

федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«Алтайский государственный технический университет им. И.И. Ползунова»

Факультет (институт) ФИТ

Кафедра прикладной математики

Отчет защищен с оценкой

Зав.кафедрой \_\_\_\_\_

\_\_\_\_\_ Боровцов Е.Г  
(подпись) (Фамилия И.О.)

## Отчет по практике

Вид	Производственная практика
-----	---------------------------

Код и наименование направления подготовки (специальности): \_\_\_\_\_

09.03.04 Программная инженерия

Направленность (профиль, специализация): \_\_\_\_\_

Разработка программно-информационных систем

Форма обучения: очная

Студента Замятин Иван Павлович

(Фамилия Имя Отчество)

Группа ПИ-02

г. Барнаул

ФГБОУ ВО «Алтайский государственный технический  
университет им. И. И. Ползунова»

Кафедра прикладной математики

**Индивидуальное задание**

на производственную практику

Технологическая (проектно-технологическая) практика

(вид и тип практики по УП)

студенту Замятин Иван Павлович группы ПИ-02

(Ф.И.О.)

**График проведения практики:**

№ п/ п	Содержание работ, выполняемых на практике	Сроки выполнения
1	Изучение организации работы предприятия и используемого на нем инструментария и ПО	19.06-21.06
2	Формулировка задач для решения в ходе практики, вида и объема результатов	21.06
3	Изучение и анализ предметной области, библиографический поиск, изучение литературы.	21.06 – 23.06
4	Постановка задачи, проектирование состава и структуры ПО	24.06 – 26.06
5	Реализация программного обеспечения	27.06 -13.07
6	Тестирование программного обеспечения	13.07 – 14.07
7	Оформление и сдача отчета по практике	15.07-16.07

Руководитель практики от университета \_\_\_\_\_ Лукоянычев В.Г., доцент, к.т.н.  
(подпись) (Ф.И.О., должность)

Задание принял к исполнению

  
(подпись)

Замятин И.П.  
(Ф.И.О.)

**Инструктаж по ОТ, ТБ, ПБ,  
ПВТР**

Инструктаж обучающегося по ознакомлению с требованиями охраны труда, техники безопасности, пожарной безопасности, а также правилами внутреннего трудового распорядка проведен «\_\_\_» \_\_\_\_\_ 2023 г.

Руководитель практики от  
университета

(подпись)

Лукоянычев В.Г., доцент, к.т.н.  
(Ф.И.О., должность)

Отзыв о прохождении практики студентом  
«Алтайского государственного технического университета им.  
И.И.Ползунова»  
Замятиным Иваном Павловичем

Замятин Иван Павлович, студент «Факультета Информационных Технологий» «Алтайского государственного технического университета им. И.И.Ползунова», проходил производственную практику в АО «Ритейл-Интеграция» в период с 19.06.2023 по 16.07.2023.

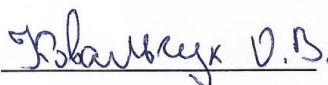
За время прохождения практики И. Замятин ознакомился с особенностями разработки драйвера для дисплея покупателя.

Во время практики И. Замятин проявил себя трудолюбивым работником, дисциплинированным исполнителем. Всю порученную работу выполнял добросовестно, применяя знания, полученные в ходе обучения в университете, стремился приобретать новые знания, чтобы быть еще более полезным. Неоднократно оказывал помощь сотрудникам предприятия.

Руководство компании АО «Ритейл-Интеграция» оценивает работу И.П. Замятина на «65».

Руководитель практики

  
(подпись)

  
(ФИО)

## Содержание

Введение.....	2
1 Описание предметной области и постановка задачи.....	5
1.1 Описание предметной области.....	5
1.2 Постановка задачи.....	11
2 Разработка программного обеспечения.....	27
2.1 Инструменты и технологии.....	27
2.2 Реализация.....	28
Заключение.....	30
Список используемых источников.....	31
Приложение А. Тестирование программного обеспечения.....	32
Приложение Б. Исходный текст программы.....	45

## **Введение**

### **1. Знакомство с организацией**

#### **1.1. Общая характеристика организации**

Компания «Ритейл Сервис» уже более 20 лет специализируется на проведении комплексной автоматизации предприятий розничной торговли, общественного питания, индустрии гостеприимства, развлекательных комплексов, производственных и других предприятий.

Компания оперативно решает любую задачу, связанную с автоматизацией предприятия и увеличением эффективности его работы. Она работает по принципу «одного окна» — осуществляет все: подбирает, доставляет и устанавливает оборудование и программное обеспечение, обучает персонал клиента, запускает предприятие клиента в работу, обеспечивает постоянную техническую поддержку и консультирование. Также одним из важнейших направлений компании является разработка программного обеспечения "ARTIX" (кассовый софт для розничных сетей продуктовой и не продуктовой направленности), его внедрение и сопровождение.

## **2. Особенности технологического процесса в учреждении**

### **2.1. Описание используемой методики управления проектами**

В компании используется методика SCRUM – методика гибкого управления проектами, помогающая командам структурировать работу и управлять ею на основе набора ценностей, принципов и практик

Всего в компании есть 2 команды Scrum, состоящей из 3 тестировщиков и 7 разработчиков. Основой методики Scrum является sprint – ограниченный отрезок времени(в данном случае – 2 недели) в течении которого необходимо выполнить, а затем обсудить выполненную работу на совещании.

### **2.2. Описание используемых языков и технологий для разработки**

В компании активно используются такие языки программирования, как C++, Java и Python. Для C++ используют фреймворк QT.

Для Java используются такие фреймворки и инструменты, как Spring, Vaadin, Junit.

Для автоматизации сборки проектов на основе описания их структуры используется фреймворк Maven.

Для отслеживания, управления и изменений схем баз данных используется библиотека Liquibase.

В компании используется такие СУБД как PostgreSQL, MySQL, MongoDB.

Для того, чтобы вести непрерывную разработку используется GitFlow.

## **3. Техническое задание**

**Название проекта:** Поддержать дисплей покупателя posiflex pd 2600.

### **Введение**

Дисплей покупателя в основном используется для вывода информации о товаре.

Данное приложение будет использоваться для тестирования работоспособности оборудования

Необходимо будет проверить печать вывода текста в 2-х строках.

### **Функциональные требования**

Написать графическое приложение, в котором можно:

- Открыть порт с выбранными параметрами (путь до порта, скорость и т.д.)
- Выводить текст (латиница) по кнопке из linedit
- Сделать кнопку очистки текста
- Переключать кодовые страницы (от 0 до 255) для вывода текста на русском языке
- Изменять кодировку текста посылаемого на дисплей покупателя (cp866 или cp1251 ) для вывода текста на русском языке
- Сделать вывод бегущей строки. Т.е. сделать checkbox (гонять текст по кругу), при выборе которого, после нажатия кнопки отправить текст на дисплей покупателя текст будет идти бегущей строкой

## **Нефункциональные требования**

- Интерфейс приложения должен быть интуитивно понятным и простым в использовании.
- Приложение должно иметь графический интерфейс пользователя (GUI).

## **Требования к реализации**

Приложение должно быть разработано на фреймворке Qt с использованием языка C++ для реализации интерфейсной части приложения и удобного программного взаимодействия с дисплеем.

## **Тестирование**

Необходимо провести тестирование приложения для проверки его функциональности и корректности работы.

Тестирование должно включать в себя отправку данных в заданной кодировке, для проверки ввода русский и английский символов. Проверку вывода данных в две строки, и проверка работы дисплея в режиме бегущей строки.

## **1. Описание предметной области и постановка задачи**

### **1.1. Описание предметной области**

Разработка программного обеспечения (ПО) – это процесс, в ходе которого происходит создание, оценка, тестирование и пуск в эксплуатацию программного обеспечения или информационной системы.

Этапы разработки ПО обычно включают в себя следующие шаги:

1. Сбор требований или анализ потребностей пользователя. На этом этапе разработчики взаимодействуют с заказчиком или конечными пользователями для определения их нужд и требований к ПО.
2. Проектирование ПО. Когда требования сформулированы и согласованы, следующим шагом является проектирование ПО. Этот этап может включать в себя создание диаграмм, прототипов и других типов документации, которые помогают сконцентрироваться на архитектуре и дизайне системы.
3. Разработка или кодирование ПО. На этом этапе разработчики начинают писать программные коды для создания ПО.
4. Тестирование ПО. После того, как код написан, он подвергается тестированию, чтобы убедиться в отсутствии ошибок и соответствии требованиям пользователя.
5. Поддержка и сопровождение ПО, включает в себя различные виды деятельности после того, как ПО внедрено и работает, такие как исправление ошибок, добавление новых функций и т.д.
6. Развертывание или внедрение ПО, это процесс установки программного обеспечения на инфраструктуре пользователя и его настройка.

Кассовое ПО (программное обеспечение) — это кассовая программа, учитывающая каждую операцию, которая проходит по кассе. Программа сама рассчитывает стоимость, принимает оплату, формирует чек, указывает реквизиты и направляет данные на печать, а информацию о транзакциях автоматически передает оператору фискальных данных и в ФНС.

Дисплей покупателя Posiflex PD-2600 — это вакуумно-флуоресцентный дисплей с ярким двухстрочным экраном, в каждой строке по 20 алфавитно-цифровых символов. Дисплей покупателя отличается небольшими размерами, имеют эргономичный дизайн. Серия PD-2600 имеет модели для автономной установки, а так же встраиваемые в терминалы Posiflex серии PB, HT, TP, KS, FT, DT.

Поддержать дисплей покупателя Posiflex PD 2600 означает обеспечить его правильную работу и функционирование. Это может включать в себя установку необходимых драйверов, настройку параметров, решение проблем с подключением и прочие технические вопросы, связанные с эксплуатацией этого устройства. В целом, поддержка дисплея покупателя может включать в себя как техническую поддержку, так и консультации по его использованию и настройке.



## 1.2. Постановка задачи

Необходимо поддерживать дисплей покупателя posiflex pd 2600.

Дисплей покупателя в основном используется для вывода информации о товаре. Данное приложение будет использоваться для тестирования работоспособности оборудования

Необходимо будет проверить печать вывода текста в 2-х строках.

### Функциональные требования

Написать графическое приложение, в котором можно:

- Открыть порт с выбранными параметрами (путь до порта, скорость и т.д.)
- Выводить текст (латиница) по кнопке из linedit
- Сделать кнопку очистки текста
- Переключать кодовые страницы (от 0 до 255) для вывода текста на русском языке
- Изменять кодировку текста посылаемого на дисплей покупателя (cp866 или cp1251 ) для вывода текста на русском языке
- Сделать вывод бегущей строки. Т.е. сделать checkbox (гонять текст по кругу), при выборе которого, после нажатия кнопки отправить текст на дисплей покупателя текст будет идти бегущей строкой

### Нефункциональные требования

- Интерфейс приложения должен быть интуитивно понятным и простым в использовании.
- Приложение должно иметь графический интерфейс пользователя (GUI).

Результатом выполнения задачи должно быть готовое GUI-приложение, позволяющие пользователю провести необходимую настройку порта, убедиться в верной кодировке отправленных сообщений, и вывода их в двух строках, и проверку работоспособности бегущей строки.

## **2. Разработка программного обеспечения**

### **2.1. Инструменты и технологии**

Для выполнения поставленной задачи используются:

**C++** – широко используется в разработке драйверов и операционных систем, что делает его очень полезным для работы с железом.

**Qt** – является фреймворком для разработки кроссплатформенных приложений на C++, который предоставляет удобный интерфейс для работы с различными аппаратными интерфейсами, включая порты ввода-вывода. Qt содержит множество классов и методов, которые позволяют легко и эффективно работать с железом, включая возможность работы с последовательным портом (Serial Port), USB-устройствами, Bluetooth и другими интерфейсами. Благодаря этому Qt является удобным инструментом для разработки программ, которые взаимодействуют с железом по порту.

**ESC/POS** – это стандарт команд, используемых для управления и управления принтерами чеков, кассовыми аппаратами и другими устройствами POS (точка продаж) через последовательный порт. ESC/POS команды позволяют настраивать шрифты, выравнивание, размер бумаги, настраивать печать штрих-кодов и многое другое. Они могут быть отправлены в устройство POS с помощью программного обеспечения, написанного на языке программирования C++, используя библиотеки, такие как Qt.

## 2.2. Реализация

При запуске программы появляется окно настройки порта и опправки сообщений. При активном подключении СОМ- порта он уже будет отображаться в окне, если их несколько то можно выбрать нужный. Далее можно выбрать скорость передачи, биты данных, бит четности и т. д.. При запуске окна, уже выбраны оптимальные настройки порта, но при желании, можно настроить под себя. При неактивном подключении идентификатор и bar\_status сообщает о том что устройство не подключено, при, успешном, подключении идентификатор меняется, что даёт пользователю понять об успешном подключении устройства.

Как только произошёл коннект с дисплеем с ним можно работать. Изначально дисплей не поддерживал кодировку для русских символов, и пришлось программно изменить её. После настройки кодировки символов можно отправлять как английские, так и русские сообщения. При постоянной отправки текста, он наслаиваться друг на друга, выглядеть это так «ПриветДомКотМолоко», т. е., отправка нескольких сообщений будет в одной строке, и каждый раз делать очистку дисплея, чтобы стирать предыдущие сообщение неудобно. Поэтому перед каждой отправкой дисплей сбрасывается, т. е. Автоматически устанавливает каретку в начальное положение (верхний левый угол), и зачищает весь текст. Это обеспечивает корректную и бесперебойную отправку сообщений.

При нажатии на «Ввод нижней строки» весь последующий отправленный текст будет отображаться на нижней строке, опять же это реализуется за счет ESC/POS команд. Вообще вся работа с текстом осуществляется благодаря этим командам. Но для работы мне хватило и основных, такие как:

- Сброс принтерам
- Перевод строки
- Настройка кодовой страницы
- Установка рабочей строки
- Выравнивание текста

Реализация бегущей строки, осуществляется программно. На порт непрерывно посылается текст с уже изменённой позицией, но длина сообщения не должна превышать 20 символов.

### Заключение

Разработанное приложение позволяет проверить работоспособность устройства, осуществить настройку их соединения, убедиться в правильной кодировке и протестировать несколько режимов работы. Всё это направлено на первичную проверку оборудования и дальнейшую разработку кассового ПО, используя уже готовые наработки. Это может быть удобно для тестирования нового оборудования, для дальнейшей работы.

## **Список используемых источников**

1. Онлайн библиотека «Википедия», URL: <https://ru.wikipedia.org>
2. Сообщество IT-специалистов «Хабр», URL: <https://habr.com/ru>
3. Документация Qt, URL: [Qt Serial Port - Qt Wiki](#)
4. ESC/POS Команды, URL: [Commande ESCPOS.pdf](#)
5. Система вопросов и ответов о программировании «Stack Overflow», URL: <https://stackoverflow.com/>

## Приложение А. Тестирование программного обеспечения

Окно работы с дисплеем (Не подключен).

GUIApp Zamyatin\_Practika

Настройки порта:

Порт

COM3

Скорость передачи

9600

Биты данных

8

Бит четности

None

Стоп-бит

1

Управление потоком

None


Отправка сообщения

☐ вводить нижнюю строку

☐ гонять текст по кругу

Отправить

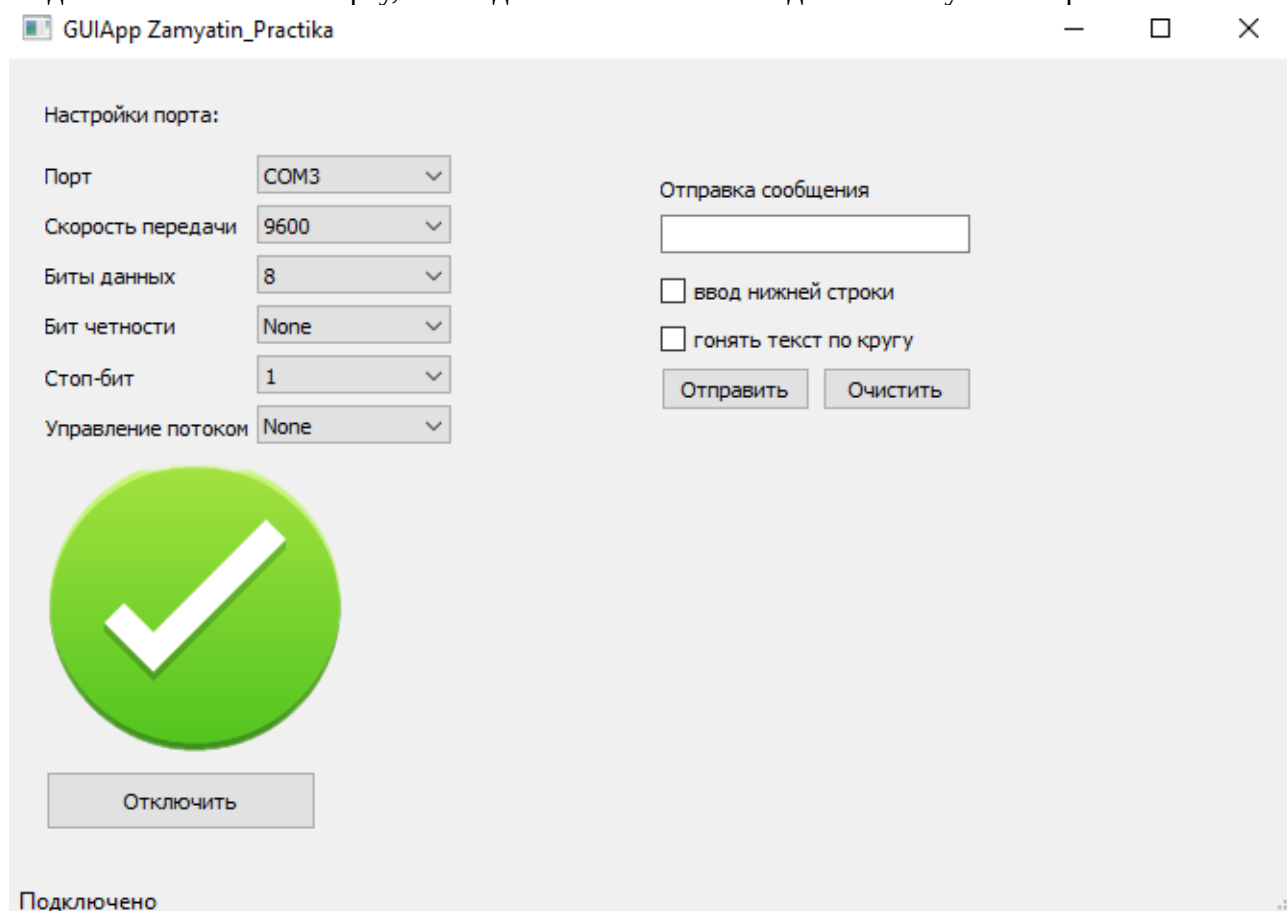
Очистить



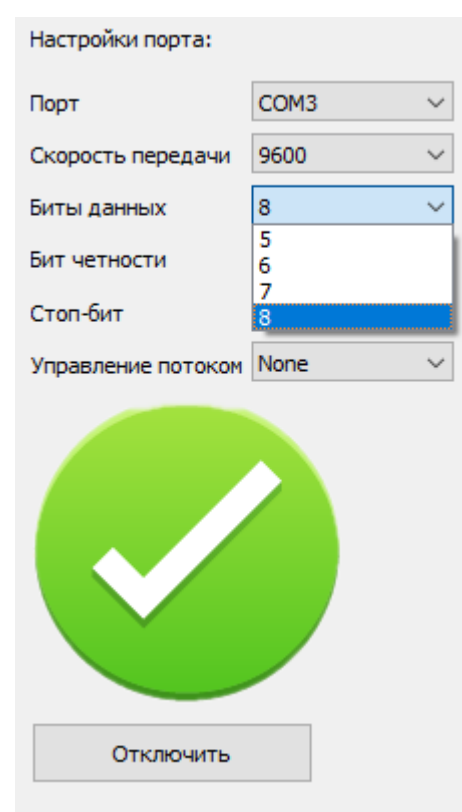
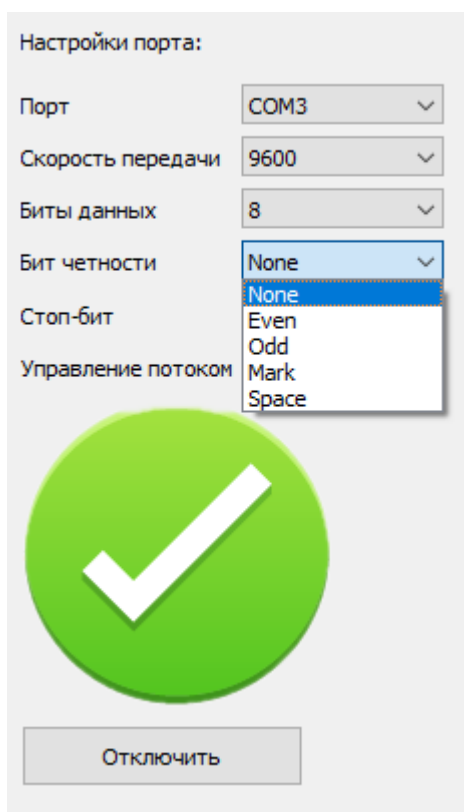
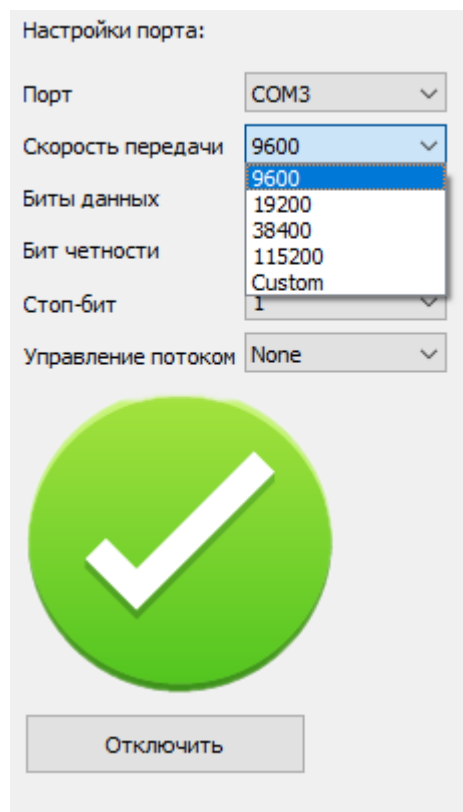
Подключить

Не подключено

Подключение по Com порту, как видно имя активного подключения уже отображено.




Настройка порта



Настройки порта:


Порт	COM3
Скорость передачи	9600
Биты данных	8
Бит четности	None
Стоп-бит	1
Управление потоком	None



Отключить

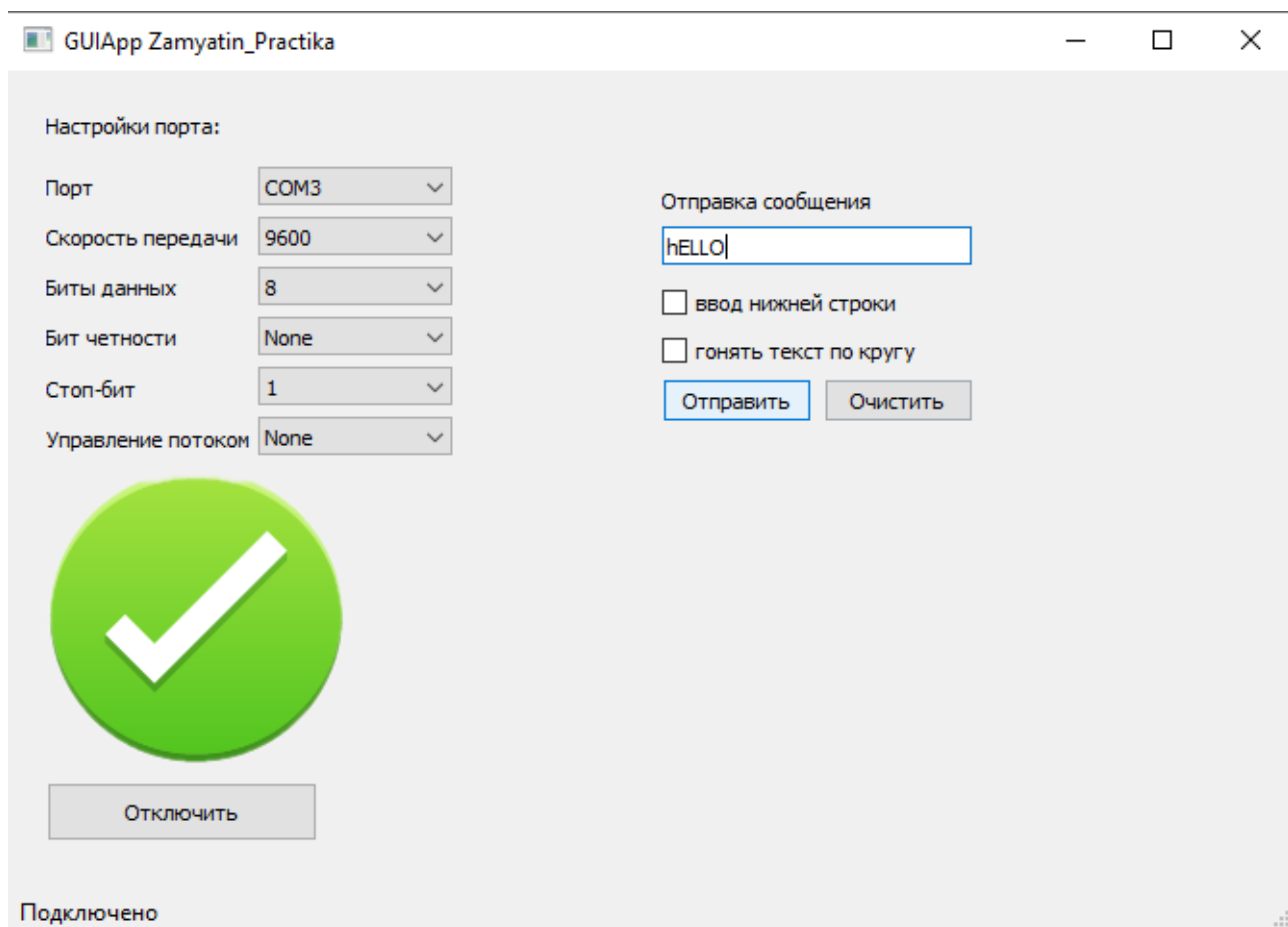
Настройки порта:

Порт	COM3
Скорость передачи	9600
Биты данных	8
Бит четности	None
Стоп-бит	1
Управление потоком	None



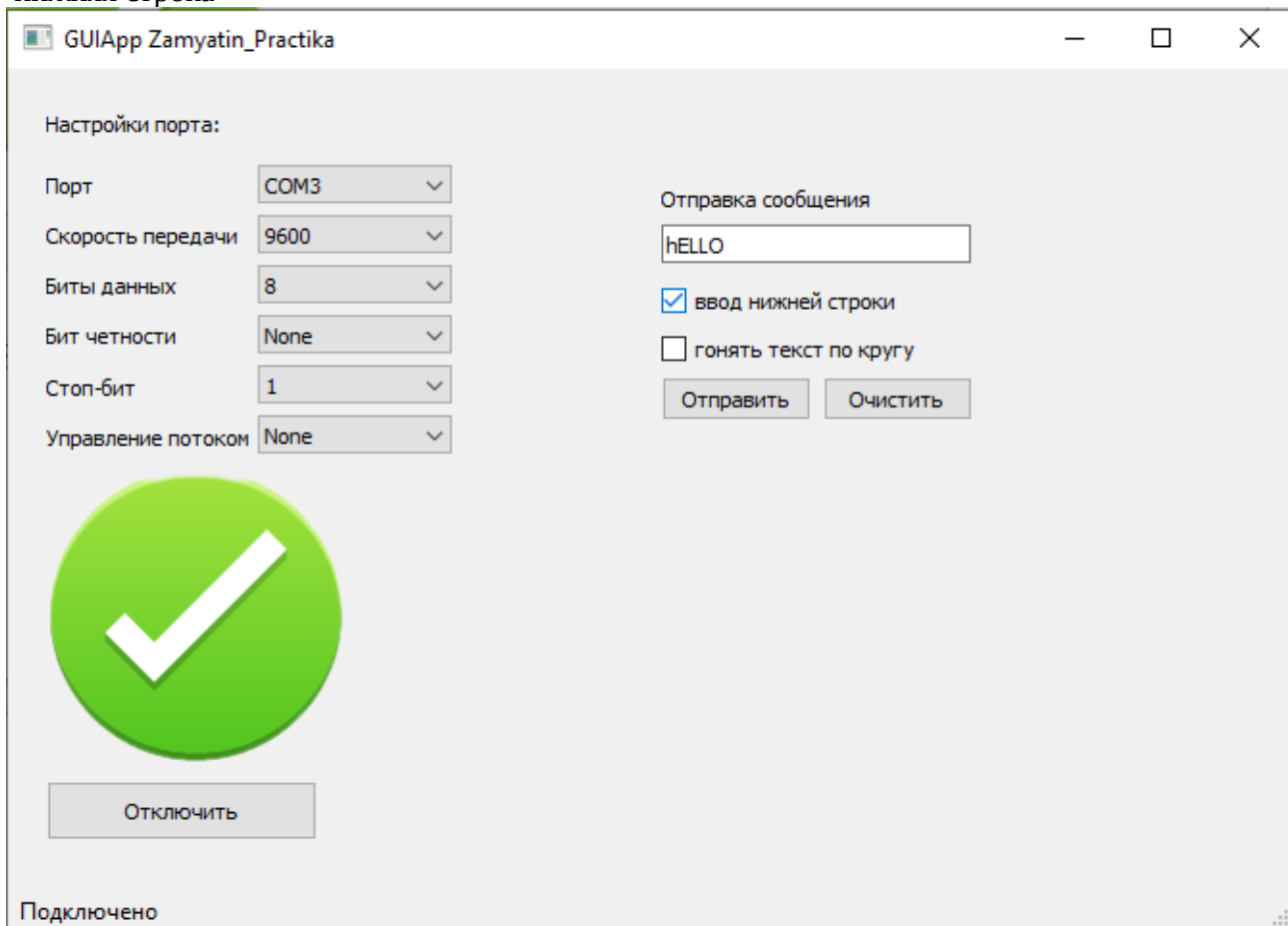
Отключить

оптовика сообщения

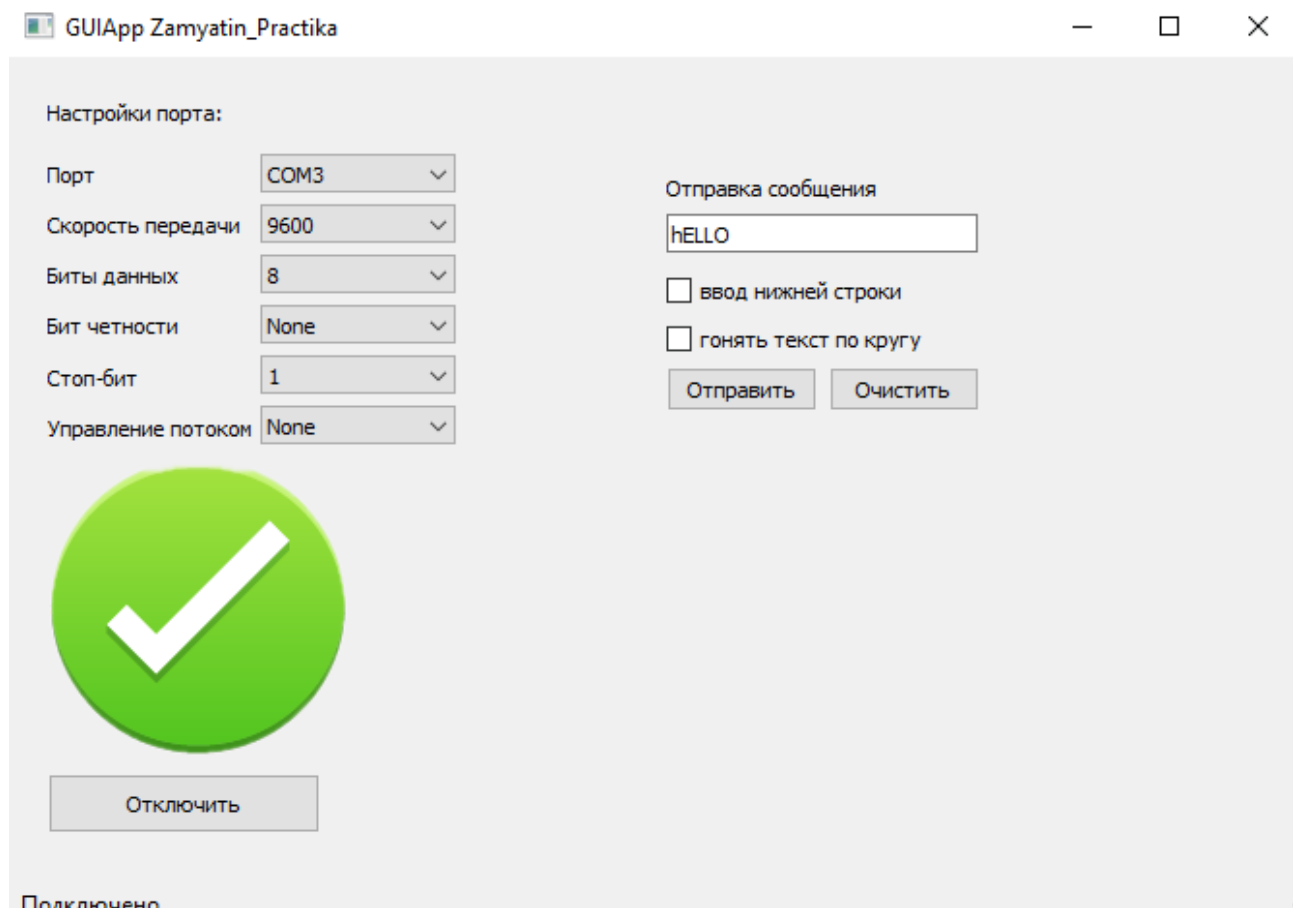




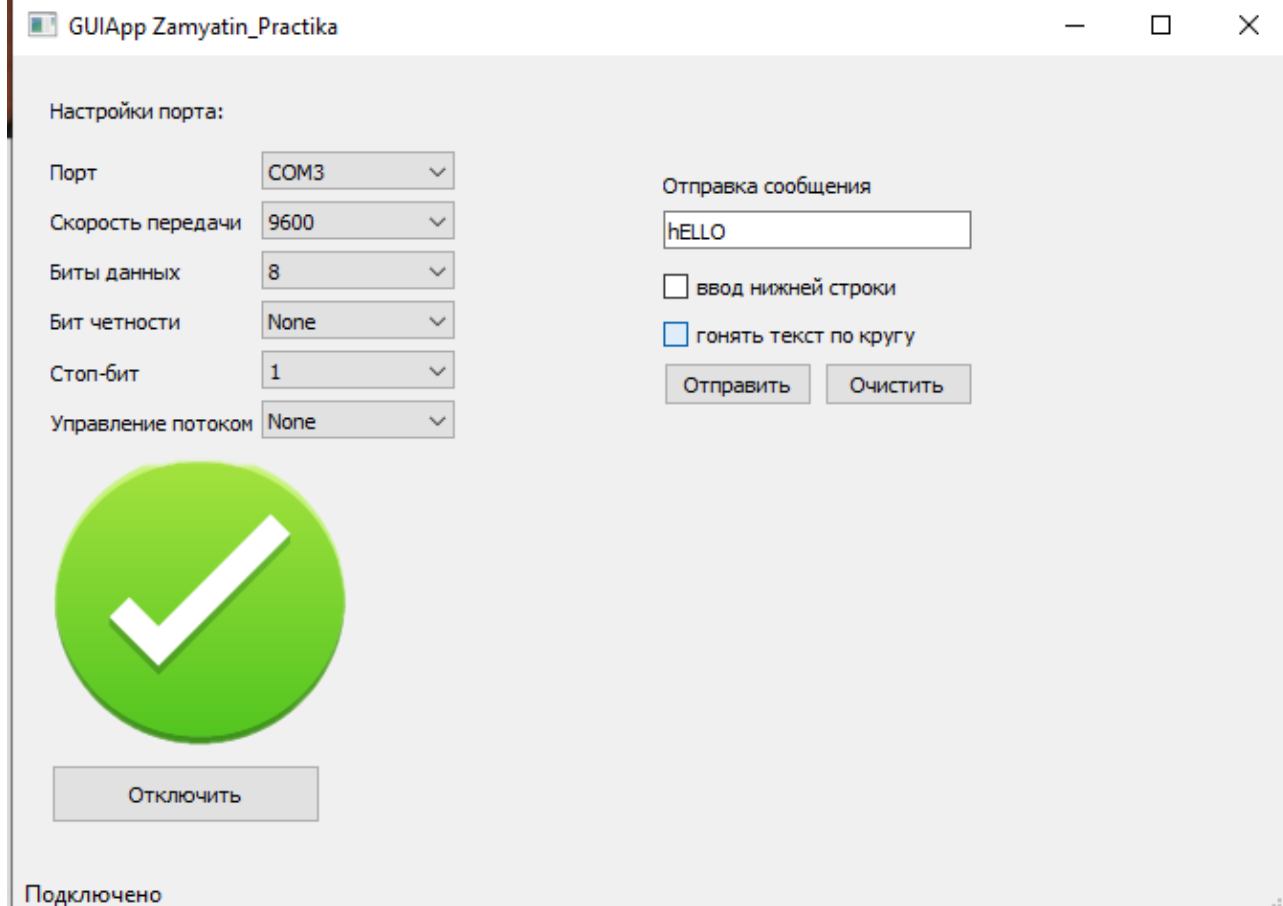
нижняя строка



## Возврат на верхнюю строку

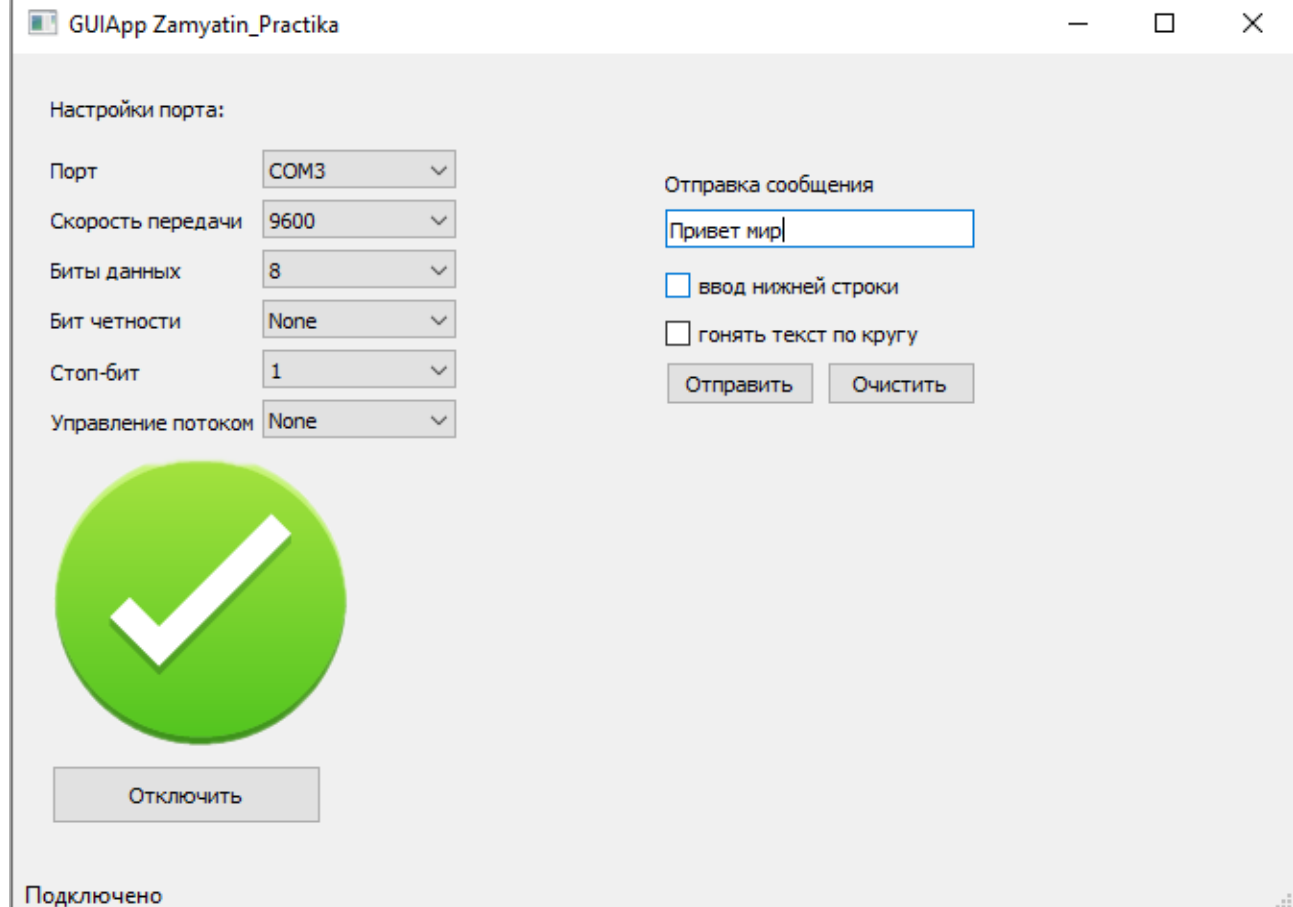


Режим бегущей строки, мой тест есть на ютубе. Переход осуществляется по ссылке внизу.

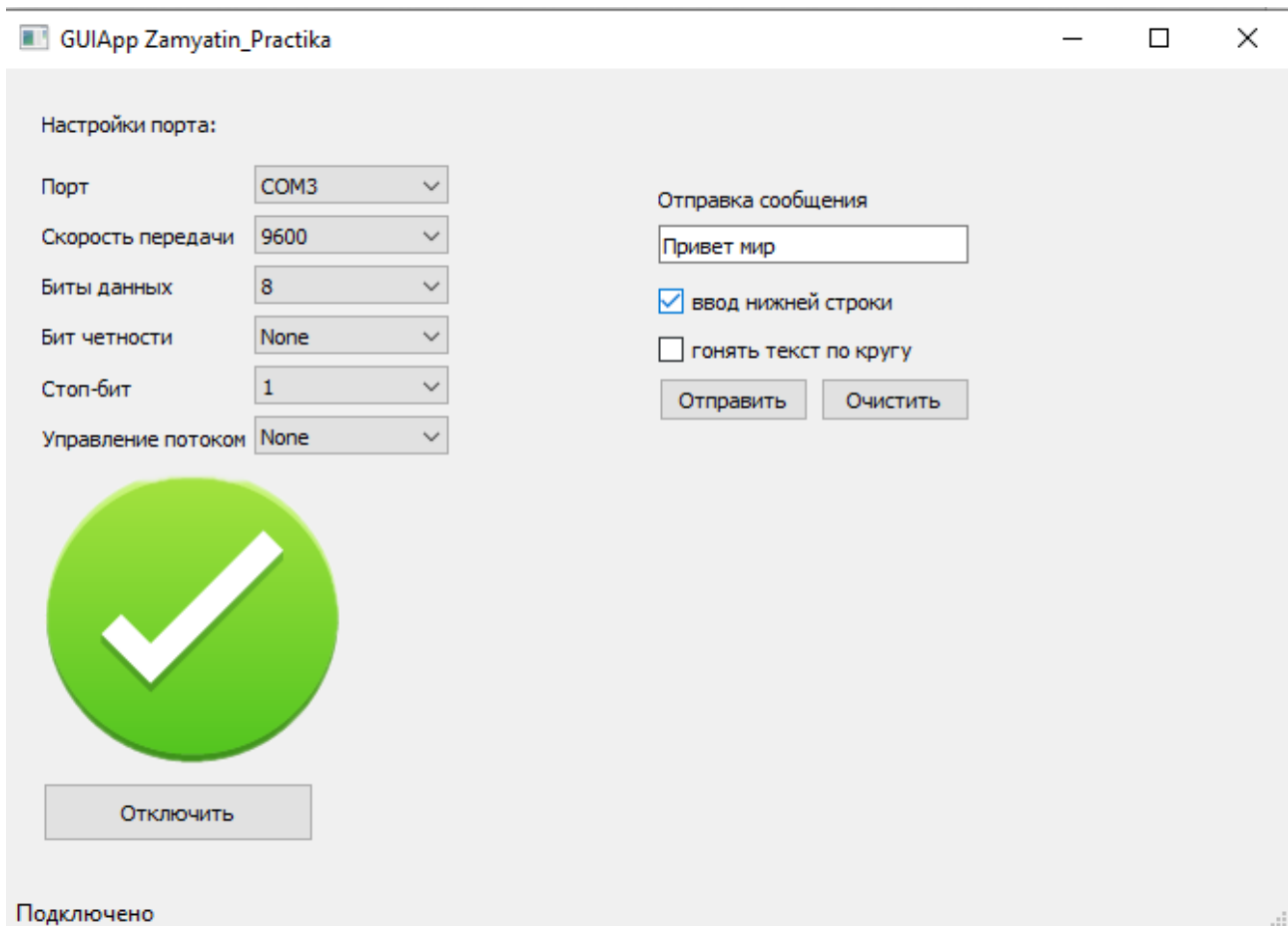


ссылка на тест бегущей строки: [ссылка на тест бегущей строки](#)

Ввод русских символов.



Нижняя строка.



## Приложение Б. Исходный текст программы

### mainwindow.h

```
#ifndef MAINWINDOW_H
#define MAINWINDOW_H

#include <QMainWindow>
#include <QtSerialPort/QtSerialPort>
#include <QtSerialPortInfo>
#include <QMessageBox>
#include <QPixmap>
#include <QTextCodec>
#include <QTimer>
#include <QDebug>
#include <QLineEdit>
#include <QThread>
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 ChangeNameButton();

    void on_Connect_clicked(); //подключение по порту

    void on_pushButton_clicked(); //отправка сообщения

    void on_pushButton_2_clicked(); //отчистка сообщения

    void on_checkBox_stateChanged(int arg1); //режим бегущей строки
    void setFirstLineTicker(bool status);
    void startFristLineTicker();
    void sendData(); //отправка/смена позиции бег. строки
    void changePosition(); //перевод на 2ю строку
    void on_checkBox_2_stateChanged(int arg1); //ожидание нажатия для перевода
строки
private:
    Ui::MainWindow *ui;
    QSerialPort serialPort;
    QTimer timer;
    QString message;
    QString firstLineTh();
    QThread *thread = new QThread();
    char copyBack;
    int N = 0;
    bool firstLineTicker = false;
    int position;
    bool isCheck_1 = false;
    bool isCheck_2 = false;
    int shiftArray; //сдвиг строки от начала
```



```
bool isConnected = false;

void connect(bool x) {
    if (isConnected!=x)
    {
        isConnected = x;
        emit changeIsConnected();
    }
}
signals:
    void changeIsConnected();
    void finished();
};
#endif // MAINWINDOW_H
```

## mainwindow.cpp

```
#include "mainwindow.h"
#include "ui_mainwindow.h"
#include <QMessageBox>
#include <QGridLayout>
#include <QSpinBox>
#include <QLineEdit>
#include <QTextCodec>
MainWindow::MainWindow(QWidget *parent)
    : QMainWindow(parent)
    , ui(new Ui::MainWindow)
{
    ui->setupUi(this);
    QSerialPort serial;

    //serial = new QSerialPort(this);
    auto infos = QSerialPortInfo::availablePorts();
    for (auto &info : infos) {
        ui->comboBox->addItem(info.portName());
    }
    // Инициализация переменных для бегущей строки
    position = 0;

    ChangeNameButton();
    QObject::connect(this, &MainWindow::changeIsConnected, this,
&MainWindow::ChangeNameButton);
    // Устанавливаем скорость передачи
    ui->comboBox_baudRate->addItem(QStringLiteral("9600"),
QSerialPort::Baud9600);
    ui->comboBox_baudRate->addItem(QStringLiteral("19200"),
QSerialPort::Baud19200);
    ui->comboBox_baudRate->addItem(QStringLiteral("38400"),
QSerialPort::Baud38400);
    ui->comboBox_baudRate->addItem(QStringLiteral("115200"),
QSerialPort::Baud115200);
    ui->comboBox_baudRate->addItem(tr("Custom"));

    // Устанавливаем биты данных
    ui->comboBox_dataBits->addItem(QStringLiteral("5"), QSerialPort::Data5);
    ui->comboBox_dataBits->addItem(QStringLiteral("6"), QSerialPort::Data6);
    ui->comboBox_dataBits->addItem(QStringLiteral("7"), QSerialPort::Data7);
    ui->comboBox_dataBits->addItem(QStringLiteral("8"), QSerialPort::Data8);
    ui->comboBox_dataBits->setCurrentIndex(3);

    // Установить бит четности
    ui->comboBox_parity->addItem(tr("None"), QSerialPort::NoParity);
    ui->comboBox_parity->addItem(tr("Even"), QSerialPort::EvenParity);
    ui->comboBox_parity->addItem(tr("Odd"), QSerialPort::OddParity);
    ui->comboBox_parity->addItem(tr("Mark"), QSerialPort::MarkParity);
    ui->comboBox_parity->addItem(tr("Space"), QSerialPort::SpaceParity);

    // Установить стоп-бит
    ui->comboBox_stopBit->addItem(QStringLiteral("1"),
QSerialPort::OneStop);
    ui->comboBox_stopBit->addItem(QStringLiteral("2"),
QSerialPort::TwoStop);

    // Добавить управление потоком
    ui->comboBox_flowBit->addItem(tr("None"), QSerialPort::NoFlowControl);
    ui->comboBox_flowBit->addItem(tr("RTS/CTS"),
QSerialPort::HardwareControl);
```



```

        ui->comboBox_flowBit->addItem(tr("XON/XOFF"),
QSerialPort::SoftwareControl);

}

MainWindow::~MainWindow()
{
    delete ui;
}

void MainWindow::ChangeNameButton()
{
    if (isConnected) {
        ui->Connect->setText("Отключить");
        QPixmap pix(":/img/img/1.png");
        ui->image->setPixmap(pix);
        ui->statusbar -> showMessage("Подключено");
    }
    else
    {
        ui->Connect->setText("Подключить");
        QPixmap pix(":/img/img/2.png");
        ui->image->setPixmap(pix);
        ui->statusbar -> showMessage("Не подключено");
    }
}

void MainWindow::setFirstLineTicker(bool status)
{
    this->firstLineTicker = status;

    // emit this->firstLineStatusTickerChanged(status);
};

void MainWindow::startFristLineTicker()
{
    this->setFirstLineTicker(true);
    thread->start();
}

void MainWindow::on_Connect_clicked()
{
    if (isConnected) {
        connect(false);
        serialPort.close();
        auto infos = QSerialPortInfo::availablePorts();
        ui->comboBox->clear();
        for (auto &info : infos) {
            ui->comboBox->addItem(info.portName());
        }
    }
    else {
        QString s = ui->comboBox->currentText();
        serialPort.setPortName(s);
        serialPort.setBaudRate(ui->comboBox_baudRate->currentText().toInt());
        serialPort.setDataBits(QSerialPort::Data8);
        serialPort.setParity(QSerialPort::NoParity);
        serialPort.setStopBits(QSerialPort::OneStop);
        connect(serialPort.open(QIODevice::ReadWrite));
        auto infos = QSerialPortInfo::availablePorts();
        ui->comboBox->clear();
        for (auto &info : infos) {

```

```

        ui->comboBox->addItem(info.portName());
    }
}

void MainWindow::on_pushButton_clicked()
{
    if (!isConnected) return;
    if (!isChecked_1 && !isChecked_2) {

        QByteArray str;
        QString unicodeText = ui->lineEdit->text();
        QTextCodec* codec = QTextCodec::codecForName("IBM866");//изменение
кодировки на cp866
        QByteArray cp866Text = codec->fromUnicode(unicodeText);
        str.append("\x1b" "\x40");//сброс дисплея
        str.append("\x1B" "\x61" "\x00");//выравнивание слева
        str.append("\x1B" "\x74" "\x06");//изменение кодировки
        str.append(cp866Text);
        serialPort.clear(QSerialPort::AllDirections);
        serialPort.write(str);

    }
}

void MainWindow::on_pushButton_2_clicked()
{
    QByteArray str ;
    str.append("\x1b" "\x40");//сброс дисплея

    serialPort.clear(QSerialPort::AllDirections);
    serialPort.write(str);
    if (timer.isActive()) {
        timer.stop();
    }
}

void MainWindow::sendData()
{
    QString unicodeText = ui->lineEdit->text();
    QTextCodec* codec = QTextCodec::codecForName("IBM866");//изменение кодировки
на cp866
    QByteArray cp866Text = codec->fromUnicode(unicodeText);
    QByteArray str (unicodeText.toLocal8Bit());

    str.append(QByteArray(20 - unicodeText.size(), ' '));//расширение сообщение
до 20 символов
    //пока поток открыт, сообщение двигаться
    while (isChecked_1) {
        if (serialPort.isOpen()) {
            serialPort.write("\x0c"+str);//сдвиг только в первой строке
            serialPort.waitForBytesWritten();
            copyBack = str.back();//копирование символа в начало
            str.remove(20 - 1, 1);//удаление с конца
            str.push_front(copyBack);//вставка в начало
            QThread::sleep(1);
        }
    }
};

```

```

}

void MainWindow::changePosition()
{
    QByteArray str;
    QString unicodeText = ui->lineEdit->text();
    QTextCodec* codec = QTextCodec::codecForName("IBM866");//изменение кодировки
на cp866
    QByteArray cp866Text = codec->fromUnicode(unicodeText);
    str.append("\x1b" "\x40");//сброс дисплея
    str.append("\x1B" "\x61" "\x00");//выравнивание слева
    str.append("\x0a");//перевод каретки на другую строку
    str.append("\x1B" "\x74" "\x06");//изменение кодировки
    str.append(cp866Text);
    serialPort.clear(QSerialPort::AllDirections);
    serialPort.write(str);
}

void MainWindow::on_checkBox_stateChanged(int arg1)
{
    if (arg1) {
        isChecked_1 = true;
        thread->start();

        QObject::connect(thread, &QThread::started, this,
&MainWindow::sendData);
        QObject::connect(this, &MainWindow::finished, thread, &QThread::quit);

    }else if(!arg1) {
        emit finished();

        thread->quit();
        thread->wait();
        isChecked_1 = false;
    }
}

void MainWindow::on_checkBox_2_stateChanged(int arg1)
{
    if (arg1) {
        isChecked_2 =true;
        QObject::connect(ui->pushButton,&QPushButton::clicked, this,
&MainWindow::changePosition);
    }
    else {
        QObject::connect(ui->pushButton, &QPushButton::clicked, this,
&MainWindow::on_pushButton_clicked);
        isChecked_2 = false;
    }
}

```

## mainwindow.ui

Пишите здесь

Настройки порта:

Порт	<input type="text"/>
Скорость передачи	<input type="text"/>
Биты данных	<input type="text"/>
Бит четности	<input type="text"/>
Стоп-бит	<input type="text"/>
Управление потоком	<input type="text"/>

Отправка сообщения

☐ ввод нижней строки

☐ гонять текст по кругу