ГУАП

КАФЕДРА № 43

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Старший преподаватель |  |  |  | Е.О. Шумова |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №5 |
| **«Использование контейнерных классов в среде Qt Creator»** |
|  |
| по курсу: ОБЪЕКТНО ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 4134к |  |  |  | Костяков Н.А. |
|  |  |  | подпись, дата |  | инициалы, фамилия |

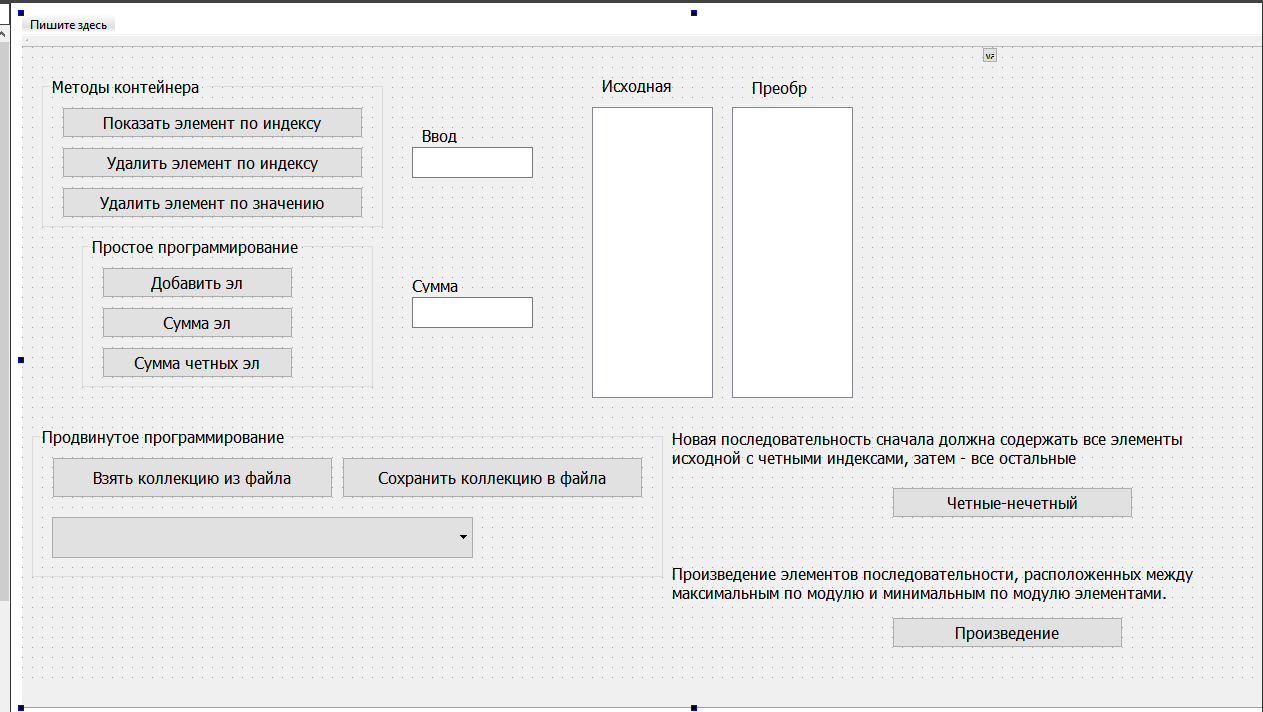
Санкт-Петербург 2022

**Цель работы:** научиться использовать контейнерные классы в среде Qt Creator.

Задание 4. Новая последовательность сначала должна содержать все элементы исходной с четными индексами, затем - все остальные (с сохранением исходного относительного порядка).

Произведение элементов последовательности, расположенных между максимальным по модулю и минимальным по модулю элементами

**Вид исходной программы**

****

**Листинг программы**

**Mainwindow.h**

#ifndef MAINWINDOW\_H

#define MAINWINDOW\_H

#include <QMainWindow>

#include <vector>

namespace Ui {

class MainWindow;

}

class MainWindow : public QMainWindow

{

Q\_OBJECT

public:

explicit MainWindow(QWidget \*parent = 0);

~*MainWindow*();

std::vector <int> vec = {32, 1, 45, 34, 23};

std::string paths[2] = {"C:\\qtprojects\\laba5\\file.txt", "C:\\qtprojects\\laba5\\file2.txt"};

private slots:

void on\_pushButton\_clicked();

void on\_pushButton\_2\_clicked();

void on\_pushButton\_3\_clicked();

void on\_pushButton\_4\_clicked();

void on\_pushButton\_5\_clicked();

void refresh();

void on\_pushButton\_6\_clicked();

void on\_pushButton\_7\_clicked();

void load(std::string path);

void save(std::string path);

void on\_pushButton\_8\_clicked();

void on\_pushButton\_9\_clicked();

void on\_pushButton\_11\_clicked();

void on\_pushButton\_10\_clicked();

private:

Ui::MainWindow \*ui;

};

#endif // MAINWINDOW\_H

**Main.cpp**

#include "mainwindow.h"

#include <QApplication>

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

{

QApplication a(argc, argv);

MainWindow w;

w.show();

return a.exec();

}

**Mainwindow.cpp**

#include "mainwindow.h"

#include "ui\_mainwindow.h"

#include "string"

#include "fstream"

MainWindow::MainWindow(QWidget \*parent) :

QMainWindow(parent),

ui(new Ui::MainWindow)

{

ui->setupUi(this);

setWindowTitle("KostyakovNikita 4134k");

refresh();

for(auto i : paths) ui->comboBox->addItem(QString::fromStdString(i));

ui->listWidget\_2->clear();

for(auto i: vec) ui->listWidget\_2->addItem(QString::number(i));

}

MainWindow::~*MainWindow*()

{

delete ui;

}

void MainWindow::refresh(){

ui->listWidget->clear();

for(auto i: vec) ui->listWidget->addItem(QString::number(i));

}

void MainWindow::load(std::string path){

std::ifstream input;

//C:\\qtprojects\\laba5\\file.txt

input.open(path);

vec.clear();

while (!input.eof())

{

std::string s;

std::getline(input, s);

if(s!=""){

int buf = std::stoi(s);

vec.push\_back(buf);

}

}

input.close();

refresh();

ui->listWidget\_2->clear();

for(auto i: vec) ui->listWidget\_2->addItem(QString::number(i));

}

void MainWindow::save(std::string path){

std::ofstream file;

file.open(path);

for (auto i : vec){

file<<i<<"\n";

}

file.close();

}

void MainWindow::on\_pushButton\_clicked()

{

if (ui->lineEdit->text()=="") return;

ui->listWidget->clear();

int index = std::stoi((ui->lineEdit->text()).toStdString());

ui->lineEdit->setText("");

ui->label\_7->setText(QString::number(vec[index]));

refresh();

}

void MainWindow::on\_pushButton\_2\_clicked()

{

if (ui->lineEdit->text()=="") return;

ui->listWidget->clear();

int index = std::stoi((ui->lineEdit->text()).toStdString());

ui->lineEdit->setText("");

auto iterator = std::begin(vec);

for (int i = 0; i<index ;i++) iterator = std::next(iterator);

vec.erase(iterator);

refresh();

}

void MainWindow::on\_pushButton\_3\_clicked()

{

if (ui->lineEdit->text()=="") return;

ui->listWidget->clear();

int index = std::stoi((ui->lineEdit->text()).toStdString());

ui->lineEdit->setText("");

auto iterator = 0;

while (index!=vec[iterator]){

iterator ++;

}

vec.erase(vec.begin()+iterator);

refresh();

}

void MainWindow::on\_pushButton\_4\_clicked()

{

if (ui->lineEdit->text()=="") return;

ui->listWidget->clear();

int index = std::stoi((ui->lineEdit->text()).toStdString());

ui->lineEdit->setText("");

vec.push\_back(index);

refresh();

}

void MainWindow::on\_pushButton\_5\_clicked()

{

int sum=0;

for(auto i : vec)

{

sum+=i;

}

ui->lineEdit\_3->setText(QString::number(sum));

refresh();

}

void MainWindow::on\_pushButton\_6\_clicked()

{

int sum=0;

for(auto i : vec) if(i%2==0) sum+=i;

ui->lineEdit\_3->setText(QString::number(sum));

refresh();

}

void MainWindow::on\_pushButton\_7\_clicked()

{

load((ui->comboBox->currentText()).toStdString());

}

void MainWindow::on\_pushButton\_8\_clicked()

{

save((ui->comboBox->currentText()).toStdString());

}

void MainWindow::on\_pushButton\_9\_clicked()

{

ui->listWidget->clear();

for (int i = 0; i<vec.size(); i+=2){

ui->listWidget->addItem(QString::number(vec[i]));

}

for (int i = 1; i<vec.size(); i+=2){

ui->listWidget->addItem(QString::number(vec[i]));

}

}

void MainWindow::on\_pushButton\_11\_clicked()

{

}

void MainWindow::on\_pushButton\_10\_clicked()

{

int min = 10000;

int max = -10000;

int i\_max=0, i\_min=0, it=0;

for (auto i: vec){

if(i>i\_max) {

max = i;

i\_max = it;

}

if(i<i\_min) {

min = i;

i\_min = it;

}

it++;

}

int mult = 1;

if (i\_min>i\_max){

int temp = i\_min;

i\_min = i\_max;

i\_max = temp;

}

for(int q = i\_min+1; q<i\_max; q++) mult\*=vec[q];

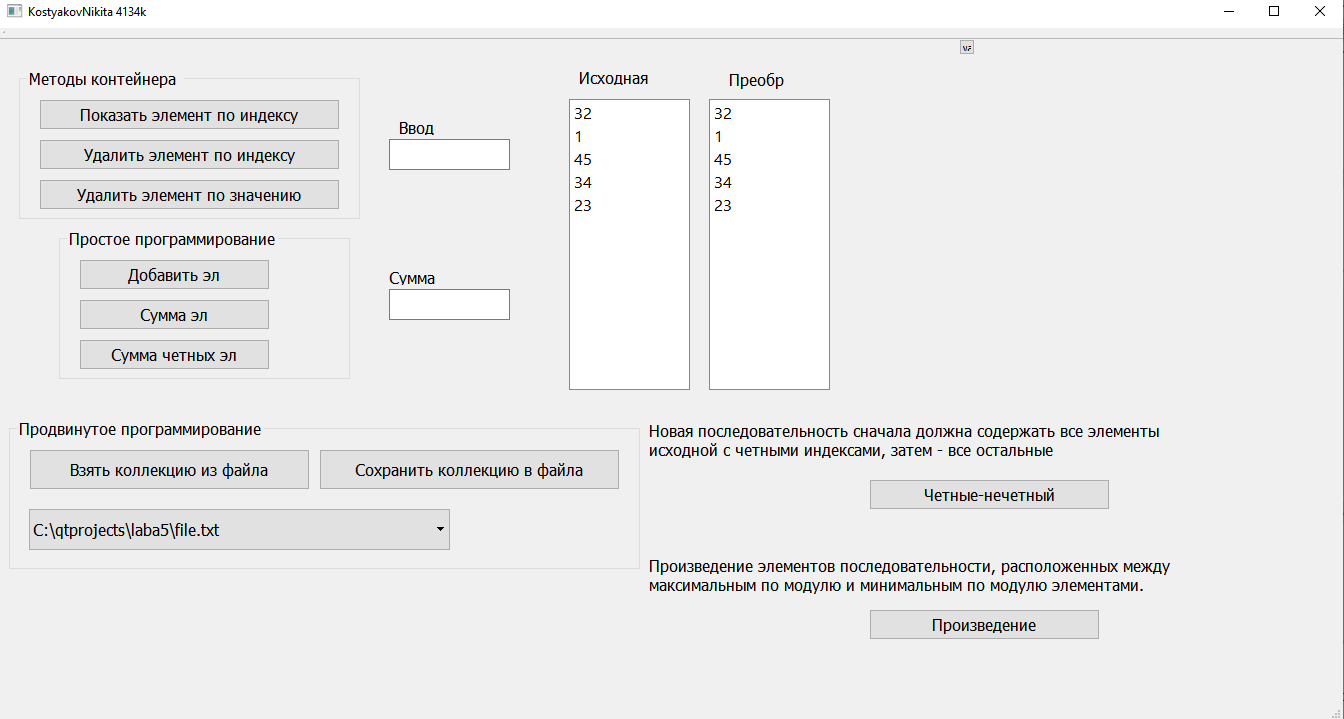
ui->listWidget->clear();

ui->listWidget->addItem(QString::number(mult));

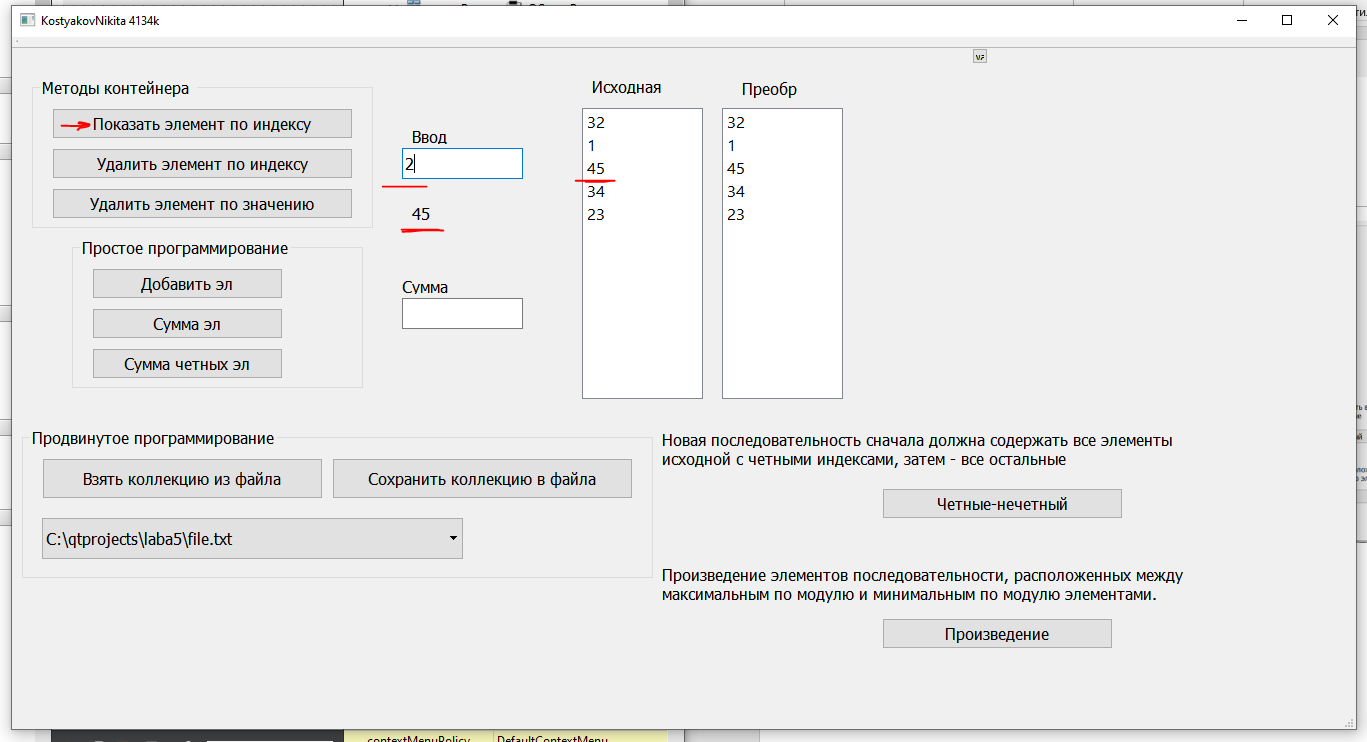
}

**Результат работы**

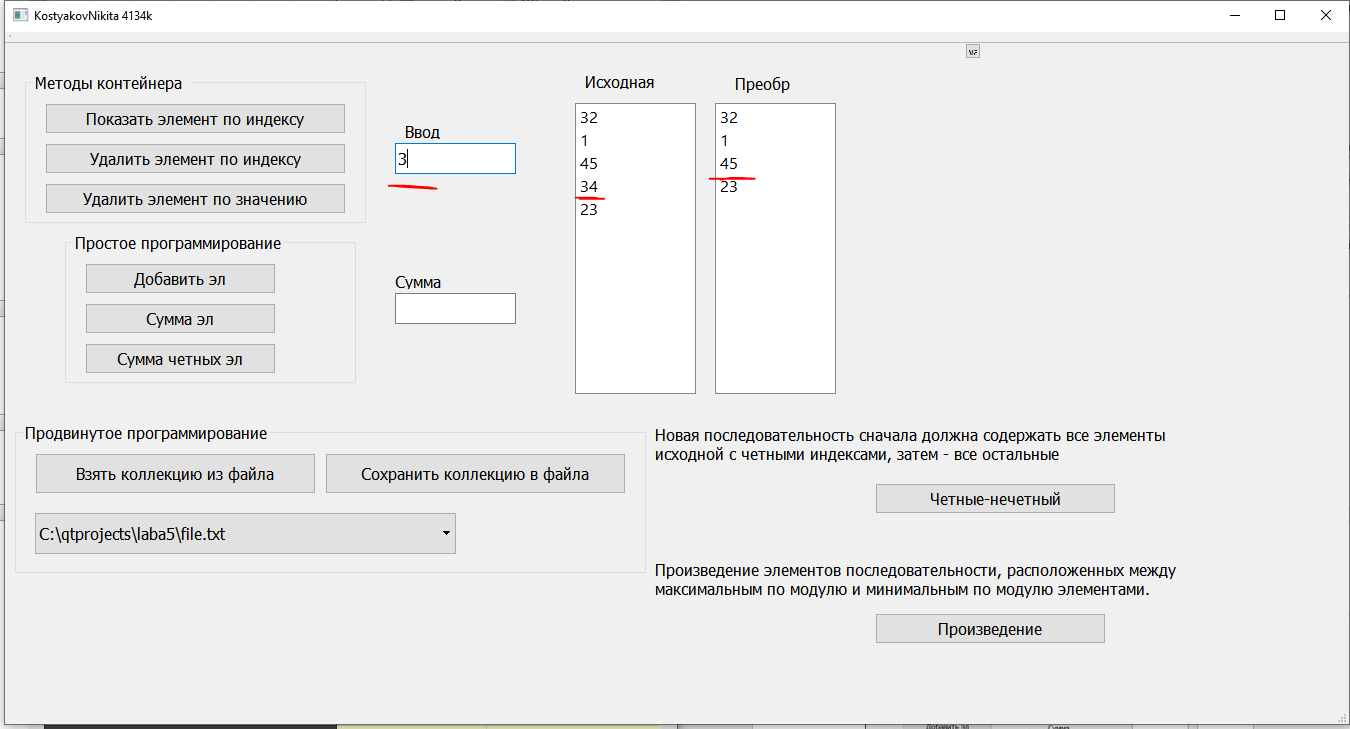
**Общий вид программы**

****

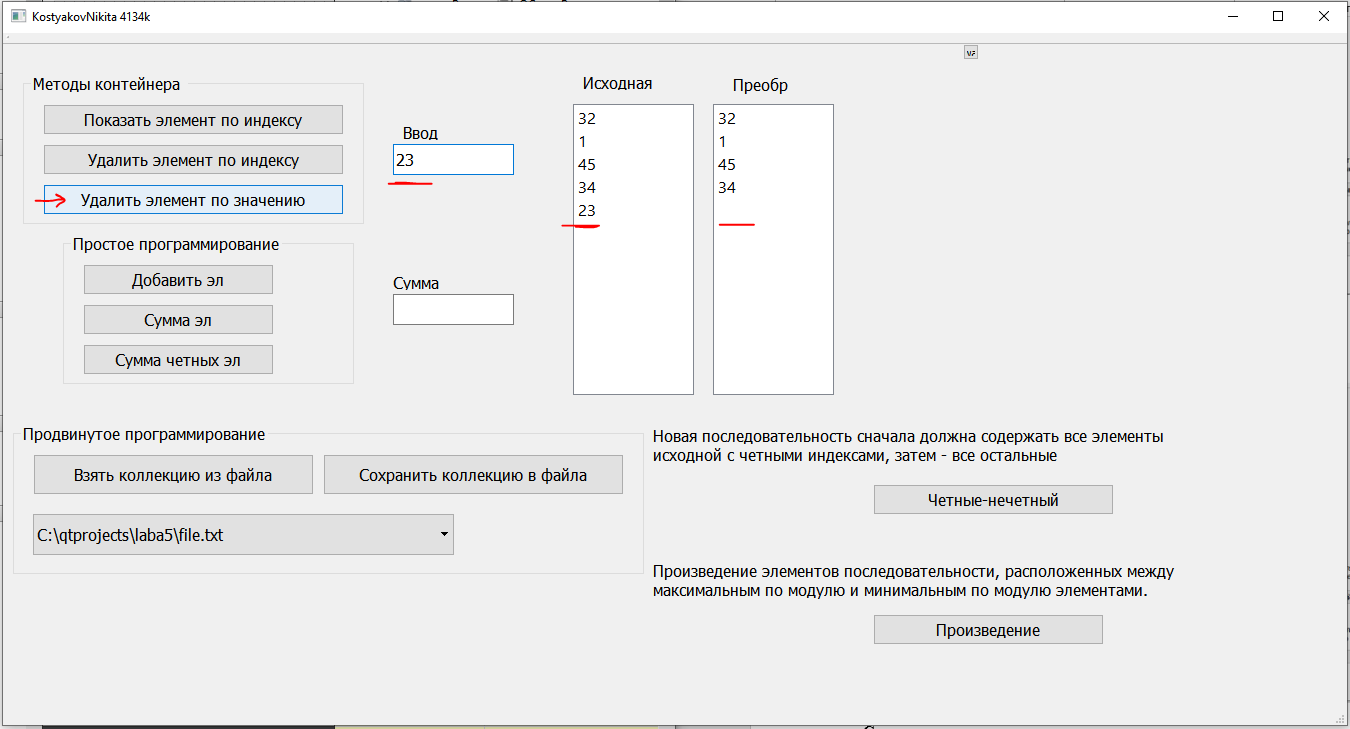
**Показать элемент по индексу**



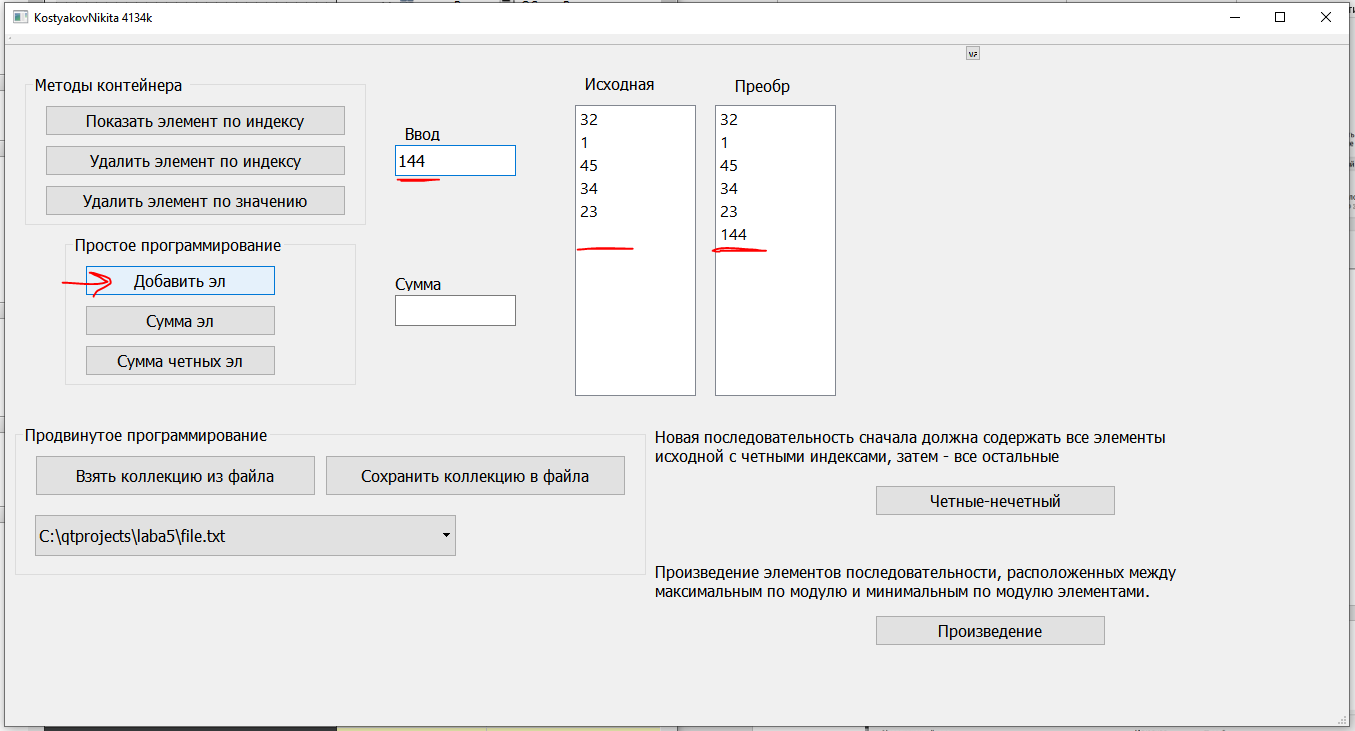
**Удалить элемент по индексу**



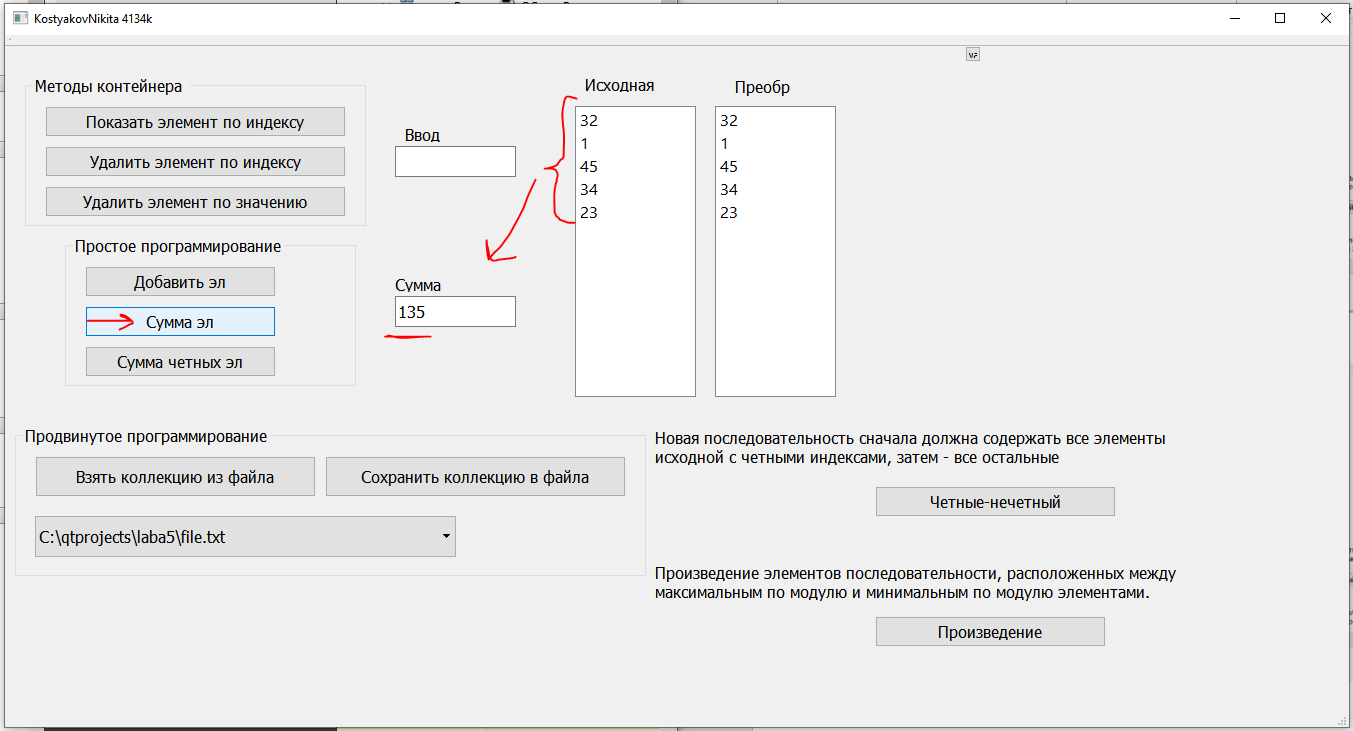
**Удалить элемент по значению**



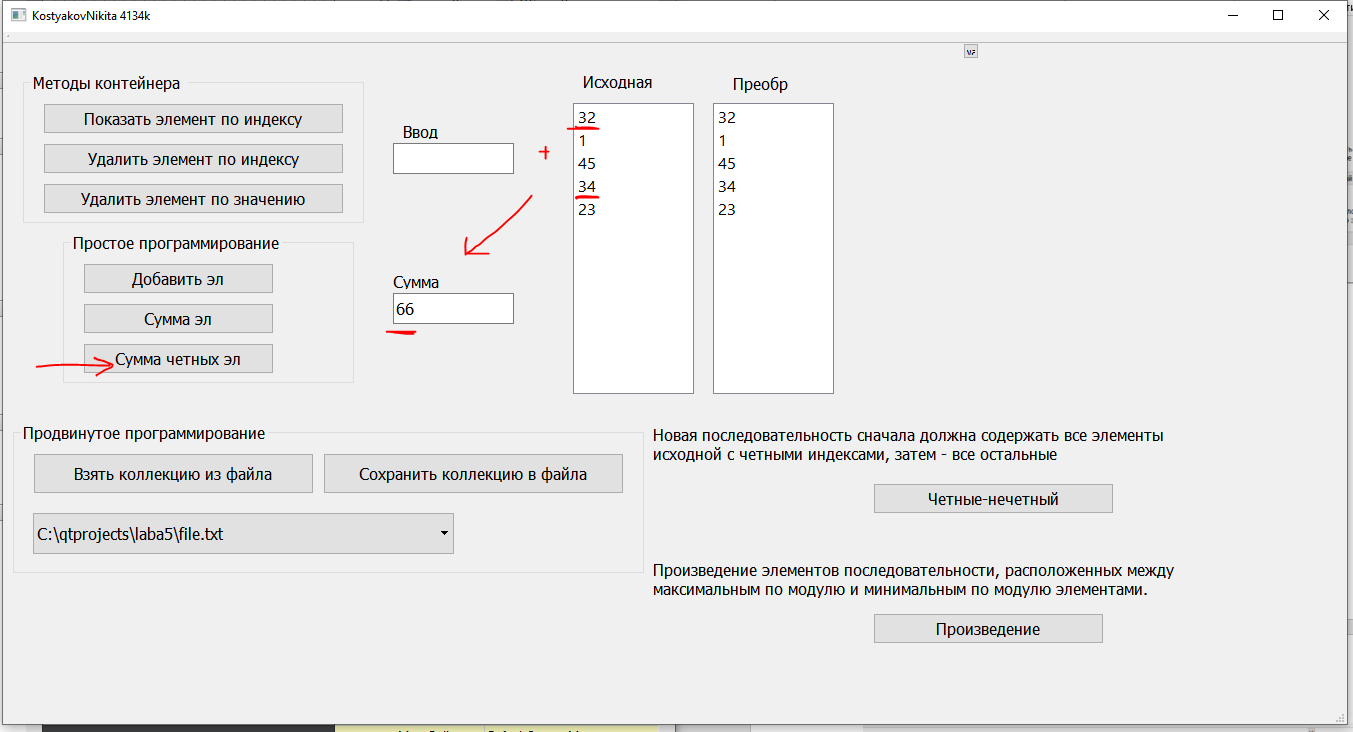
**Добавить эл**



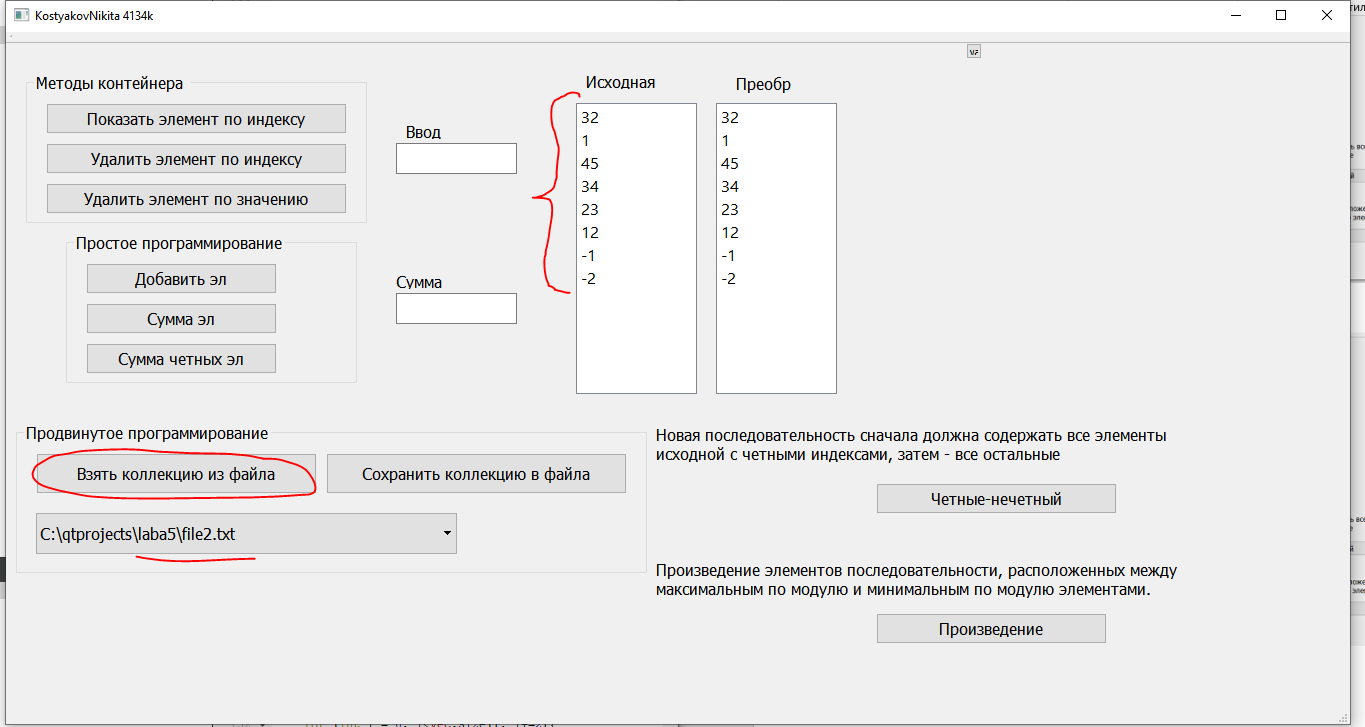
**Сумма эл**

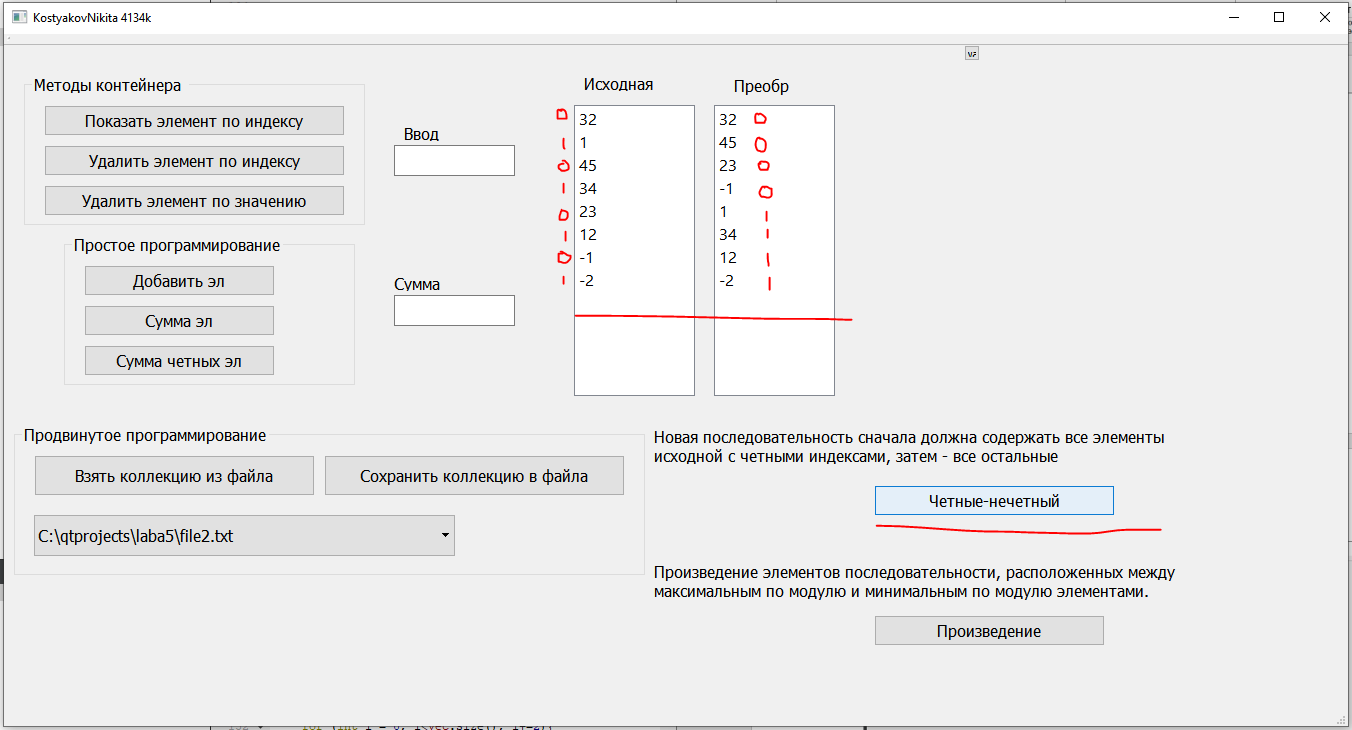


**Сумма четных эл**

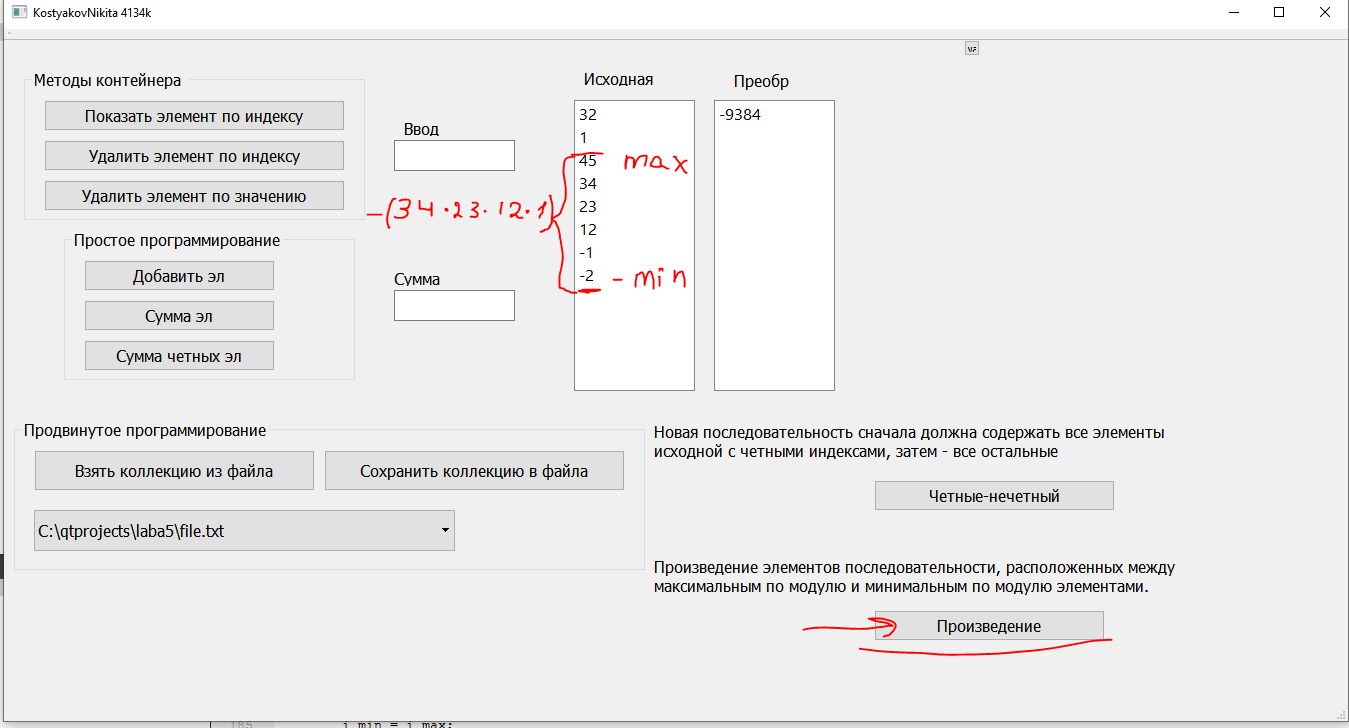


**Взять коллекцию из файла**



**Четный-нечетный**

**Произведение**



**Выводы**

Я научился использовать контейнерные классы в среде qtctreator