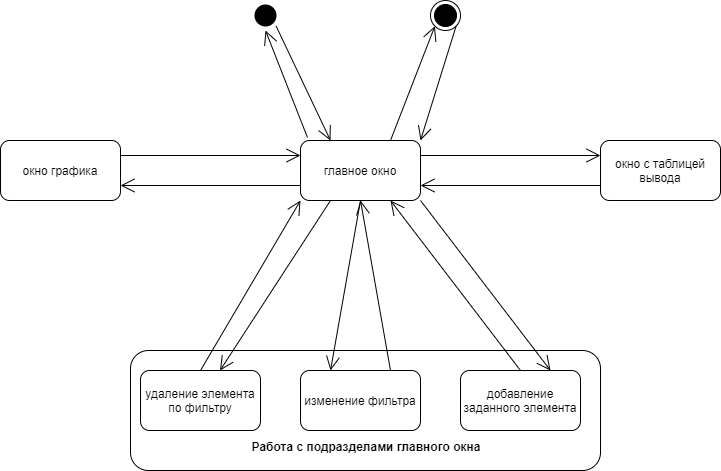
**Скриншоты**



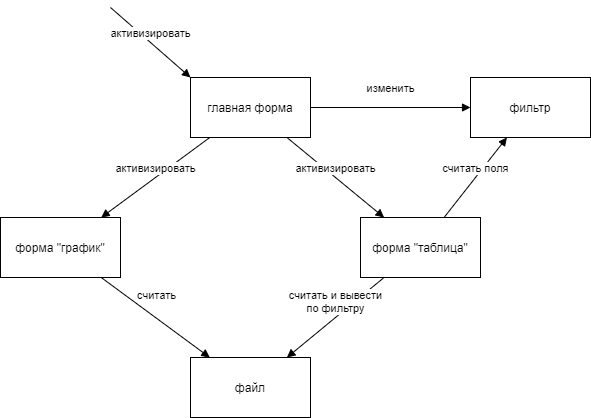
**Диаграмма классов**

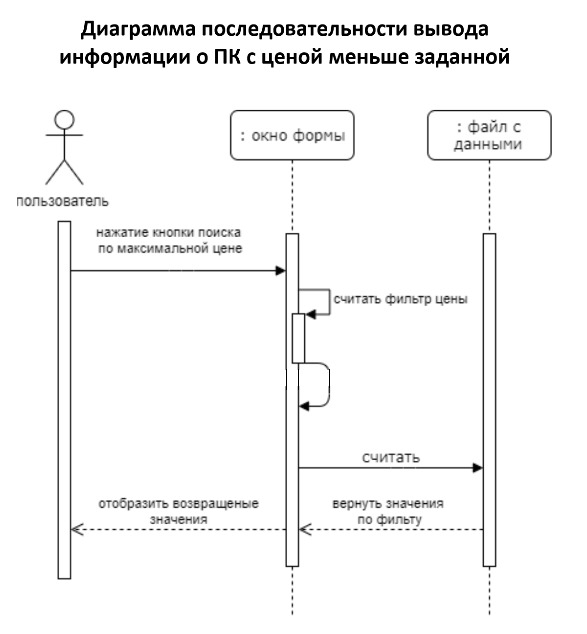


**Диаграмма состояний интерфейса**



**Объектная декомпозиция**



****

**Вывод**

С помощью Qt можно осуществлять работу с файлами(класс Qfile из Qt или стандартный тип FILE из C++), создавать многооконные приложения, графические интерфейсы, способные представлять информацию в виде таблиц и графиков.