Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа №3

По дисциплине: «ООТПиСП»

Тема: «Контейнеры в Qt 5»

Вариант 6

Выполнил:

Студент 3 курса

Группы ПО-7

Комиссаров А.Е.

Проверил:

Хацкевич М.В.

2022

**Цель работы:** Познакомиться с контейнерами и взаимодействием с ними в Qt 5.

**Задание 1: Вариант 4** Даны две переменные целого типа: A и B (переменные вводятся в **lineedit**). Если их значения не равны, то присвоить каждой переменной сумму этих значений, а если равны, то присвоить переменным нулевые значения.

**mainwindow.cpp**

#include "mainwindow.h"

#include "ui\_mainwindow.h"

MainWindow::**MainWindow**(QWidget \***parent**)

: QMainWindow(*parent*)

, ui(*new* Ui::MainWindow)

{

ui->setupUi(*this*);

}

MainWindow::~**MainWindow**()

{

*delete* ui;

}

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

{

QStringList **values** = ui->lineEdit->text().split(" ");

int **val1** = values[0].toInt();

int **val2** = values[1].toInt();

*if* (val1 == val2) {

ui->label->setText(QString("Result: 0 0"));

} *else* {

int **sum** = val1 + val2;

ui->label->setText(QString("Result: %1 %2").arg(sum).arg(sum));

}

}

**Задание 2: Вариант 1** 1) Дан массив размера N. Вывести его элементы в обратном порядке. 2) Дано число k (0 < k < 11) и матрица размера m x n. Найти сумму и произведение элементов k-го столбца данной матрицы (нахождение суммы или произведения определяется пользователем в виджете comboBox).

**mainwindow.cpp**

#include "mainwindow.h"

#include "ui\_mainwindow.h"

#include "QRandomGenerator"

MainWindow::**MainWindow**(QWidget \***parent**)

: QMainWindow(*parent*)

, ui(*new* Ui::MainWindow)

{

ui->setupUi(*this*);

}

MainWindow::~**MainWindow**()

{

*delete* ui;

}

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

{

int **size** = 0;

size = ui->lineEdit->text().toInt();

QVector<int> **vect**;

QString **str**;

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

vect.push\_back(i + 1);

str += QString::number(vect[i]);

}

ui->label->setText(QString("Array: %1").arg(str));

std::reverse(vect.begin(), vect.end());

str = "";

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

str += QString::number(vect[i]);

}

ui->label\_2->setText(QString("Reverse Array: %1").arg(str));

}

void MainWindow::**on\_calc\_btn\_clicked**()

{

int **n**, **m**;

n = ui->rows\_input->text().toInt();

m = ui->cols\_input->text().toInt();

*if* (!n || !m) *return*;

*if* (n >= 11 || m >= 11) *return*;

QVector< QVector<int> > **matrix**(n, QVector<int> (m));

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

*for* (int **j** = 0; j < m; j++)

{

matrix[i][j] = QRandomGenerator::global()->bounded(10);

}

QString **strMatrix**;

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

{

*for* (int **j** = 0; j < m; j++)

strMatrix += QString::number(matrix[i][j]) + " ";

strMatrix += "\n";

}

ui->textEdit->setFontPointSize(20);

ui->textEdit->setText(strMatrix);

int **columnNumber** = ui->kcolumn\_input->text().toInt();

*if* (!columnNumber) *return*;

*if* (columnNumber > m) *return*;

int **actionIndex** = ui->comboBox->currentIndex();

int **result** = 0;

*if* (actionIndex == 0) {

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

result += matrix[i][columnNumber-1];

}

} *else* *if* (actionIndex == 1) {

result = 1;

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

result \*= matrix[i][columnNumber-1];

}

}

ui->result\_lbl->setText(QString("Result: %1").arg(result));

}

**Задание 3: Вариант 4** 1) Заполнить список случайными элементами и реализовать удаление элементов с позиций с N по K. 2) Заполнить связный список случайными элементами и отсортировать их по убыванию.

**mainwindow.cpp**

#include "mainwindow.h"

#include "ui\_mainwindow.h"

#include <QRandomGenerator>

MainWindow::MainWindow(QWidget \*parent)

: QMainWindow(parent)

, ui(*new* Ui::MainWindow)

{

ui->setupUi(*this*);

*for* (int i = 0; i < 10; i++) {

list1.append(QRandomGenerator::global()->bounded(10));

ui->task1\_textedit->append(QString::number(list1.at(i)));

}

*for* (int i = 0; i < 10; i++) {

list2.append(QRandomGenerator::global()->bounded(10));

ui->textEdit\_2->append(QString::number(list2.at(i)));

}

}

MainWindow::~MainWindow()

{

*delete* ui;

}

void MainWindow::on\_task\_1\_btn\_clicked()

{

int from = ui->from\_input->text().toInt();

int to = ui->to\_input->text().toInt();

*if* (from <= 0 || to > list1.size()) *return*;

list1.erase(list1.begin() + from - 1, list1.begin() + to);

ui->task1\_textedit->setText("");

*for* (int i = 0; i < list1.size(); i++) {

ui->task1\_textedit->append(QString::number(list1.at(i)));

}

}

void MainWindow::on\_sort\_btn\_clicked()

{

std::sort(list2.begin(), list2.end());

ui->textEdit\_2->setText("");

*for* (int i = 0; i < list2.size(); i++) {

ui->textEdit\_2->append(QString::number(list2.at(i)));

}

}

**Задание 4: Вариант 1** 1) Заполнить стек 10 случайными числами из интервала [-10; 20]. Просмотреть содержимое стека. Найти сумму положительных чисел, хранящихся в стеке. 2) Сформировать очередь из 8 чисел. Записать в очередь модуль разности между двумя соседними элементами очереди.

**mainwindow.cpp**

#include "mainwindow.h"

#include "ui\_mainwindow.h"

#include <QRandomGenerator>

MainWindow::MainWindow(QWidget \*parent)

: QMainWindow(parent)

, ui(*new* Ui::MainWindow)

{

ui->setupUi(*this*);

}

MainWindow::~MainWindow()

{

*delete* ui;

}

void MainWindow::on\_sum\_btn\_clicked()

{

ui->stack\_textedit->clear();

QStack<int> stack;

*for* (int i = 0; i < 10; i++) {

int value = QRandomGenerator::global()->bounded(-10, 20);

ui->stack\_textedit->append(QString::number(value));

stack.push(value);

}

int sum = 0;

*while* (!stack.isEmpty()) {

int value = stack.pop();

*if* (value > 0) sum += value;

}

ui->sum\_result\_lbl->setText(QString("Result %1").arg(sum));

}

void MainWindow::on\_calc\_btn\_clicked()

{

QQueue<int> que;

QString str;

ui->que\_textedit->clear();

str.clear();

*for* (int i = 0; i < 8; i++) {

int value = QRandomGenerator::global()->bounded(0, 10);

str += QString::number(value) + " ";

que.enqueue(value);

}

ui->que\_textedit->setText(str);

int position = QRandomGenerator::global()->bounded(1, que.size());

int el = que.at(position - 1);

int neighbour = que.at(position);

int sum = el + neighbour;

qDebug() << que;

que.insert(position, sum);

str.clear();

qDebug() << que;

*for* (int i = 0; i < 9; i++) {

int value = que.dequeue();

str += QString::number(value) + " ";

}

ui->que\_textedit->append(str);

ui->queresult\_lbl->setText(QString("Position: %1, %2 + %3, Result: %4").arg(position).arg(el).arg(neighbour).arg(sum));

}

**Задание 4: Вариант 1** Имеется список класса (все имена различны). Определить, есть ли в классе человек, который побывал в гостях у всех. (Для каждого ученика составить множество побывавших у него в гостях друзей, сам ученик в это множество не входит.)

**mainwindow.cpp**

#include "mainwindow.h"

#include "ui\_mainwindow.h"

QString listToString(*const* QList<QString> &list);

MainWindow::MainWindow(QWidget \*parent)

: QMainWindow(parent)

, ui(*new* Ui::MainWindow)

{

ui->setupUi(*this*);

classList.append("Ivanov I.P.");

classList.append("Sidorov G.V.");

classList.append("Petrenko A.P.");

classList.append("Shipov E.O.");

*for* (*auto* student : classList) {

ui->students\_comboBox->addItem(student);

}

QList<QString> guests0 = QList<QString>() << classList.at(1) << classList.at(2);

QList<QString> guests1 = QList<QString>() << classList.at(0);

QList<QString> guests2 = QList<QString>() << classList.at(0) << classList.at(1);

QList<QString> guests3 = QList<QString>() << classList.at(0) << classList.at(1) << classList.at(2);

map.insert(classList.at(0), guests0);

map.insert(classList.at(1), guests1);

map.insert(classList.at(2), guests2);

map.insert(classList.at(3), guests3);

*for* (*auto* key : map.keys()) {

QString geustsStr = listToString(map.value(key));

ui->students->append(QString("Студент %1 был в гостях у: %2").arg(key).arg(geustsStr));

}

ui->result->setText(QString("Студент, который был у всех в гостях: "));

bool flag = *false*;

*for* (*auto* key : map.keys()) {

*if* (map.value(key).length() == classList.length() - 1) {

flag = *true*;

ui->result->append(key);

}

}

*if* (!flag) {

ui->result->append(QString("таких студентов нтет."));

}

}

MainWindow::~MainWindow()

{

*delete* ui;

}

QString listToString(*const* QList<QString> &list) {

QString s = "";

*for* (*auto* &value : list) {

s += QString("%1,").arg(value);

}

s.chop(1);

*return* s;

}

void MainWindow::on\_add\_student\_btn\_clicked()

{

QString student = ui->students\_lineEdit->text();

*if* (student.length() <= 0) *return*;

ui->students\_comboBox->addItem(student);

ui->students\_lineEdit->clear();

}

void MainWindow::on\_pushButton\_clicked()

{

ui->students->clear();

*if* (ui->students\_comboBox->currentIndex() == 0) {

*for* (*auto* key : map.keys()) {

QString geustsStr = listToString(map.value(key));

ui->students->append(QString("Студент %1 был в гостях у: %2").arg(key).arg(geustsStr));

}

*return*;

}

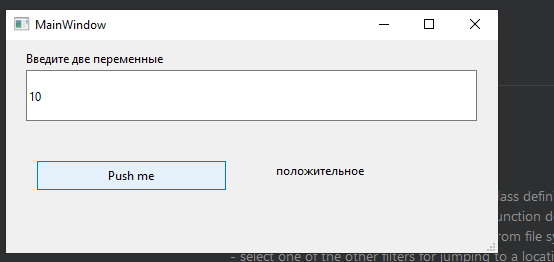
QString currentStudent = classList.at(ui->students\_comboBox->currentIndex());

QString geustsStr = listToString(map.value(currentStudent));

ui->students->append(QString("Студент %1 был в гостях у: %2").arg(currentStudent).arg(geustsStr));\

}

**Результат выполнения программы:**

****

**Вывод:** В ходе лабораторной работы изучил основы работы с контейнерами в Qt 5 , а также закрепил полученные знания на практике.