МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.Г.ШУХОВА»

(БГТУ им. В.Г.Шухова)

**Лабораторная работа №7**

дисциплина «Сети ЭВМ и телекоммуникации»

тема: «Протоколы SMTP и POP3»

Выполнил: студент группы ВТ-31

Ковалёв И. Д.

Проверил: Федотов Е. А.

Белгород 2020

**Цель работы**: изучить принципы и характеристику протоколов POP3 и SMTP и составить программу для приема/отправки электронной почты.

**Теоретические сведения**

**Протокол POP3**

POP3 – стандартный Интернет-протокол прикладного уровня, используемый клиентами электронной почты для извлечения электронного сообщения с удаленного сервера по TCP/IP-соединению. В некоторых небольших узлах Интернет бывает непрактично поддерживать систему передачи сообщений. Рабочая станция может не иметь достаточных ресурсов для обеспечения непрерывной работы SMTP-сервера. Для “домашних ЭВМ” слишком дорого поддерживать связь с Интернет круглые сутки. Но доступ к электронной почте необходим как для таких малых узлов, так и индивидуальных ЭВМ. Для решения этой проблемы разработан протокол POP3.

Этот протокол обеспечивает доступ узла к базовому почтовому серверу. POP3 не ставит целью предоставление широкого списка манипуляций с почтой. Почтовые сообщения принимаются почтовым сервером и сохраняются там, пока на рабочей станции клиента не будет запущено приложение POP3. Это приложение устанавливает соединение с сервером и забирает сообщения оттуда. Почтовые сообщения на сервере стираются.

POP3 поддерживает простые требования «загрузи-и-удали» для доступа к удаленным почтовым ящикам. Хотя большая часть POP- клиентов предоставляют возможность оставить почту на сервере после загрузки, использующие POP клиенты обычно соединяются, извлекают все письма, сохраняют их на пользовательском компьютере как новые сообщения, удаляют их с сервера, после чего разъединяются.

Другие протоколы, в частности IMAP, предоставляют более полный и комплексный удаленный доступ к типичным операциям с почтовым ящиком.

Доступные сообщения клиента фиксируются при открытии почтового ящика POP-сессией и определяются количеством сообщений для сессии, или, по желанию, с помощью уникального идентификатора, присваиваемого сообщению POP-сервером. Этот уникальный идентификатор является постоянным и уникальным для почтового ящика и позволяет клиенту получить доступ к одному и тому же сообщению в разных POP-сессиях. Почта извлекается и помечается для удаления с помощью номера сообщения. При выходе клиента из сессии помеченные сообщения удаляются из почтового ящика.

Обычно РОРЗ-сервис устанавливается на 110-й ТСР-порт сервера, который будет находиться в режиме ожидания входящего соединения.

Когда клиент хочет воспользоваться РОРЗ-сервисом, он просто устанавливает TCP-соединение с портом 110 этого хоста. После установления соединения сервис РОРЗ отправляет подсоединившемуся клиенту приветственное сообщение. После этого клиент и сервер начинают обмен командами и данными. По окончании обмена РОРЗ-канал закрывается.

РОРЗ-сервер может использовать в своей работе таймер контроля времени соединения. Этот таймер отсчитывает время "бездействия" ("idle") клиента в сессии от последней переданной команды. Если время сессии истекло, сервер закрывает TCP-соединение, не переходя в состояние UPDATE (иными словами, откатывает транзакцию или на языке баз данных — выполняет ROLLBACK).

**Протокол SMTP**

SMTP - широко используемый сетевой протокол, предназначенный для передачи электронной почты в сетях TCP/IP. SMTP впервые был описан в RFC 821; последнее обновление в RFC 5321 включает масштабируемое расширение - ESMTP. В настоящее время под «протоколом SMTP», как правило, подразумевают и его расширения.

Протокол SMTP предназначен для передачи исходящей почты, используя для этого порт TCP 25.

Со стороны пользователя обычно одна и та же программа выступает в роли и POP3 клиента, и SMTP клиента отправителя.

Если на каком-нибудь этапе передачи SMTP клиент обнаружит невозможность подключиться к следующему серверу, он будет пытаться отправить сообщение через некоторое время – 1 час, 4 часа, день и т.д. до 4 суток в общем случае. Причем временные отрезки между попытками, как правило, зависят от настроек программы-пересыльщика почты. Одновременно, такой сервер должен уведомить отправителя сообщения о невозможности доставить почту, послав ему стандартное письмо "Failed delivery" (доставка невозможна) и рассказав о графике дальнейших попыток по продвижению исходного сообщения. Если канал связи не восстановится за указанный большой промежуток времени (например, 4 дня), посланная информация будет считаться утерянной.

Как только почта достигнет конечного пункта (SMTP сервера адресата сообщения), она будет сложена в почтовый ящик абонента, который всегда сможет в удобное для него время изъять ее по протоколам POP3 или IMAP, в зависимости от того, какой из них поддерживается провайдером.

Анализируя заголовок письма, можно узнать какими путями оно путешествовало, как долго длился сам путь, как называлась почтовая программа отправителя и многое другое.

**Задание к работе**

1. Разработать программу, которая отправляет письмо с вложениями на указанный электронный адрес.

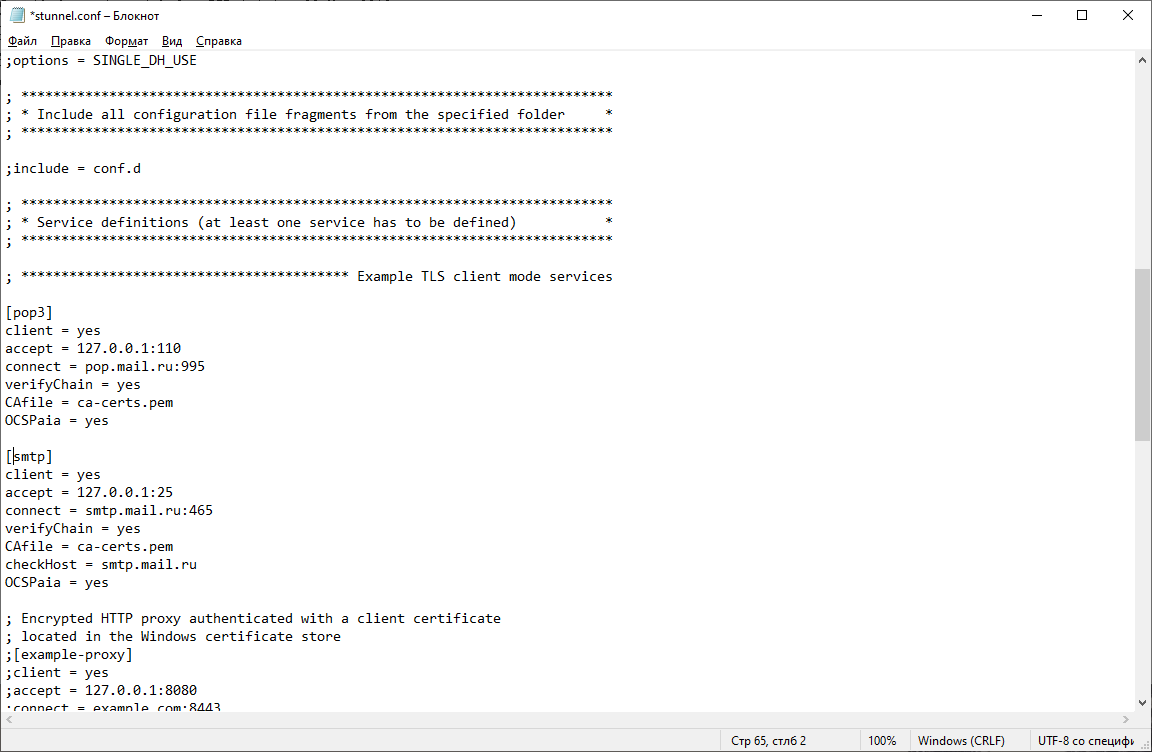
2. Разработать программу, получающую ответ, отправленный на указанный электронный адрес.

В программе для отправки письма пользователь должен указать почтовый сервер, почтовый адрес отправителя и получателя, логин, пароль, тему письма, текст сообщения, отправляемый файл.

В программе для получения списка писем пользователь должен указать почтовый сервер, логин и пароль. Сообщения выводятся в виде списка с номером и размером. Пользователь может прочитать или удалить любое из них.

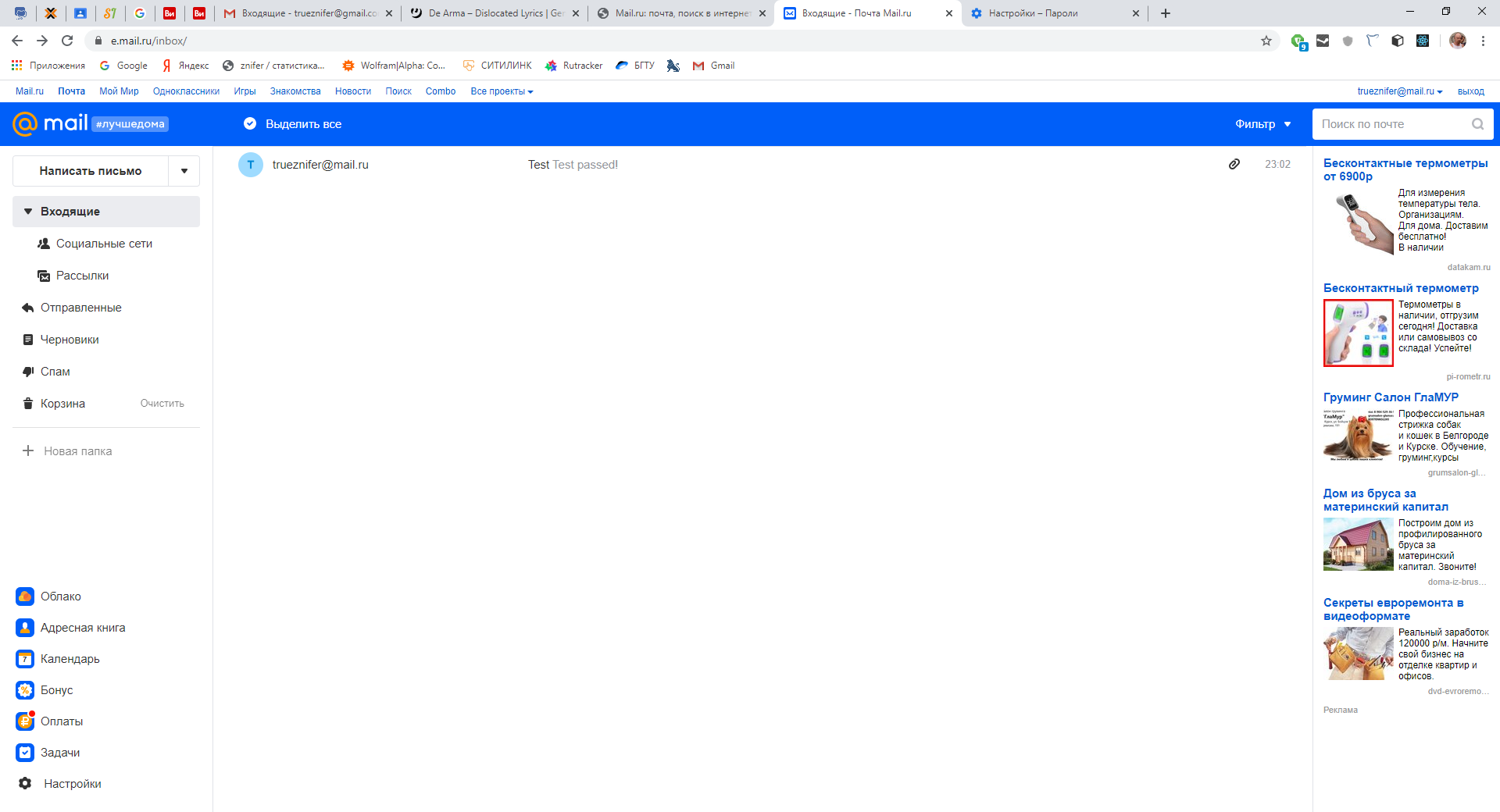
**Ход работы**

Поскольку в данный момент времени не удалось найти почтовый сервис, который бы не требовал ssl-сертификацию приложения, для промежуточной сертификации запроса была использована программа stunnel с такой конфигурацией:

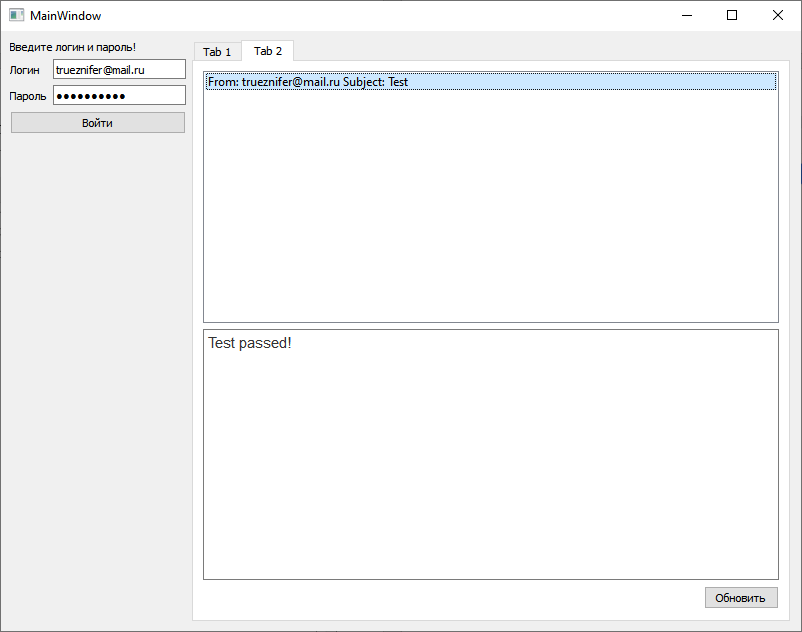


В качестве почтового сервиса была использована электронная почта от mail.ru.

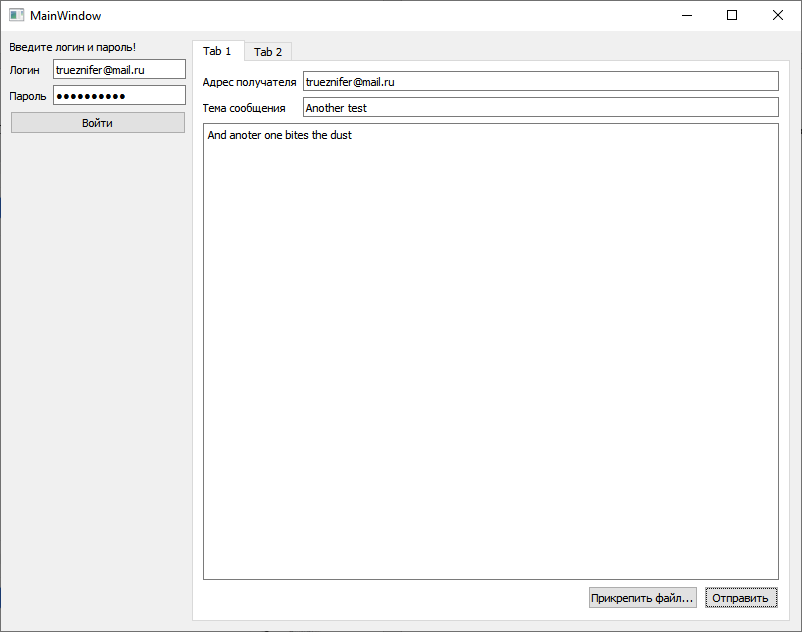
Для проверки работоспособности разработанной программы было отправлено сообщение:



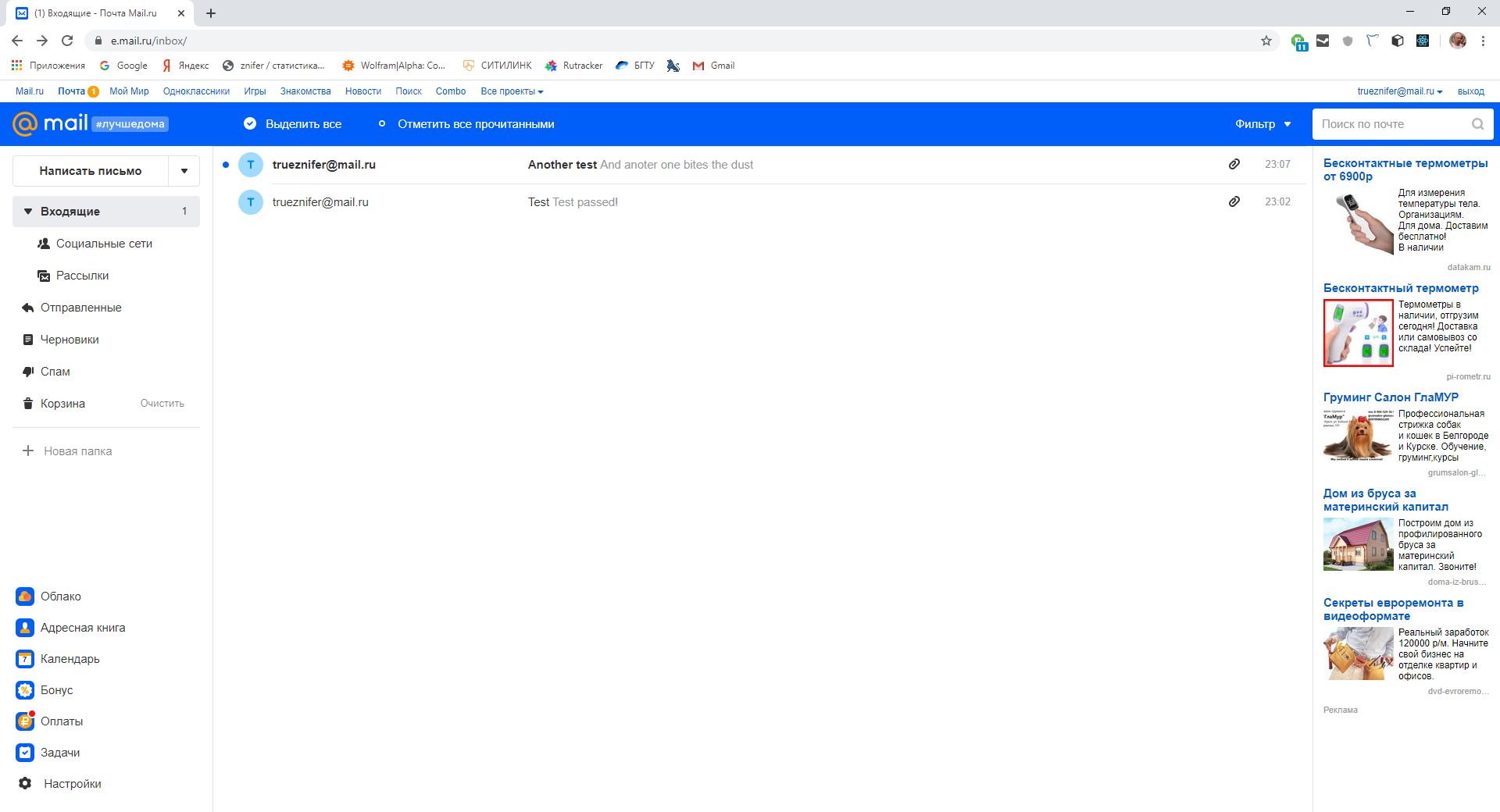
Проверим наличие этого сообщения с помощью программы:



Работает. Теперь отправим сообщение из разработанной программы:



И проверим почтовый ящик:



Сработало!

**Вывод:** в процессе выполнения работы были изучены принципы работы протоколов SMTP и POP3.

SMTP – это протокол, который поддерживает только отправку сообщений электронной почты. POP3 – протокол, который поддерживает получение сообщений. Данные протоколы используют протокол TCP/IP как транспортное средство для передачи сообщений.

**Контрольные вопросы**

1. Что представляет собой протокол POP3? С какой целью он был разработан?

POP3 – это протокол прикладного уровня, который предназначен для извлечения электронных сообщений с сервера с помощью TCP/IP. Разработан для рабочих станций, которые не имеют возможность обеспечивать постоянное подключение к почтовому серверу.

С помощью этого протокола клиент обращается к серверу для загрузки на клиент и удаления с сервера сообщений. Порядок работы:

- клиент соединяется сервером

- клиент запрашивает все сообщения

- клиент обновляет локальную базу сообщений

- сообщения с сервера удаляются

2. Опишите процесс работы протокола POP3.

Стандартный порт протокола РОР3 – 110. Сервер слушает входящие соединения на этом порту. Клиент открывает ТСР соединение, после чего сервер посылает клиенту приветствующее сообщение. Получив это сообщение, клиент может отправлять серверу команды и принимать данные. По окончанию ТСР-соединение закрывается.

3 Из каких частей состоит POP3-сессия?

Сессия РОР3 состоит из таких этапов:

- Приветствие после которого устанавливается состояние аутентификации

- Регистрация клиента

- Сессия переходит в состояние транзакции

- После окончания работы клиентом, сессия переходит в состояние завершения транзакции и фиксирует изменения

- Если сессия не была корректно закрыта, сервер пропустит состояние обновления, и отменит изменения, которые были произведены

4 Как осуществляется взаимодействие SMTP и POP3?

Оба протокола, как правило, работают на одном сервере. SMTP обеспечивает отправку и пересылку электронной почты, а РОР3 – получение этой почты клиентом. Когда клиент отправляет сообщение, открывается соединение SMTP с сервером, а сервер передает почту к почтовому серверу получателя. Загружая сообщение, клиент открывает сессию РОР3, и получает хранящееся на сервере сообщение.

5 Минимальный набор команд и порядок их применения для отправки почты по протоколу SMTP.

- HELO – идентификация

- MAIL – задает адрес отправителя

- RCPT – задает адрес получателя

- DATA – задает тело сообщения

- QUIT – окончание сессии.

**Листинг разработанной программы:**

#ifndef MAINWINDOW\_H

#define MAINWINDOW\_H

#include <QMainWindow>

#include <QMessageBox>

#include <QFile>

#include <QFileDialog>

#include <QTextStream>

#include <QTcpSocket>

#include <QListWidget>

QT\_BEGIN\_NAMESPACE

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

QT\_END\_NAMESPACE

*class* **MainWindow** : *public* QMainWindow

{

Q\_OBJECT

*public*:

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

~***MainWindow***();

QString **getMessageInfo**(int index);

*private* slots:

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

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

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

void **on\_mailList\_itemClicked**(QListWidgetItem \*item);

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

*private*:

Ui::MainWindow \*ui;

bool isUserLogged;

int port;

QString login, password, attachFile, server;

QByteArray attachment;

QTcpSocket socket, pop3socket;

bool isConnected;

int currentMessage;

QStringList infos;

};

#endif *//* *MAINWINDOW\_H*

#include "mainwindow.h"

#include "ui\_mainwindow.h"

MainWindow::**MainWindow**(QWidget \*parent):QMainWindow(parent), ui(*new* Ui::MainWindow){

ui->setupUi(*this*);

isUserLogged = *false*;

ui->passwordEdit->setEchoMode(QLineEdit::*Password*);

isConnected = *false*;

server = "127.0.0.1";

currentMessage = 0;

}

MainWindow::~***MainWindow***(){

*delete* ui;

}

void MainWindow::**on\_loginButton\_clicked**(){

QString request;

QString response;

password = ui->passwordEdit->text();

login = ui->loginEdit->text();

socket.*connectToHost*(server, 25);

*if*(socket.*waitForConnected*(5000)){

socket.*waitForReadyRead*();

response = socket.readAll();

request = "HELO " + server + "\r\n";

socket.write(request.toLocal8Bit());

socket.*waitForBytesWritten*();

socket.*waitForReadyRead*();

response = socket.readAll();

request = "AUTH LOGIN\r\n";

socket.write(request.toLocal8Bit());

socket.*waitForBytesWritten*();

socket.*waitForReadyRead*();

response = socket.readAll();

request = login.toUtf8().toBase64() + "\r\n";

socket.write(request.toLocal8Bit());

socket.*waitForBytesWritten*();

socket.*waitForReadyRead*();

response = socket.readAll();

request = password.toUtf8().toBase64() + "\r\n";

socket.write(request.toLocal8Bit());

socket.*waitForBytesWritten*();

socket.*waitForReadyRead*();

response = socket.readAll();

*this*->isConnected = *true*;

pop3socket.*connectToHost*(server, 110);

*if*(pop3socket.*waitForConnected*(5000)){

pop3socket.*waitForReadyRead*();

response = pop3socket.readAll();

request = "USER " + *this*->ui->loginEdit->text() + "\r\n";

pop3socket.write(request.toLocal8Bit());

pop3socket.*waitForBytesWritten*();

pop3socket.*waitForReadyRead*();

response = pop3socket.readAll();

request = "PASS " + password + "\r\n";

pop3socket.write(request.toLocal8Bit());

pop3socket.*waitForBytesWritten*();

pop3socket.*waitForReadyRead*();

response = pop3socket.readAll();

}

}

*if* (isConnected){

QMessageBox::warning(*this*, "Получилось!", "Произошло подключение");

}

}

void MainWindow::**on\_sendButton\_clicked**(){

QString request;

QString response;

*if* (*this*->isConnected){

request = "MAIL FROM: <" + login + ">\r\n";

socket.write(request.toLocal8Bit());

socket.*waitForBytesWritten*();

socket.*waitForReadyRead*();

response = socket.readAll();

request = "RCPT TO: <" + ui->receiverAdress->text() + ">\r\n";

socket.write(request.toLocal8Bit());

socket.*waitForBytesWritten*();

socket.*waitForReadyRead*();

response = socket.readAll();

request = "DATA\r\n";

socket.write(request.toLocal8Bit());

socket.*waitForBytesWritten*();

socket.*waitForReadyRead*();

response = socket.readAll();

request = "";

QTextStream messageStream(&request);

messageStream << "From: " << login << "\r\n";

messageStream << "Subject: " << ui->theme->text() << "\r\n";

messageStream << "MIME-Version: 1.0\r\n";

messageStream << "Content-Type: multipart/mixed; boundary=\"\*\*\*\*\*\*\*\*\*\*\"\r\n";

messageStream << "--\*\*\*\*\*\*\*\*\*\*\r\n";

messageStream << "Content-Type: text/html; charset=\"koi8-r\"\r\n";

messageStream << "\r\n";

messageStream << ui->message->toPlainText() << "\r\n";

messageStream << "--\*\*\*\*\*\*\*\*\*\*\r\n";

messageStream << "Content-Type: application/octet-stream; name=\"" << *this*->attachFile << "\"\r\n";

messageStream << "Content-Transfer-Encoding: base64\r\n\r\n";

messageStream << *this*->attachment.toBase64() << "\r\n";

messageStream << "--\*\*\*\*\*\*\*\*\*\*\r\n";

messageStream << ".\r\n";

socket.write(request.toLocal8Bit());

socket.*waitForBytesWritten*();

socket.*waitForReadyRead*();

response = socket.readAll();

}

}

void MainWindow::**on\_attachFileButton\_clicked**(){

*this*->attachFile = QFileDialog::getOpenFileName(*this*, "Выбрать вложение", *nullptr*, "Изображения (\*.png \*.jpg \*.bmp)");

QFile file(*this*->attachFile);

file.*open*(QIODevice::*ReadOnly*);

*this*->attachment = file.readAll();

file.*close*();

}

void MainWindow::**on\_mailList\_itemClicked**(QListWidgetItem \*item){

}

void MainWindow::**on\_updateButton\_clicked**(){

ui->mailList->clear();

*if* (!isConnected){

*return*;

}

QString request;

QString response;

request = "LIST\r\n";

pop3socket.write(request.toLocal8Bit());

pop3socket.*waitForBytesWritten*();

pop3socket.*waitForReadyRead*();

response = pop3socket.readAll();

int messages = response.split(" ")[1].toInt();

QStringList items;

*while* (items.length() < messages){

pop3socket.*waitForReadyRead*();

response = pop3socket.readAll();

items.append(response.split("\r\n"));

}

int pos = 0;

*for* (*auto* item: items){

pos++;

*if* (item != "." && item != ""){

QString tmp = getMessageInfo(pos);

infos.append(tmp);

QStringList lines = tmp.split("\n");

QString info = "";

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

*if* (lines[i].contains("Subject") || lines[i].contains("From")){

info += lines[i] + " ";

}

}

ui->mailList->addItem(info);

infos.append(tmp);

}

}

response = pop3socket.readAll();

}

QString MainWindow::**getMessageInfo**(int index){

QString request;

QString response;

currentMessage = index;

request = "RETR " + QString::number(index) + "\r\n";

pop3socket.write(request.toLocal8Bit());

pop3socket.*waitForBytesWritten*();

pop3socket.*waitForReadyRead*();

response = pop3socket.readAll();

QString message;

QTextStream message\_stream(&message);

int read = 0;

int all = response.split(" ")[1].toInt();

*while* (pop3socket.*waitForReadyRead*(1000)){

read += pop3socket.*bytesAvailable*();

response = pop3socket.readAll();

message\_stream << response;

}

*return* message;

}