## **2. Введение**

**Цель работы**:  
 Создать графическое приложение на **Qt**, в котором будут выполняться операции над строками в двух полях ввода (LineEdit):

* Переворачивание строки (reverse),
* Обмен содержимым (swap),
* Копирование строки (copy),
* Переворачивание обеих строк (reverseAll).

**Задачи**:

1. Изучить способы создания GUI-приложения при помощи библиотеки **Qt**.
2. Разместить на форме два компонента QLineEdit.
3. Реализовать операции reverse, swap, copy, reverseAll без использования контейнеров **STL**.
4. Использовать базовые методы и возможности класса QString.

**Краткая характеристика Qt**:  
 Библиотека Qt — это кроссплатформенный фреймворк для разработки приложений с графическим интерфейсом. Предоставляет широкий набор классов и инструментов для работы с интерфейсом (виджеты, диалоговые окна), сетевыми соединениями, файлами, потоками и многим другим.

## **3. Постановка задачи**

**Описание задания**:

* На форме должны быть два поля ввода (QLineEdit), расположенные горизонтально.
* Выполняются операции:
  1. reverse(strEdtLeft) – переворот (реверс) строки в левом поле;
  2. reverse(strEdtRight) – переворот (реверс) строки в правом поле;
  3. reverseAll() – переворот обеих строк одновременно;
  4. swap(strEdtLeft, strEdtRight) – обмен содержимым левого и правого поля;
  5. copy(strEdtLeft, strEdtRight) – копирование строки из левого поля в правое;
  6. copy(strEdtRight, strEdtLeft) – копирование строки из правого поля в левое.

**Условия**:

* **Не используются** контейнеры **STL** (никаких std::vector, std::deque, std::list и т.д.).
* Разрешено использовать только базовые методы класса QString, а также ручные циклы for, while и т.п.
* Среда разработки (по умолчанию): Qt Creator.

## **4. Проектирование приложения**

### **4.1. Выбор инструментов**

* **Qt Creator**: удобная среда для разработки GUI-приложений на языке C++ с использованием фреймворка Qt.
* **QMainWindow**: выбран в качестве главного окна (MainWindow), поскольку он позволяет легко создавать меню, панели инструментов и т.д.

### **4.2. Описание интерфейса пользователя**

1. Два поля ввода (объекты QLineEdit): lineEditLeft и lineEditRight.
2. Набор кнопок или элементов меню для запуска операций:
   * Reverse Left, Reverse Right, Reverse All, Swap, Copy L->R, Copy R->L.
3. Расположение элементов:
   * Поля ввода выровнены горизонтально.
   * Кнопки можно расположить вертикально или горизонтально.

### **4.3. Схема интерфейса**

Примерная схема:

┌───────────────────────────────────────────────────┐

│ MainWindow │

│ │

│ ┌─────────────────┐ ┌─────────────────┐ │

│ │ lineEditLeft │ │ lineEditRight │ │

│ └─────────────────┘ └─────────────────┘ │

│ │

│ ┌───────────────────────────────────────────────┐ │

│ │ Reverse Left | Reverse Right | Reverse All │ │

│ └───────────────────────────────────────────────┘ │

│ ┌───────────────────────────────────────────────┐ │

│ │ Swap | Copy L->R | Copy R->L │ │

│ └───────────────────────────────────────────────┘ │

│ │

└───────────────────────────────────────────────────┘

## **5. Реализация приложения**

### **5.1. Создание проекта и структура файлов**

Для сборки без открытия проекта в Qt Creator используется файл **.pro** (пример имени: lab06\_07.pro).

**Структура**:

lab06\_07/

├── lab06\_07.pro

├── main.cpp

├── mainwindow.h

└── mainwindow.cpp

### **5.2. Файл lab06\_07.pro**

#-------------------------------------------------

# Проект для сборки при помощи qmake

# qmake

# make

# ./lab06\_07

#-------------------------------------------------

QT += core gui

greaterThan(QT\_MAJOR\_VERSION, 4): QT += widgets

TARGET = lab06\_07

TEMPLATE = app

SOURCES += \

main.cpp \

mainwindow.cpp

HEADERS += \

mainwindow.h

**Ключевые моменты**:

* Подключаем модули core, gui и при необходимости widgets (если Qt5+).
* Указываем исходные файлы (SOURCES) и заголовочные (HEADERS).

### **5.3. Файл main.cpp**

#include <QApplication>

#include "mainwindow.h"

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

{

QApplication a(argc, argv);

MainWindow w;

w.show();

return a.exec();

}

**Описание**:

* Точка входа в Qt-приложение. Создаётся объект QApplication, затем главное окно (MainWindow), которое отображается на экране.

### **5.4. Файл mainwindow.h**

#ifndef MAINWINDOW\_H

#define MAINWINDOW\_H

#include <QMainWindow>

class QLineEdit;

class QPushButton;

class MainWindow : public QMainWindow

{

Q\_OBJECT

public:

explicit MainWindow(QWidget \*parent = nullptr);

~MainWindow();

private slots:

void onReverseLeft();

void onReverseRight();

void onReverseAll();

void onSwap();

void onCopyLeftToRight();

void onCopyRightToLeft();

private:

// Метод для переворота строки без использования STL

QString reverseString(const QString &input);

private:

// Поля для UI

QLineEdit\* lineEditLeft;

QLineEdit\* lineEditRight;

// Кнопки

QPushButton\* btnReverseLeft;

QPushButton\* btnReverseRight;

QPushButton\* btnReverseAll;

QPushButton\* btnSwap;

QPushButton\* btnCopyLtoR;

QPushButton\* btnCopyRtoL;

};

#endif // MAINWINDOW\_H

**Описание**:

* Объявляются два поля ввода и шесть кнопок.
* Объявляются слоты для каждой операции (reverse, swap, copy, reverseAll).
* Метод reverseString используется для ручного разворота строки (без STL).

### **5.5. Файл mainwindow.cpp**

#include "mainwindow.h"

#include <QLineEdit>

#include <QPushButton>

#include <QHBoxLayout>

#include <QVBoxLayout>

#include <QWidget>

MainWindow::MainWindow(QWidget \*parent)

: QMainWindow(parent)

, lineEditLeft(nullptr)

, lineEditRight(nullptr)

, btnReverseLeft(nullptr)

, btnReverseRight(nullptr)

, btnReverseAll(nullptr)

, btnSwap(nullptr)

, btnCopyLtoR(nullptr)

, btnCopyRtoL(nullptr)

{

// Центральный виджет для QMainWindow

QWidget\* central = new QWidget(this);

setCentralWidget(central);

// Создаём два QLineEdit

lineEditLeft = new QLineEdit(this);

lineEditRight = new QLineEdit(this);

// Создаём кнопки

btnReverseLeft = new QPushButton("Reverse Left", this);

btnReverseRight = new QPushButton("Reverse Right", this);

btnReverseAll = new QPushButton("Reverse All", this);

btnSwap = new QPushButton("Swap", this);

btnCopyLtoR = new QPushButton("Copy L->R", this);

btnCopyRtoL = new QPushButton("Copy R->L", this);

// Подключаем слоты

connect(btnReverseLeft, &QPushButton::clicked, this, &MainWindow::onReverseLeft);

connect(btnReverseRight, &QPushButton::clicked, this, &MainWindow::onReverseRight);

connect(btnReverseAll, &QPushButton::clicked, this, &MainWindow::onReverseAll);

connect(btnSwap, &QPushButton::clicked, this, &MainWindow::onSwap);

connect(btnCopyLtoR, &QPushButton::clicked, this, &MainWindow::onCopyLeftToRight);

connect(btnCopyRtoL, &QPushButton::clicked, this, &MainWindow::onCopyRightToLeft);

// Макет для двух QLineEdit (горизонтальная компоновка)

QHBoxLayout\* lineEditsLayout = new QHBoxLayout;

lineEditsLayout->addWidget(lineEditLeft);

lineEditsLayout->addWidget(lineEditRight);

// Макет для кнопок (вертикальная компоновка)

QVBoxLayout\* buttonsLayout = new QVBoxLayout;

buttonsLayout->addWidget(btnReverseLeft);

buttonsLayout->addWidget(btnReverseRight);

buttonsLayout->addWidget(btnReverseAll);

buttonsLayout->addWidget(btnSwap);

buttonsLayout->addWidget(btnCopyLtoR);

buttonsLayout->addWidget(btnCopyRtoL);

// Общий вертикальный макет

QVBoxLayout\* mainLayout = new QVBoxLayout(central);

mainLayout->addLayout(lineEditsLayout);

mainLayout->addLayout(buttonsLayout);

// Размеры окна

resize(400, 200);

setWindowTitle("Lab 6, Task 7 (No STL)");

}

MainWindow::~MainWindow()

{

// При наличии родителя (this) объекты будут удалены Qt автоматически

}

// --- Методы-обработчики ---

void MainWindow::onReverseLeft()

{

QString text = lineEditLeft->text();

lineEditLeft->setText(reverseString(text));

}

void MainWindow::onReverseRight()

{

QString text = lineEditRight->text();

lineEditRight->setText(reverseString(text));

}

void MainWindow::onReverseAll()

{

QString leftText = lineEditLeft->text();

QString rightText = lineEditRight->text();

lineEditLeft->setText(reverseString(leftText));

lineEditRight->setText(reverseString(rightText));

}

void MainWindow::onSwap()

{

QString temp = lineEditLeft->text();

lineEditLeft->setText(lineEditRight->text());

lineEditRight->setText(temp);

}

void MainWindow::onCopyLeftToRight()

{

lineEditRight->setText(lineEditLeft->text());

}

void MainWindow::onCopyRightToLeft()

{

lineEditLeft->setText(lineEditRight->text());

}

// --- Вспомогательная функция переворота строки ---

QString MainWindow::reverseString(const QString &input)

{

// Ручной разворот строки БЕЗ использования std::reverse или контейнеров STL

QString result;

result.reserve(input.length()); // Оптимизация: сразу резервируем место

for (int i = input.length() - 1; i >= 0; --i) {

result.append(input.at(i));

}

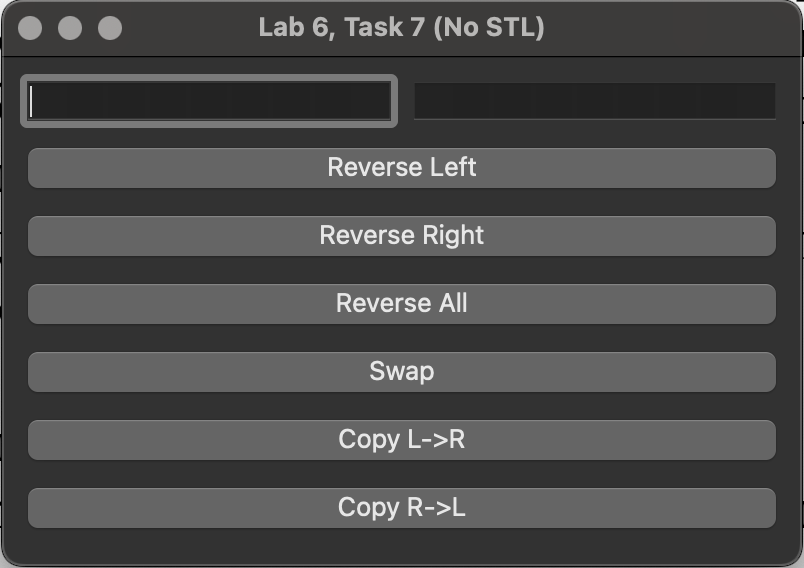
return result;

}

**Основные моменты**:

* Каждая операция реализована в собственном слоте.
* Переворот строки (reverseString) происходит вручную, циклом for, проходящим от конца строки к началу.
* **Контейнеры STL не применяются**. Используем только QString, методы append, at, text и т.д.

### **5.6. Скриншоты программы**

**

## **6. Тестирование приложения**

### **6.1. Описание тестов**

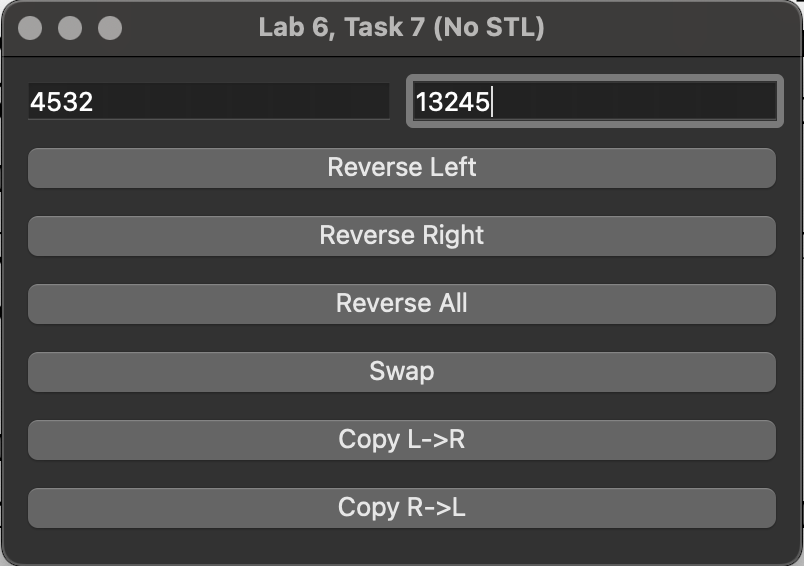
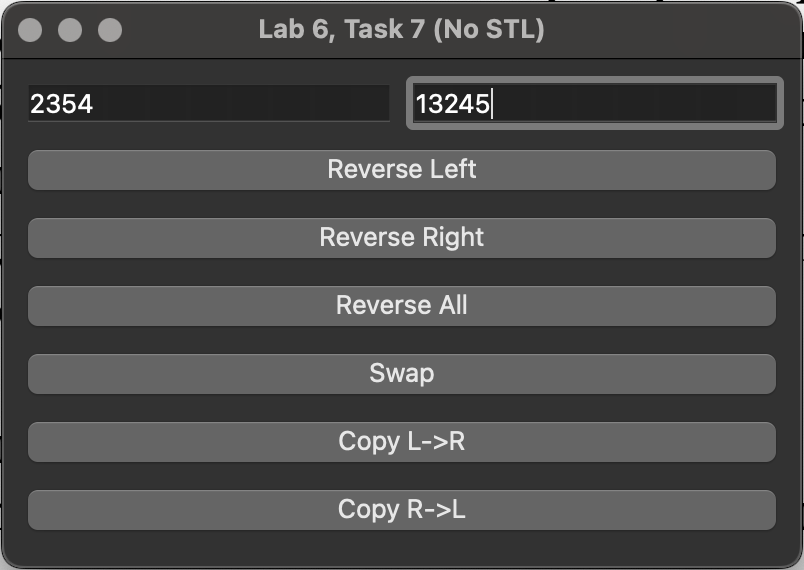
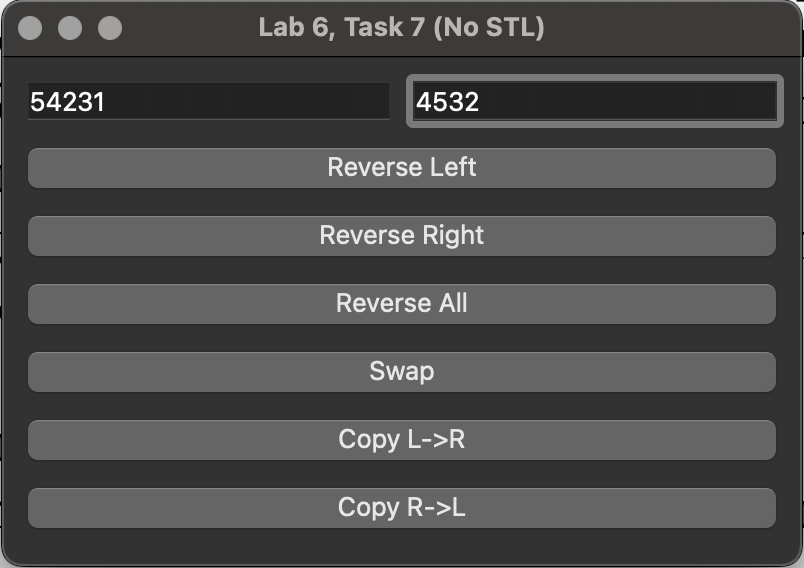
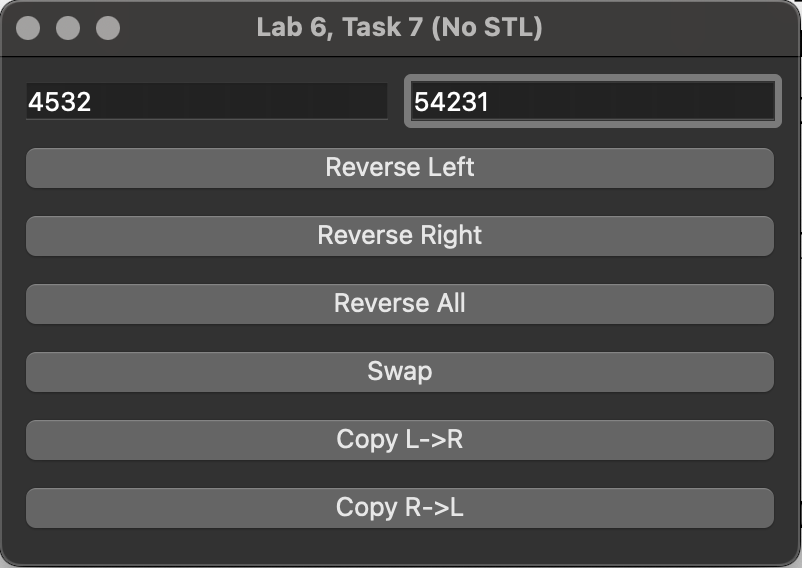
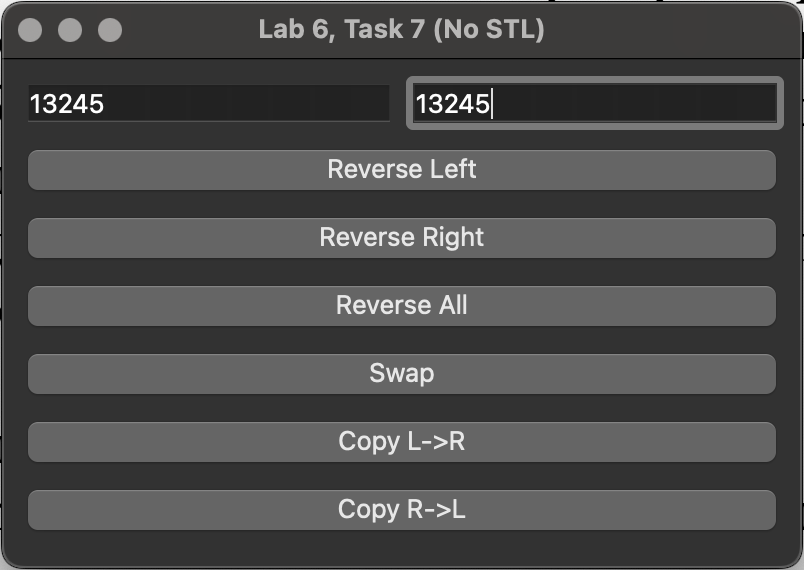
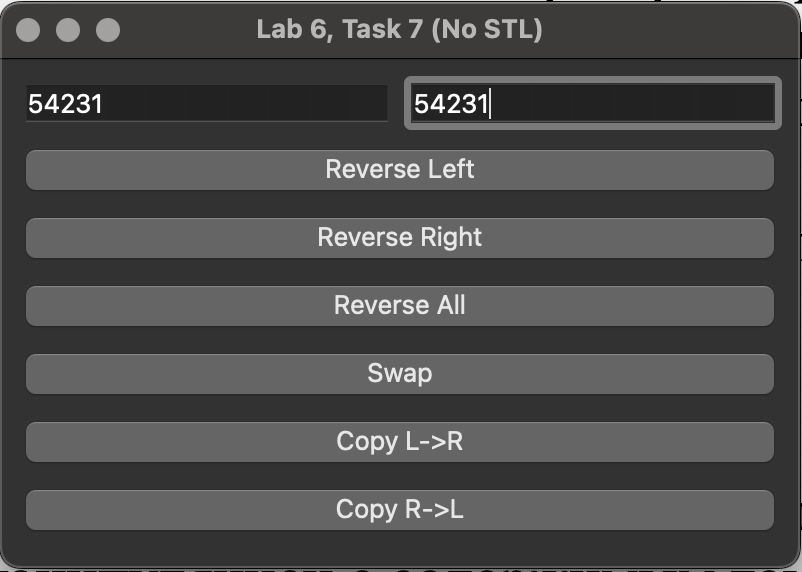
Для проверки корректности работы приложения вводились различные комбинации строк:

1. **Пустая строка**
   * Левое поле: "" (пусто), правое поле: "" (пусто).
   * Проверяются операции: reverse, swap, copy.
   * Ожидается, что строка после реверса останется пустой, обмен и копирование тоже ничего не изменяют.
2. **Короткие строки**
   * Левое поле: "A", правое поле: "BC".
   * Проверяются все операции:
     + reverseLeft -> "A" (остается "A").
     + reverseRight -> "CB".
     + swap -> местами меняются содержимым.
     + copy -> проверяется копирование из левого в правое и наоборот.
3. **Слово**
   * Левое поле: "Hello", правое поле: "Qt".
   * Проверяем, что при реверсе "Hello" превращается в "olleH", а "Qt" -> "tQ".
   * Операции swap и copy работают штатно.
4. **Включение пробелов и спецсимволов**
   * Левое поле: "Test 123", правое поле: "!".
   * Проверяем поведение при реверсе (пробелы и цифры тоже меняются местами), обмен, копирование.

### **6.2. Таблица с тестовыми примерами**

| **№** | **Входные данные (Left / Right)** | **Операция** | **Ожидаемый результат** | **Фактический результат** | **Статус** |
| --- | --- | --- | --- | --- | --- |
| 1 | "" / "" | Reverse Left | "" / "" | "" / "" | OK |
| 2 | "A" / "BC" | Reverse Right | "A" / "CB" | "A" / "CB" | OK |
| 3 | "A" / "BC" | Swap | "BC" / "A" | "BC" / "A" | OK |
| 4 | "A" / "BC" | Copy L->R | "A" / "A" | "A" / "A" | OK |
| 5 | "Hello" / "Qt" | Reverse All | "olleH" / "tQ" | "olleH" / "tQ" | OK |
| 6 | "Test 123" / "!" | Reverse Left | "321 tseT" / "!" | "321 tseT" / "!" | OK |

### **6.3. Скриншоты с примерами работы**

*  
  
*

## **7. Выводы**

1. **Реализованы** все требуемые операции: reverse, swap, copy, reverseAll.
2. **GUI-приложение** успешно создано с использованием **Qt** (класс QMainWindow).
3. **STL-контейнеры не использовались** при работе со строками. Все операции над строками (QString) производились вручную или с помощью простых методов (append, at, text).
4. Цель лабораторной работы достигнута: разработано и протестировано приложение, позволяющее оперировать строками в двух полях ввода.
5. **Возможные улучшения**:
   * Добавить меню или панель инструментов вместо (или вместе) с кнопками.
   * Реализовать дополнительные функции, например, изменение регистра, удаление пробелов, проверку палиндрома и т.п.

## **8. Приложение**

Ниже приведён полный исходный код программы для удобства (файлы: .pro, main.cpp, mainwindow.h, mainwindow.cpp).

### **8.1. Файл lab06\_07.pro**

#-------------------------------------------------

# Проект для сборки при помощи qmake

# qmake

# make

# ./lab06\_07

#-------------------------------------------------

QT += core gui

greaterThan(QT\_MAJOR\_VERSION, 4): QT += widgets

TARGET = lab06\_07

TEMPLATE = app

SOURCES += \

main.cpp \

mainwindow.cpp

HEADERS += \

mainwindow.h

### **8.2. Файл main.cpp**

#include <QApplication>

#include "mainwindow.h"

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

{

QApplication a(argc, argv);

MainWindow w;

w.show();

return a.exec();

}

### **8.3. Файл mainwindow.h**

#ifndef MAINWINDOW\_H

#define MAINWINDOW\_H

#include <QMainWindow>

class QLineEdit;

class QPushButton;

class MainWindow : public QMainWindow

{

Q\_OBJECT

public:

explicit MainWindow(QWidget \*parent = nullptr);

~MainWindow();

private slots:

void onReverseLeft();

void onReverseRight();

void onReverseAll();

void onSwap();

void onCopyLeftToRight();

void onCopyRightToLeft();

private:

QString reverseString(const QString &input);

private:

QLineEdit\* lineEditLeft;

QLineEdit\* lineEditRight;

QPushButton\* btnReverseLeft;

QPushButton\* btnReverseRight;

QPushButton\* btnReverseAll;

QPushButton\* btnSwap;

QPushButton\* btnCopyLtoR;

QPushButton\* btnCopyRtoL;

};

#endif // MAINWINDOW\_H

### **8.4. Файл mainwindow.cpp**

#include "mainwindow.h"

#include <QLineEdit>

#include <QPushButton>

#include <QHBoxLayout>

#include <QVBoxLayout>

#include <QWidget>

MainWindow::MainWindow(QWidget \*parent)

: QMainWindow(parent)

, lineEditLeft(nullptr)

, lineEditRight(nullptr)

, btnReverseLeft(nullptr)

, btnReverseRight(nullptr)

, btnReverseAll(nullptr)

, btnSwap(nullptr)

, btnCopyLtoR(nullptr)

, btnCopyRtoL(nullptr)

{

QWidget\* central = new QWidget(this);

setCentralWidget(central);

lineEditLeft = new QLineEdit(this);

lineEditRight = new QLineEdit(this);

btnReverseLeft = new QPushButton("Reverse Left", this);

btnReverseRight = new QPushButton("Reverse Right", this);

btnReverseAll = new QPushButton("Reverse All", this);

btnSwap = new QPushButton("Swap", this);

btnCopyLtoR = new QPushButton("Copy L->R", this);

btnCopyRtoL = new QPushButton("Copy R->L", this);

connect(btnReverseLeft, &QPushButton::clicked, this, &MainWindow::onReverseLeft);

connect(btnReverseRight, &QPushButton::clicked, this, &MainWindow::onReverseRight);

connect(btnReverseAll, &QPushButton::clicked, this, &MainWindow::onReverseAll);

connect(btnSwap, &QPushButton::clicked, this, &MainWindow::onSwap);

connect(btnCopyLtoR, &QPushButton::clicked, this, &MainWindow::onCopyLeftToRight);

connect(btnCopyRtoL, &QPushButton::clicked, this, &MainWindow::onCopyRightToLeft);

QHBoxLayout\* lineEditsLayout = new QHBoxLayout;

lineEditsLayout->addWidget(lineEditLeft);

lineEditsLayout->addWidget(lineEditRight);

QVBoxLayout\* buttonsLayout = new QVBoxLayout;

buttonsLayout->addWidget(btnReverseLeft);

buttonsLayout->addWidget(btnReverseRight);

buttonsLayout->addWidget(btnReverseAll);

buttonsLayout->addWidget(btnSwap);

buttonsLayout->addWidget(btnCopyLtoR);

buttonsLayout->addWidget(btnCopyRtoL);

QVBoxLayout\* mainLayout = new QVBoxLayout(central);

mainLayout->addLayout(lineEditsLayout);

mainLayout->addLayout(buttonsLayout);

resize(400, 200);

setWindowTitle("Lab 6, Task 7 (No STL)");

}

MainWindow::~MainWindow()

{

}

void MainWindow::onReverseLeft()

{

QString text = lineEditLeft->text();

lineEditLeft->setText(reverseString(text));

}

void MainWindow::onReverseRight()

{

QString text = lineEditRight->text();

lineEditRight->setText(reverseString(text));

}

void MainWindow::onReverseAll()

{

QString leftText = lineEditLeft->text();

QString rightText = lineEditRight->text();

lineEditLeft->setText(reverseString(leftText));

lineEditRight->setText(reverseString(rightText));

}

void MainWindow::onSwap()

{

QString temp = lineEditLeft->text();

lineEditLeft->setText(lineEditRight->text());

lineEditRight->setText(temp);

}

void MainWindow::onCopyLeftToRight()

{

lineEditRight->setText(lineEditLeft->text());

}

void MainWindow::onCopyRightToLeft()

{

lineEditLeft->setText(lineEditRight->text());

}

QString MainWindow::reverseString(const QString &input)

{

QString result;

result.reserve(input.length());

for (int i = input.length() - 1; i >= 0; --i) {

result.append(input.at(i));

}

return result;

}

**Таким образом,** в отчёте показано, как реализовать лабораторную работу №6 (задача 7) с помощью Qt. Приложение соответствует требованиям:

* Имеет два QLineEdit.
* Поддерживает операции reverse, swap, copy, reverseAll.
* **STL-контейнеры не используются** — все операции над строками выполняются с помощью методов QString и ручных циклов.