|  |  |
| --- | --- |
| 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 **09.03.01 Информатика и вычислительная техника**

**Отчет**

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

**Название:** Qt. Создание контейнеров

**Дисциплина:** Объектно-ориентированное программирование

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ6-22Б |  |  | Д.С.Твердюк |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | А.А. Веселовский |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

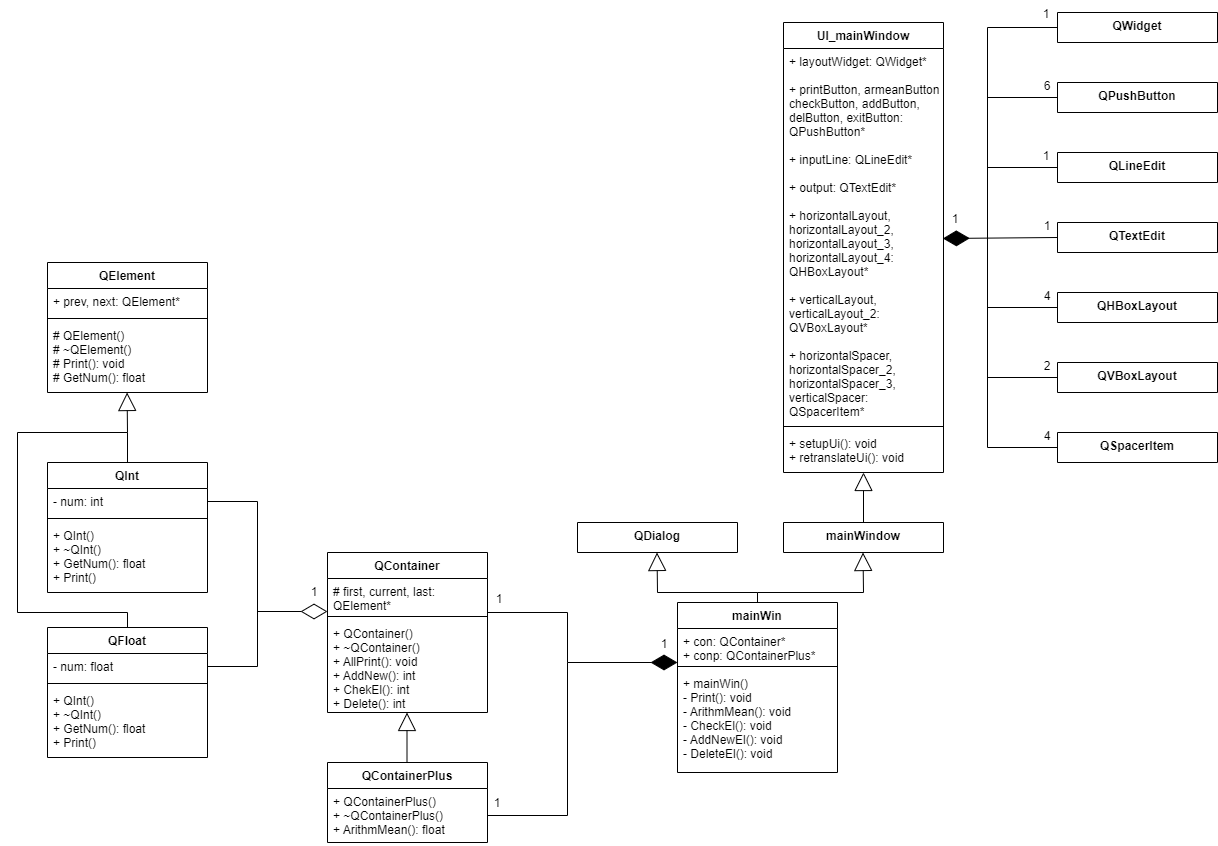
Москва, 2022

**Ход работы:**

Моделировать множество, в качестве элементов которого могут использоваться целые и вещественные числа. Операции: добавление элемента, удаление элемента, определение вхождения элемента во множество, печать элементов. Создать класс-потомок, который содержит функцию определения среднего арифметического целых чисел. Тестировать полученную модель.

Разработать собственную иерархию классов, готовые контейнеры Qt не использовать. Пользовательский интерфейс для работы с моделью реализовать на Qt.

Диаграмма классов:



Файл lab10.pro:

TEMPLATE = app

TARGET = lab10

QT += gui widgets

CONFIG += c++11 debug\_and\_release

DEFINES += QT\_DEPRECATED\_WARNINGS

#Input

SOURCES += \

container.cpp \

lab10.cpp \

mainWindow.cpp

HEADERS += \

container.h \

mainWindow.h

FORMS += \

mainWindow.ui

Файл lab10.cpp:

#include <QApplication>

#include "mainWindow.h"

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

QApplication app(argc, argv);

mainWin \*mw = new mainWin();

mw->show();

return app.exec();

}

Файл mainWindow.h:

#ifndef MAINWINDOW\_H

#define MAINWINDOW\_H

#include <QDialog>

#include "ui\_mainWindow.h"

#include "container.h"

class mainWin: public QDialog, public Ui::mainWindow {

Q\_OBJECT

public:

QContainer \* con;

QContainerPlus \* conp;

mainWin(QWidget \* parent = 0);

private slots:

void Print();

void ArithmMean();

void CheckEl();

void AddNewEl();

void DeleteEl();

};

#endif // MAINWINDOW\_H

Файл mainWindow.cpp:

#include "mainWindow.h"

mainWin::mainWin(QWidget \* parent): QDialog(parent) {

setupUi(this);

connect(printButton, SIGNAL(clicked()), this, SLOT(Print()));

connect(armeanButton, SIGNAL(clicked()), this, SLOT(ArithmMean()));

connect(checkButton, SIGNAL(clicked()), this, SLOT(CheckEl()));

connect(addButton, SIGNAL(clicked()), this, SLOT(AddNewEl()));

connect(delButton, SIGNAL(clicked()), this, SLOT(DeleteEl()));

con = new QContainer();

conp = new QContainerPlus();

}

void mainWin::Print(){

output->append("Содержимое множества:");

con->AllPrint(output);

output->append("-------------");

}

QString str;

void mainWin::ArithmMean() {

// \*conp = \*con;

str.setNum(conp->ArithmMean());

output->append("Среднее арифметическое целых чисел: " + str);

}

void mainWin::CheckEl() {

int res;

str = inputLine->text();

if (str != "") {

if (str.indexOf(".") != -1) {

QFloat \*f = new QFloat(str.toFloat());

res = con->CheckEl(f);

}

else {

QInt \*i = new QInt(str.toInt());

res = con->CheckEl(i);

}

if (res) {

output->append("Данный элемент входит в множество (Позиция " + str.setNum(res) + ")");

}

else output->append("Данный элемент отсутствует в множестве");

}

else output->append("Введена пустая строка");

inputLine->clear();

}

void mainWin::AddNewEl() {

str = inputLine->text();

*if* (str != "") {

*if* (str.indexOf(".") != -1) {

QFloat \*f = *new* QFloat(str.toFloat()), \*fp = *new* QFloat(str.toFloat());

*if* (!(con->AddNew(*f*))) {

conp->AddNew(*fp*);

output->append("Элемент успешно добавлен");

}

*else* output->append("Элемент уже входит во множество");

}

*else* {

QInt \*i = *new* QInt(str.toInt()), \*ip = *new* QInt(str.toInt());

*if* (!(con->AddNew(*i*))) {

conp->AddNew(*ip*);

output->append("Элемент успешно добавлен");

}

*else* output->append("Элемент уже входит во множество");

}

}

*else* output->append("Введена пустая строка");

inputLine->clear();

}

void mainWin::DeleteEl() {

str = inputLine->text();

if (str != "") {

if (str.indexOf(".") != -1) {

QFloat \*f = new QFloat(str.toFloat());

if (!(con->Delete(f))) {

conp->Delete(f);

output->append("Элемент успешно удалён");

}

else output->append("Элемент не найден");

}

else {

QInt \*i = new QInt(str.toInt());

if (!(con->Delete(i))) {

conp->Delete(i);

output->append("Элемент успешно удалён");

}

else output->append("Элемент не найден");

}

}

else output->append("Введена пустая строка");

inputLine->clear();

}

Файл container.h:

#ifndef CONTAINER\_H

#define CONTAINER\_H

#include <QTextEdit>

class QElement {

public:

QElement \*prev, \*next;

QElement(){prev = next = nullptr;};

virtual ~QElement(){};

virtual float GetNum() = 0;

virtual void Print(QTextEdit \* out) = 0;

};

class QContainer {

protected:

QElement \*first, \*current, \*last;

public:

QContainer(){first = current = last = nullptr;};

~QContainer();

void AllPrint(QTextEdit \* out);

int AddNew(QElement \*e);

int CheckEl(QElement \*e);

int Delete(QElement \*e);

};

class QContainerPlus: public QContainer {

public:

QContainerPlus(): QContainer(){};

~QContainerPlus(){};

float ArithmMean();

};

class QInt: public QElement {

private:

int num;

public:

QInt(int n): QElement(), num(n){};

~QInt(){};

float GetNum() override;

void Print(QTextEdit \* out) override;

};

class QFloat: public QElement {

private:

float num;

public:

QFloat(float n): QElement(), num(n){};

~QFloat(){};

float GetNum() override;

void Print(QTextEdit \* out) override;

};

#endif // CONTAINER\_H

Файл container.cpp:

#include "container.h"

#include "mainWindow.h"

#include <QString>

QContainer::~QContainer() {

while (first != nullptr) {

current = first->next;

delete first;

first = current;

}

}

void QContainer::AllPrint(QTextEdit \* out) {

current = first;

while (current != nullptr) {

current->Print(out);

current=current->next;

}

}

int QContainer::AddNew(QElement \*e) {

*if* (!CheckEl(*e*)) {

*if* (first == *nullptr*) {

first = last = e;

}

*else* {

last->next = e;

e->prev = last;

last = e;

}

*return* 0;

}

*else* *return* 1;

}

int QContainer::CheckEl(QElement \*e) {

int cnt = 0;

bool flag = 0;

current = first;

while ((current != nullptr) && (!flag)) {

cnt++;

if (e->GetNum() == current->GetNum()) {

flag = 1;

}

current = current->next;

}

if (!flag)

cnt = 0;

return cnt;

}

int QContainer::Delete(QElement \*e) {

current = first;

while ((current != nullptr) && (current->GetNum() != e->GetNum()))

current = current->next;

if (current != nullptr) {

if ((current == first) && (current == last)) {

delete first;

first = last = nullptr;

return 0;

}

else if (current == first) {

current = first->next;

delete first;

first = current;

first->prev = nullptr;

return 0;

}

else if (current == last) {

current = last->prev;

delete last;

last = current;

last->next = nullptr;

return 0;

}

else {

QElement \*wor;

wor = current;

current = current->prev;

current->next = wor->next;

wor->next->prev = current;

delete wor;

return 0;

}

}

else return 1;

}

float QContainerPlus::ArithmMean() {

float mean = 0, trash;

int cnt = 0;

current = first;

while (current != nullptr) {

if (modff(current->GetNum(), &trash) == 0) {

cnt ++;

mean += current->GetNum();

}

current = current->next;

}

if (cnt != 0) {

mean /= cnt;

return mean;

}

else return 0;

}

float QInt::GetNum() {

return num;

}

void QInt::Print(QTextEdit \* out) {

QString str;

str.setNum(num);

out->append(str + " ");

}

float QFloat::GetNum() {

return num;

}

void QFloat::Print(QTextEdit \* out) {

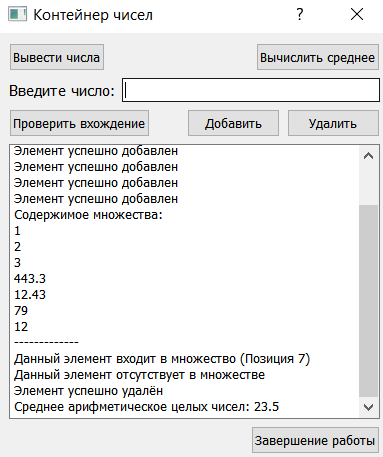
QString str;

str.setNum(num);

out->append(str + " ");

}

Результат работы написанной программы:



**Вывод:** программа работает корректно на заданных исходных данных.