|  |  |
| --- | --- |
| 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 №** | 9 |

**Название:**

Программирование с использованием QT.

**Дисциплина:** Объектно-ориентированное программирование

Вариант 16.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ6-21Б |  |  | Т.Е.Старжевский |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | О.А.Веселовская |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2022

На рисунке 5 показан внешний вид простого приложения, в котором предлагается ввести возраст с использованием одного из трех вариантов ввода:

1) непосредственного ввода числа,

2) посредством стрелок (элемент типа QSpinBox), последовательно увеличиваю- щих или уменьшающих значение,

3) с помощью специального ползунка (слайдера – элемент типа QSlider).

Замените в программе схему выравнивания QHBoxLayout на QVBoxLayout и зафик- сируйте результат.

#ifndef MAINWINDOW\_H

#define **MAINWINDOW\_H**

#include <QMainWindow>

QT\_BEGIN\_NAMESPACE

namespace **Ui** { class **MainWindow**; }

QT\_END\_NAMESPACE

class **MainWindow** : public QMainWindow

{

Q\_OBJECT

public:

**MainWindow**(QWidget \***parent** = nullptr);

~**MainWindow**();

private slots:

void **on\_horizontalSlider\_valueChanged**(int **value**);

void **on\_spinBox\_valueChanged**(int **arg1**);

private:

Ui::MainWindow \***ui**;

};

#endif // MAINWINDOW\_H

#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\_horizontalSlider\_valueChanged**(int **value**)

{

ui->horizontalSlider->setValue(value);

ui->spinBox->setValue(value);

ui->lineEdit->setText(QString::number(value));

}

void MainWindow::**on\_spinBox\_valueChanged**(int **arg1**)

{

ui->horizontalSlider->setValue(arg1);

ui->lineEdit->setText(QString::number(arg1));

}

#include "mainwindow.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** = "spin\_1\_" + QLocale(locale).name();

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

a.installTranslator(&*translator*);

break;

}

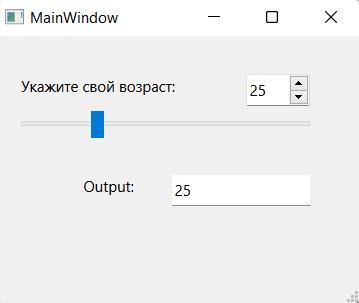
}

MainWindow **w**;

w.show();

return a.exec();

}



Измените тип разделителя с QSplitter(Qt::Horizontal); на QSplitter(Qt::Vertical); и зафиксируйте полученный результат.

#ifndef MAINWINDOW\_H

#define **MAINWINDOW\_H**

#include <QMainWindow>

QT\_BEGIN\_NAMESPACE

namespace **Ui** { class **MainWindow**; }

QT\_END\_NAMESPACE

class **MainWindow** : public QMainWindow

{

Q\_OBJECT

public:

**MainWindow**(QWidget \***parent** = nullptr);

~**MainWindow**();

private slots:

void **on\_pushButton\_clicked**();

void **on\_dial\_valueChanged**(int **value**);

void **on\_spinBox\_valueChanged**(int **arg1**);

void **on\_pushButton\_2\_clicked**();

void **on\_dial\_2\_valueChanged**(int **value**);

void **on\_spinBox\_2\_valueChanged**(int **arg1**);

private:

Ui::MainWindow \***ui**;

};

#endif // MAINWINDOW\_H

#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**()

{

this->close();

}

void MainWindow::**on\_dial\_valueChanged**(int **value**)

{

ui->spinBox->setValue(value);

ui->textEdit->append(QString::number(value));

}

void MainWindow::**on\_spinBox\_valueChanged**(int **arg1**)

{

ui->dial->setValue(arg1);

}

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

{

close();

}

void MainWindow::**on\_dial\_2\_valueChanged**(int **value**)

{

ui->spinBox\_2->setValue(value);

ui->textEdit\_2->append(QString::number(value));

}

void MainWindow::**on\_spinBox\_2\_valueChanged**(int **arg1**)

{

ui->dial\_2->setValue(arg1);

}

#include "mainwindow.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** = "square\_2\_" + QLocale(locale).name();

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

a.installTranslator(&*translator*);

break;

}

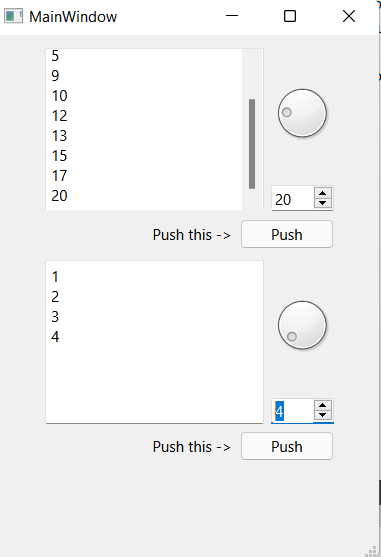
}

MainWindow **w**;

w.show();

return a.exec();

}



Добавьте кнопки, выполняющие: бинарные операции x^y , lg (по аналогии с операциями +,-,/,\*), а также унарную x! и константы π и e (по аналогии с операцией -/+) и разместите этот ряд кнопок горизонтально, сверху от цифровых кнопок.

#ifndef MAINWINDOW\_H

#define **MAINWINDOW\_H**

#include "qpushbutton.h"

#include <QMainWindow>

QT\_BEGIN\_NAMESPACE

namespace **Ui** { class **MainWindow**; }

QT\_END\_NAMESPACE

class **MainWindow** : public QMainWindow

{

Q\_OBJECT

public:

**MainWindow**(QWidget \***parent** = nullptr);

~**MainWindow**();

private:

Ui::MainWindow \***ui**;

bool **is\_result**; //Если результат был, то сохраняем второе число меняя первое пока не нажата новая операция

bool **is\_ClearAfterFirst**; //Использую для очистки графы ответа для ввода второго числа после нажатия на любую операцию

bool **need\_to\_clear**;

double **first\_num**; //Запоминаю первое число

double **second\_num**; //Запоминаю второе число

//Прикол в том, что нужно запомнить второе и отнимать его каждый раз, пока не нажал новую операцию

QPushButton \***current\_button**;

private slots:

void **number\_pressed**();

void **on\_pushButton\_dot\_clicked**();

void **on\_pushButton\_erase\_clicked**();

void **on\_pushButton\_eLast\_clicked**();

void **on\_pushButton\_pi\_clicked**();

void **on\_pushButton\_e\_clicked**();

void **on\_PushButton\_swap\_clicked**();

void **on\_operations**();

void **on\_pushButton\_anwser\_clicked**();

//void clearing();

void **on\_pushButton\_log\_clicked**();

void **on\_pushButton\_fact\_clicked**();

};

int **factorial** (int **num**);

#endif // MAINWINDOW\_H

#include "mainwindow.h"

#include "ui\_mainwindow.h"

#include <QtMath>

const int **n**=26;

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

: QMainWindow(*parent*)

, ui(new Ui::MainWindow)

{

ui->setupUi(this);

connect(ui->pushButton\_0,SIGNAL(clicked()),this,SLOT(number\_pressed()));

connect(ui->pushButton\_1,SIGNAL(clicked()),this,SLOT(number\_pressed()));

connect(ui->pushButton\_2,SIGNAL(clicked()),this,SLOT(number\_pressed()));

connect(ui->pushButton\_3,SIGNAL(clicked()),this,SLOT(number\_pressed()));

connect(ui->pushButton\_4,SIGNAL(clicked()),this,SLOT(number\_pressed()));

connect(ui->pushButton\_5,SIGNAL(clicked()),this,SLOT(number\_pressed()));

connect(ui->pushButton\_6,SIGNAL(clicked()),this,SLOT(number\_pressed()));

connect(ui->pushButton\_7,SIGNAL(clicked()),this,SLOT(number\_pressed()));

connect(ui->pushButton\_8,SIGNAL(clicked()),this,SLOT(number\_pressed()));

connect(ui->pushButton\_9,SIGNAL(clicked()),this,SLOT(number\_pressed()));

connect(ui->pushButton\_repeat,SIGNAL(clicked()),this,SLOT(on\_operations()));

connect(ui->pushButton\_del,SIGNAL(clicked()),this,SLOT(on\_operations()));

connect(ui->pushButton\_minus,SIGNAL(clicked()),this,SLOT(on\_operations()));

connect(ui->pushButton\_plus,SIGNAL(clicked()),this,SLOT(on\_operations()));

connect(ui->pushButton\_power,SIGNAL(clicked()),this,SLOT(on\_operations()));

ui->pushButton\_plus->setCheckable(true);

ui->pushButton\_minus->setCheckable(true);

ui->pushButton\_del->setCheckable(true);

ui->pushButton\_repeat->setCheckable(true);

ui->pushButton\_log->setCheckable(true);

ui->pushButton\_fact->setCheckable(true);

ui->pushButton\_power->setCheckable(true);

is\_result = false;

is\_ClearAfterFirst = false;

need\_to\_clear = false;

ui->Answer\_help->setText("");

current\_button = nullptr;

}

MainWindow::~**MainWindow**()

{

delete ui;

}

void MainWindow::**number\_pressed**()

{

QPushButton \***button** = (QPushButton\*) sender();

QString **Answer** = ui->Answer->text();

if (is\_result) {

ui->Answer->setText(button->text());

is\_result = false;

} else if (is\_ClearAfterFirst) {

ui->Answer->setText(button->text());

is\_ClearAfterFirst = false;

} else if (Answer.length()<n) {

double **num** = (Answer + button->text()).toDouble();

QString **last**;

if (Answer.contains("."))

last = Answer + button->text();

else

last = QString::number(num,'g',15);

ui->Answer->setText(last);

}

}

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

{

if(!ui->Answer->text().contains("."))

ui->Answer->setText(ui->Answer->text()+".");

}

/\*void MainWindow::clearing()

{

if (need\_to\_clear)

{

ui->Answer->setText("0");

ui->Answer\_help->setText("");

is\_result = false;

is\_ClearAfterFirst = false;

current\_button->setChecked(false);

current\_button = nullptr;

first\_num = 0;

second\_num = 0;

need\_to\_clear = false;

}

} \*/

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

{

ui->Answer->setText("0");

ui->Answer\_help->setText("");

is\_result = false;

is\_ClearAfterFirst = false;

if (current\_button != nullptr)

current\_button->setChecked(false);

first\_num = 0;

second\_num = 0;

}

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

{

QString **was** = ui->Answer->text(), **last**;

for (int **i** = 0; i<was.length()-1;i++)

{

last += was[i];

}

//проверка если ласт длина равна 0 то ласт равен 0

if (last.length()==0) last = "0";

ui->Answer->setText(last);

first\_num = last.toDouble();

}

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

{

ui->Answer->setText("3.1415926535897932384");

}

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

{

ui->Answer->setText("2.7182818284590452353");

}

void MainWindow::**on\_PushButton\_swap\_clicked**()

{

QString **answer** = ui->Answer->text(),**last**;

if (answer[0] == '-')

for (int **i**=1;i<answer.length();i++)

last += answer[i];

else last = "-" + answer;

if (last.length()==0) last = "0";

ui->Answer->setText(last);

first\_num = last.toDouble();

}

void MainWindow::**on\_operations**()

{

if (current\_button!=nullptr)

current\_button->setChecked(false);

current\_button = (QPushButton \*) sender();

first\_num = ui->Answer->text().toDouble();

QString **s3** =current\_button->text();

if (s3 == "xʸ") s3 = "^";

ui->Answer\_help->setText(ui->Answer->text() + s3);

current\_button->setChecked(true);

is\_ClearAfterFirst = true;

is\_result = false;

}

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

{

bool **is\_error** = false;

double **result** = 0;

if (!is\_result)

second\_num = ui->Answer->text().toDouble(); //Не меняем второе число, пока не будет нажата новая операция

if (ui->pushButton\_plus->isChecked()) {

result = first\_num + second\_num;

} else if (ui->pushButton\_minus->isChecked()) {

result = first\_num - second\_num;

} else if (ui->pushButton\_repeat->isChecked()) {

result = first\_num \* second\_num;

} else if (ui->pushButton\_del->isChecked()) {

if (second\_num != 0) result = first\_num / second\_num;

else { result = 0; is\_error = true; }

} else if (ui->pushButton\_power->isChecked()) {

result = qPow(first\_num,second\_num);

}

if (is\_error) {

ui->Answer\_help->setText("Invalid parametres");

}

else if (current\_button!=nullptr){

QString **s1** = QString::number(first\_num,'g',n), **s2** = QString::number(second\_num,'g',n), **s3** = current\_button->text();

if (current\_button->text() == "xʸ") s3 = "^";

ui->Answer\_help->setText(s1 + s3 + s2 + " = ");

ui->Answer->setText(QString::number(result,'g', n-1));

}

first\_num = result;

is\_result = true;

}

int **factorial** (int **num**) {

if (num ==0 ) return num;

else if (num == 1) return num;

else return num \* factorial(num-1);

}

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

{

first\_num = ui->Answer->text().toDouble();

if (first\_num <=0 ) ui->Answer\_help->setText("Invalid parametres");

else {

ui->Answer\_help->setText("lg"+ QString::number(first\_num,'g',n-1) + " = ");

double **answer**;

answer = qLn(first\_num)/qLn(10);

ui->Answer->setText(QString::number(answer,'g',n-1));

}

}

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

{

first\_num = qFloor(ui->Answer->text().toDouble());

if (first\_num <0 ) ui->Answer\_help->setText("Invalid parametres");

else if (first\_num > 31)

ui->Answer\_help->setText("Overflow");

else {

ui->Answer\_help->setText(QString::number(first\_num,'g',n-1) + "! = ");

ui->Answer->setText(QString::number(factorial(first\_num),'g',n-1));

}

}

#include "mainwindow.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 = "Calculator\_" + QLocale(locale).name();

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

a.installTranslator(&translator);

break;

}

}

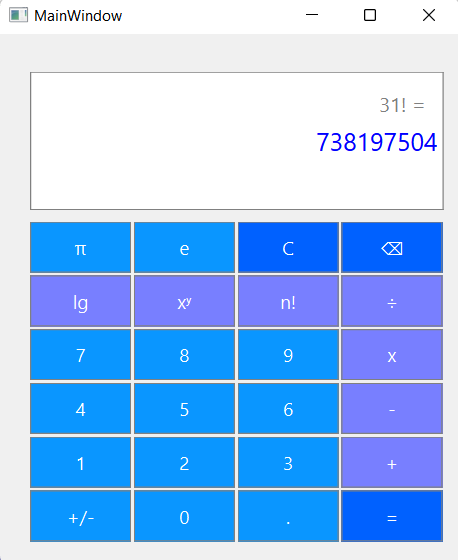
MainWindow w;

w.show();

return a.exec();

}





Разработать приложение, имеющее строку ввода данных, кнопку запуска преобра- зования и текстовое поле, предназначенное только для отображения информации. При этом не использовать QtDesigner! Любой текст строки ввода должен отображаться в тек- стовом поле сразу после завершения ввода. В начале строки должна быть вставлена по- метка «input:». При нажатии кнопки преобразования строка ввода должна быть преобра- зована либо в верхний регистр, либо в нижний противоположно тому, что производилось при предыдущем нажатии кнопки.

#ifndef MAINWINDOW\_H

#define MAINWINDOW\_H

#include <QMainWindow>

QT\_BEGIN\_NAMESPACE

namespace Ui { class MainWindow; }

QT\_END\_NAMESPACE

class MainWindow : public QMainWindow

{

Q\_OBJECT

bool key;

public:

MainWindow(QWidget \*parent = nullptr);

~MainWindow();

private slots:

void on\_pushButton\_clicked();

private:

Ui::MainWindow \*ui;

};

#endif // MAINWINDOW\_H

#include "mainwindow.h"

#include "ui\_mainwindow.h"

MainWindow::MainWindow(QWidget \*parent)

: QMainWindow(parent)

, ui(new Ui::MainWindow)

{

ui->setupUi(this);

key = true;

}

MainWindow::~MainWindow()

{

delete ui;

}

void MainWindow::on\_pushButton\_clicked()

{

ui->textEdit\_2->append("Input: " + ui->textEdit->toPlainText());

if (key)

ui->textEdit\_2->append("All lower: " + ui->textEdit->toPlainText().toLower());

else ui->textEdit\_2->append("All up: " + ui->textEdit->toPlainText().toUpper());

key = !key;

}

#include "mainwindow.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 = "last\_" + QLocale(locale).name();

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

a.installTranslator(&translator);

break;

}

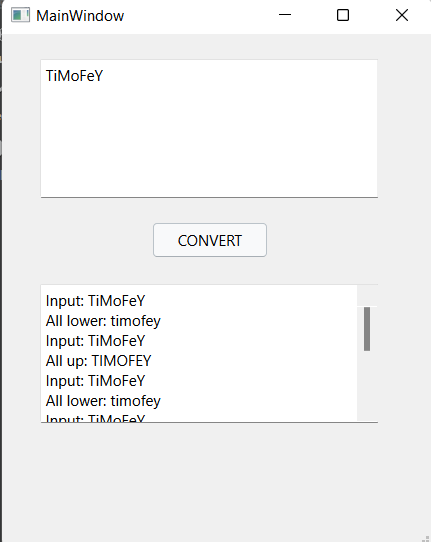
}

MainWindow w;

w.show();

return a.exec();

}



**Вывод:** Научился программировать с использованием QT.