|  |  |
| --- | --- |
| 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 **Информатика и системы управления**

КАФЕДРА **Компьютерные системы и сети (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.03.03 Прикладная информатика**

**Отчет**

|  |  |
| --- | --- |
| **по лабораторной работе №** | 10 |

**Название:**

Qt. Создание контейнеров

**Дисциплина:** Объектно-ориентированное программирование

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ6-25 Б |  |  |  |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | Scan0023Scan0026 |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2023

**Вариант 8**

**Задание:** Моделировать дек, в качестве элементов которой могут использоваться целые числа и буквы. Операции: добавление элемента, удаление элемента, печать элементов очереди. При добавлении и удалении элементов выбирать сторону дека: начало или конец. Создать класс-потомок, который содержит функцию нахождения суммы всех чисел в деке. Тестировать полученную модель.

Разработать собственную иерархию классов, готовые контейнеры Qt не использовать. Пользовательский интерфейс для работы с моделью реализовать на Qt. В отчете представить диаграмму классов и обосновать выбранную структуру представления данных.

**Код модуля main.cpp:**

#include "stdialog.h"

#include <QApplication>

#include <QLocale>

#include <QTranslator>

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

{

QApplication a(argc, argv);

QTranslator translator;

const QStringList uiLanguages = QLocale::system().uiLanguages();

for (const QString &locale : uiLanguages) {

const QString baseName = "prog\_" + QLocale(locale).name();

if (translator.load(":/i18n/" + baseName)) {

a.installTranslator(&translator);

break;

}

}

stDialog \* dialog = new stDialog();

dialog->show();

return a.exec();

}

**Код модуля element.h:**

//описание самого списка

#ifndef ELEMENT\_H

#define ELEMENT\_H

#include <iostream>

#include <string>

#include <string.h>

#include <stdlib.h>

#include "Num.h"

using namespace std;

class Spisok{

protected: Element\*first, \*last, \*cur;

public: Spisok(){first=last=cur=nullptr;}

~Spisok(){

while (cur=first, cur!=nullptr){

first=first->suc;

delete cur;

}

};

void Add(Element\*a, bool f);//добавление элемента

void Del(bool f);//удаление элемента

Element \*First(){return cur=first;}

Element \*Next(){return cur=cur->suc;}

Element \*Last(){return cur=last;}

Element \*Prev(){return cur=cur->pre;}

};

class summa:public Spisok{

public:

summa():Spisok(){}

int Summa(){

int k, s=0;

cur=first;

while(cur!=nullptr){

if (TNum \*q=dynamic\_cast<TNum\*> (cur)) {

k = q->num;

s+=k;

}

cur=cur->suc;

}

return s;

}

};

#endif // ELEMENT\_H

**Код модуля element.cpp:**

#include "element.h"

void Spisok::Add(Element\*e, bool f){

if (first==nullptr) first=last=e;

else{if (f==0){e->suc=first;

first->pre=e;

first=e;}

else {

cur=last;

cur->suc=e;

e->pre=cur;

cur=e;

last=e;}}

}

void Spisok::Del(bool p){

if(p==1){

if (last!=nullptr){

last=last->pre;

if (last!=nullptr)last->suc=nullptr;}

}

else {

if (first!=nullptr){

first=first->suc;

if (first!=nullptr)first->pre=nullptr;

}

}

if (last==nullptr)first=nullptr;

}

**Код модуля Num.h:**

#ifndef NUM\_H

#define NUM\_H

class Element

{

public:

Element \*pre, \*suc;

Element(){pre=suc=nullptr;}

virtual ~Element(){};

};

class TNum:public Element{

public:

int num;

TNum(int n):Element(), num(n){}

~TNum()override{};

};

class TChar:public Element{

public:

char ch;

TChar(char c):Element(), ch(c){}

~TChar()override{};

};

#endif // NUM\_H

**Код модуля stdialog.h:**

#ifndef STDIALOG\_H

#define STDIALOG\_H

#include <QDialog>

#include <QLineEdit>

#include <QPushButton>

#include <QTextEdit>

#include <QRadioButton>

#include<QLabel>

class stDialog : public QDialog

{ Q\_OBJECT

public:

stDialog(QWidget \*parent = nullptr);

~stDialog(){};

protected:

QLineEdit\* Edit;

QLineEdit\* s;

QLabel\* slabel;

QPushButton\* addb;

QPushButton\* delb;

QRadioButton\* st;//стек или очередь

QRadioButton\* ord;

QPushButton\* Exit;

QTextEdit\* Text;

QString getEdit();

private slots:

/// Слот для обработки нажатий всех кнопок

void clicked();

void clicked1();

};

#endif // STDIALOG\_H

**Код модуля stdialog.cpp:**

#include "stdialog.h"

#include "element.h"

#include <QHBoxLayout>

#include <QVBoxLayout>

#include <QDebug>

summa N;

bool f=false;

stDialog::stDialog(QWidget \*parent)

{

this->setWindowTitle("Список");

Edit= new QLineEdit;

addb=new QPushButton;

delb=new QPushButton;

Exit=new QPushButton;

s=new QLineEdit;

slabel=new QLabel;

Text= new QTextEdit;

st=new QRadioButton;

ord=new QRadioButton;

slabel->setText("Сумма чисел: ");

addb->setText("Добавить");

delb->setText("Удалить");

st->setText("В начало");

ord->setText("В конец");

Exit->setText("Выйти");

Edit->setFocus();

s->setReadOnly(true);

Text->setReadOnly(true);

QVBoxLayout \*layout = new QVBoxLayout();

layout->addWidget(st);

layout->addWidget(ord);

layout->addWidget(Edit);

layout->addWidget(addb);

//setLayout(layout);

QVBoxLayout\*layout1=new QVBoxLayout();

layout1->addWidget(delb, 0, Qt::AlignBottom);

layout1->addWidget(slabel);

layout1->addWidget(s);

layout1->addWidget(Exit, 0, Qt::AlignBottom);

QHBoxLayout \*layout2=new QHBoxLayout();

layout2->addWidget(Text);

layout2->insertLayout(1, layout1, 0);

layout2->insertLayout(0, layout, 0);

setLayout(layout2);

connect(addb, SIGNAL (clicked()), this, SLOT (clicked()));

connect(delb, SIGNAL(clicked()), this, SLOT(clicked1()));

connect(Exit, SIGNAL(clicked()), this, SLOT(close()));

};

void stDialog::clicked(){

Text->clear();

QString str = getEdit();

Element \*p, \*cur;

if (str.toInt()!=0){

QString k=str.at(0);

p=new TNum(k.toInt());

}

else p=new TChar(str.toStdString().c\_str()[0]);

if (st->isChecked()) N.Add(p, 0);

else if (ord->isChecked()) N.Add(p, 1);

p=N.First();

cur=p;

while(cur!=nullptr){

if (TNum \*q=dynamic\_cast<TNum\*> (cur)) str=QString::number(q->num);

if (TChar \*q=dynamic\_cast<TChar\*>(cur)) str=q->ch;

Text->append(str);

cur=cur->suc;

}

Edit->clear();

Edit->setFocus();

str.setNum(N.Summa());

s->setText(str);

};

void stDialog::clicked1(){

Text->clear();

QString str;

Element \*p, \*cur;

if (st->isChecked()) N.Del(0);

else if (ord->isChecked()) N.Del(1);

p=N.First();

cur=p;

while(cur!=nullptr){

if (TNum \*q=dynamic\_cast<TNum\*> (cur)) str.setNum(q->num);

if (TChar \*q=dynamic\_cast<TChar\*>(cur)) str=q->ch;

Text->append(str);

cur=cur->suc;

}

Edit->clear();

Edit->setFocus();

str.setNum(N.Summa());

s->setText(str);

};

QString stDialog::getEdit(){

return Edit->text();

};

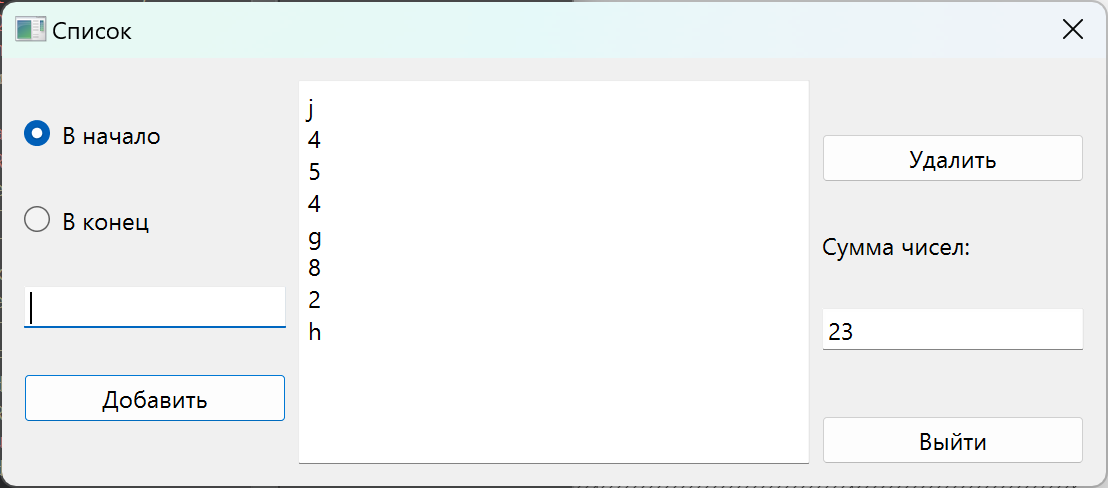


Рисунок 1 – работающая версия программы



Рисунок 2 – диаграмма классов

Я выбрала наполнение как механизм построения классов, чтобы можно было не зависеть от предопределённого количества элементов в списке и добавлять новые по мере необходимости.

**Вывод**: мы научились работать с контейнерами в Qt Creator, добавлять и удалять элементы из него из разных концов дека.