

Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное
учреждение высшего образования
«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»



Кафедра теоретической и прикладной информатики

Лабораторная работа № 4
по дисциплине «Информационная безопасность»



ФАКУЛЬТЕТ:	ПМИ
ГРУППА:	ПМИ-61
СТУДЕНТЫ:	Ершов П.К., Мамонова Е.В., Цыденов З.Б.
ВАРИАНТ:	2
КОНТАКТНЫЙ АДРЕС	Ершов П.К. - PMI62.Ershov@outlook.com Мамонова Е.В. - PMI62.Mamonova@outlook.com Цыденов З.Б. - tsydebov.zanabazar@gmail.com
ПРЕПОДАВАТЕЛЬ:	Авдеев Т.В.

Новосибирск

2020

1. Цель работы

Ознакомиться с концепцией ролевого управления доступом и способами защиты программного обеспечения от существующих угроз. Научиться разрабатывать приложения, которые используют ролевое управление доступом для разграничения полномочий пользователей. Получить навыки защиты разработанной программы от несанкционированного копирования и других угроз, которым может подвергаться программное обеспечение.

2. Задание

- I. Реализовать приложение с графическим интерфейсом, удовлетворяющее следующим требованиям.
 1. Приложение проводит аутентификацию пользователя.
 2. Каждый пользователь программы должен относиться к какой-нибудь группе пользователей (роли), членам которой доступны различные функциональные возможности программы.
 3. Программа должна принимать от пользователя некоторые данные и, возможно, после некоторой обработки, отображать их. При этом должна осуществляться защита от возможных атак на приложение. При разработке защиты нужно предположить, что приложение работает с базой данных, в которой сохраняет введенные пользователем данные.
- II. Реализовать приложение-инсталлятор, позволяющее установить на компьютер пользователя приложение, реализованное в предыдущем пункте задания. Требования к приложению следующие.
 1. Приложение-инсталлятор совместно с устанавливаемым приложением должно обеспечивать защиту программного продукта от несанкционированного тиражирования.
 2. Приложение-инсталлятор должно иметь защиту от возможных атак на него.
- III. Протестировать правильность работы разработанных приложений.

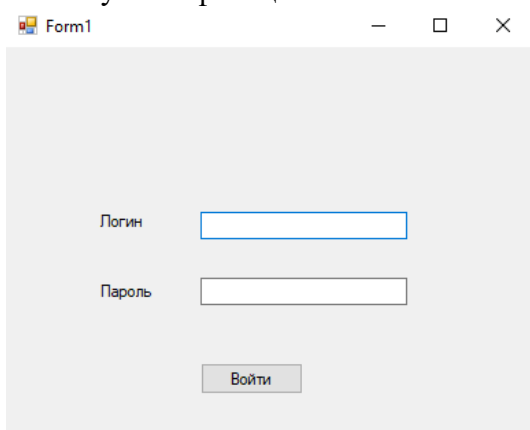
3. Описание разработанного программного средства

Для демонстрации концепции ролевого доступа было выбрано приложение, имитирующее терминал доступа к баз данных.

Приложение требует логин и пароль для входа. Пароль, для безопасности хэшируется с применением алгоритма SHA512, а затем сравнивается с хэшем пароля из базы данных.

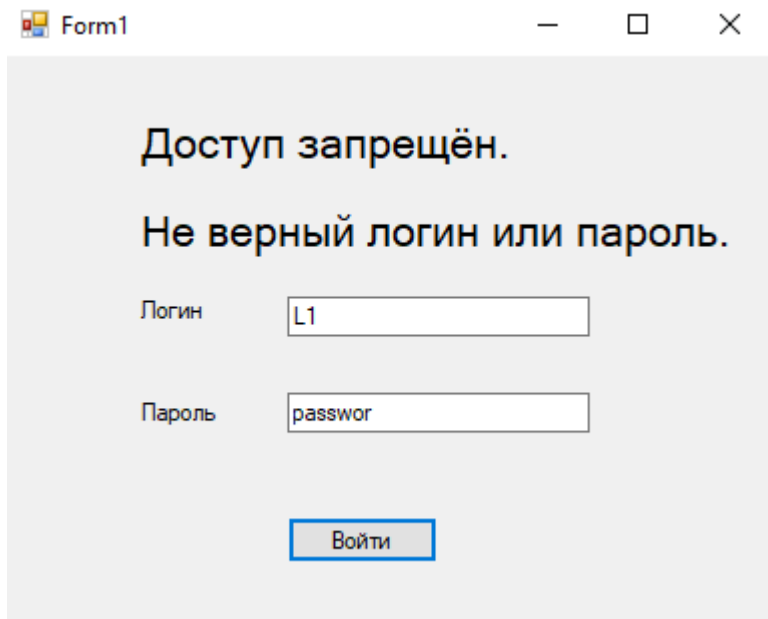
В случае несоответствия хэшей пароля, приложение запрещает доступ к основным функциям. Для безопасного доступа к базе данных используется динамический SQL. Для его реализации была выбрана библиотека Npgsql. Основная цель использования динамического SQL – защита от SQL инъекций.

Окно аутентификации:



The image shows a screenshot of a Windows application window titled "Form1". The window has a standard Windows title bar with minimize, maximize, and close buttons. The main area of the window is light gray and contains a login form. The form consists of two text input fields. The first field is labeled "Логин" (Login) and the second field is labeled "Пароль" (Password). Below these fields is a button labeled "Войти" (Login). The labels and the button are in a dark gray font.

Результат не корректного ввода пароля:



Form1

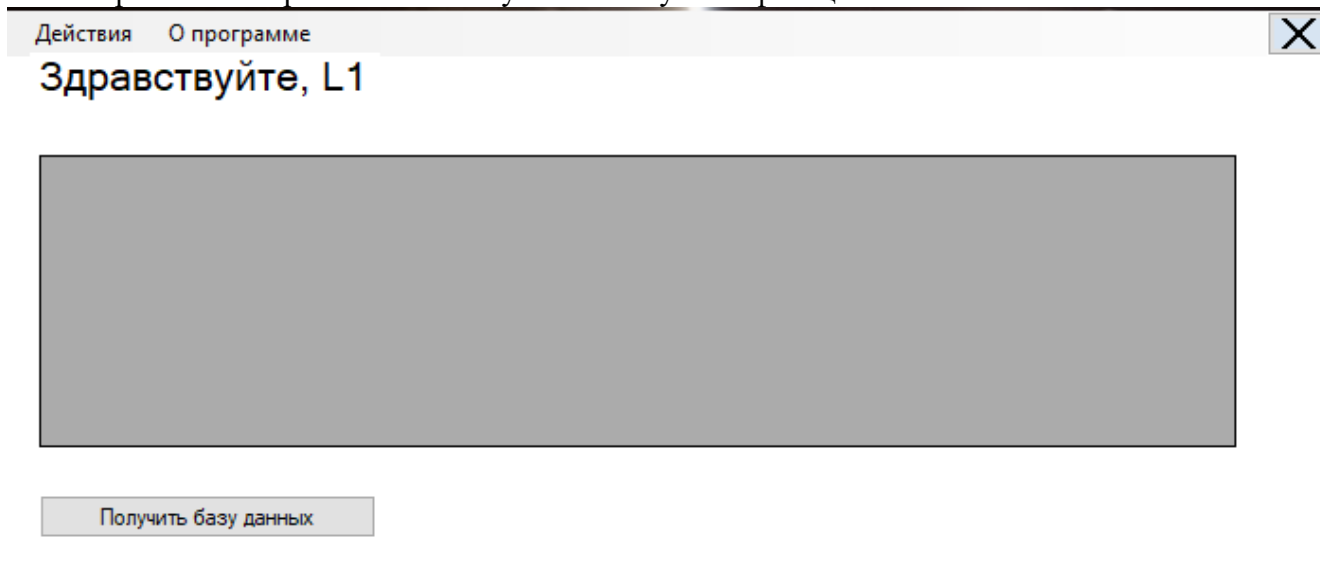
Доступ запрещён.

Не верный логин или пароль.

Логин

Пароль

Окно терминала открывается после успешной аутентификации.

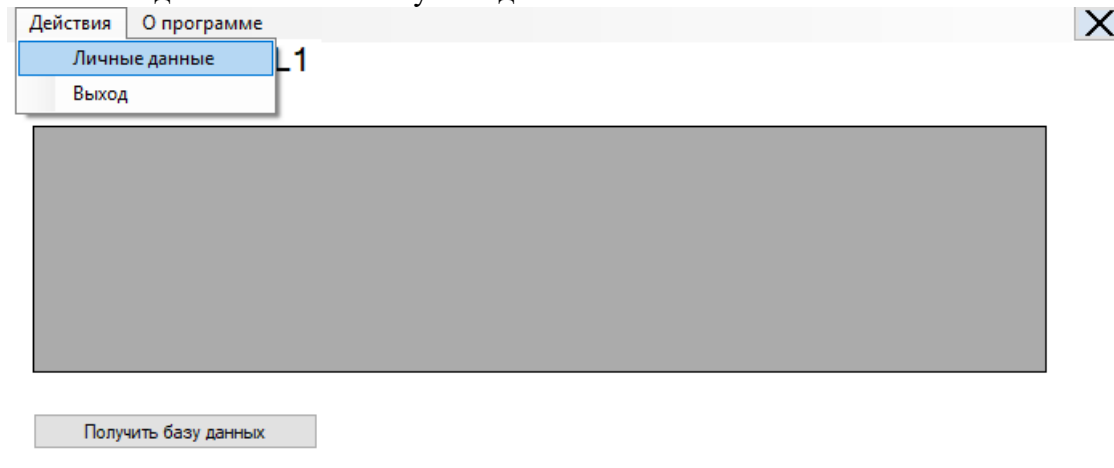


Действия О программе

Здравствуйте, L1

При этом, окно входа скрывается, чтобы избежать многократного входа.

В Личных данных можно получить данные о самом пользователе:

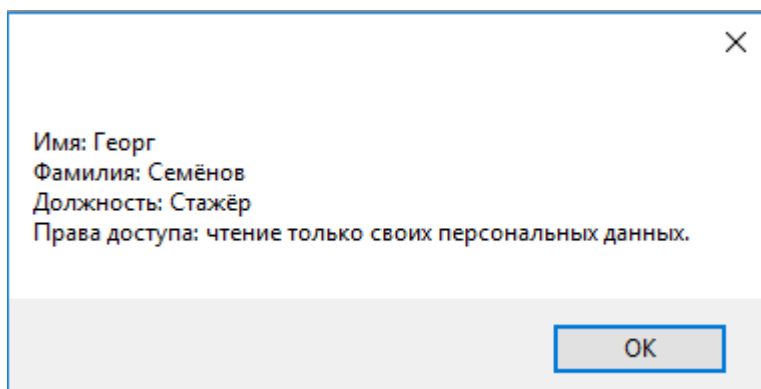


Действия О программе

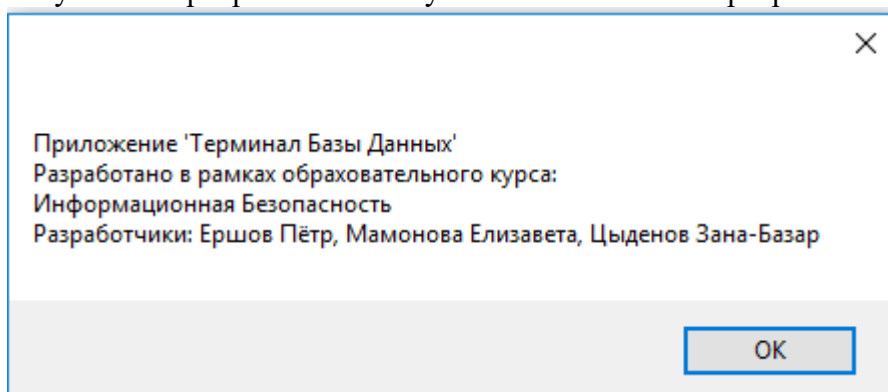
Личные данные

Выход

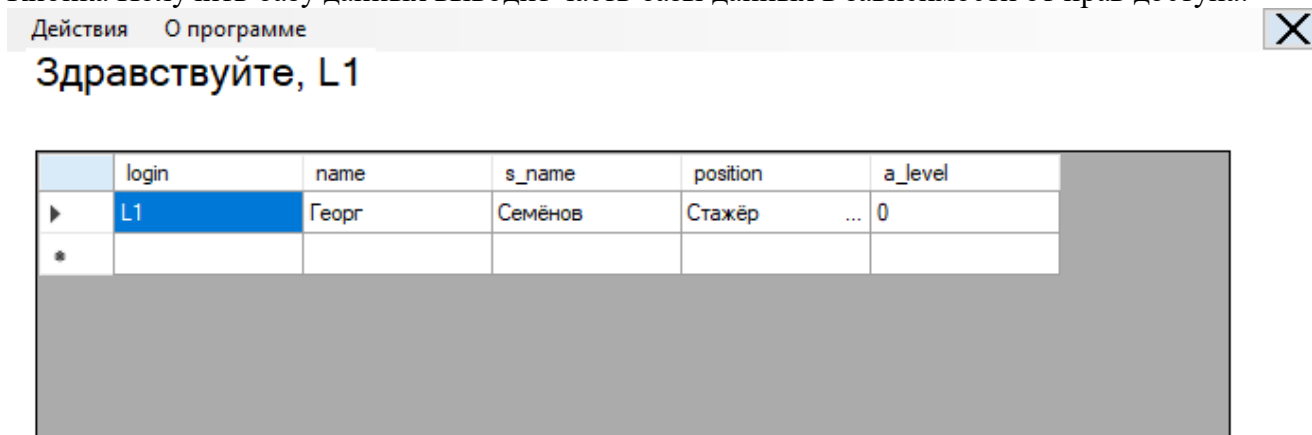
L1



В пункте О программе можно узнать о назначении программы:



Кнопка Получить базу данных выводит часть базы данных в зависимости от прав доступа:



В данном случае, уровень доступа разрешает только просмотр персональных данных.

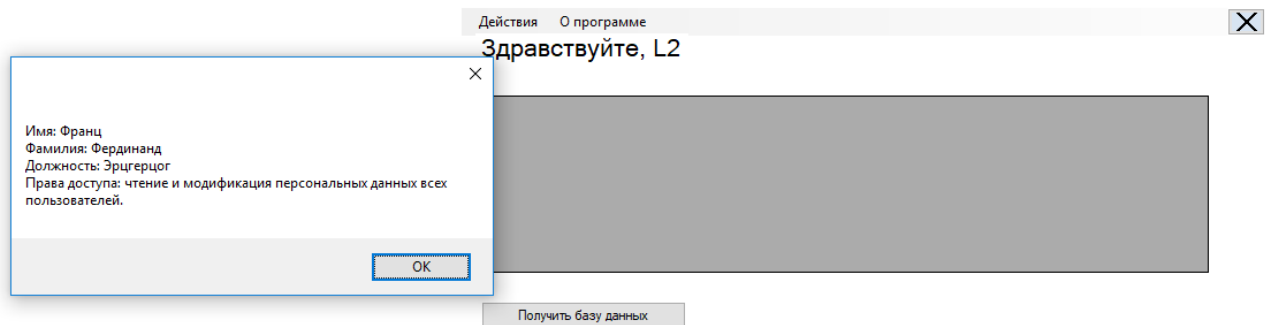
Всего в программе 3 уровня доступа:

Уровень доступа	Доступные права
0 (гостевой)	Только чтение персональной информации
1 (пользовательский)	Только чтение всей базы данных
2 (административный)	Чтение и модификация базы данных

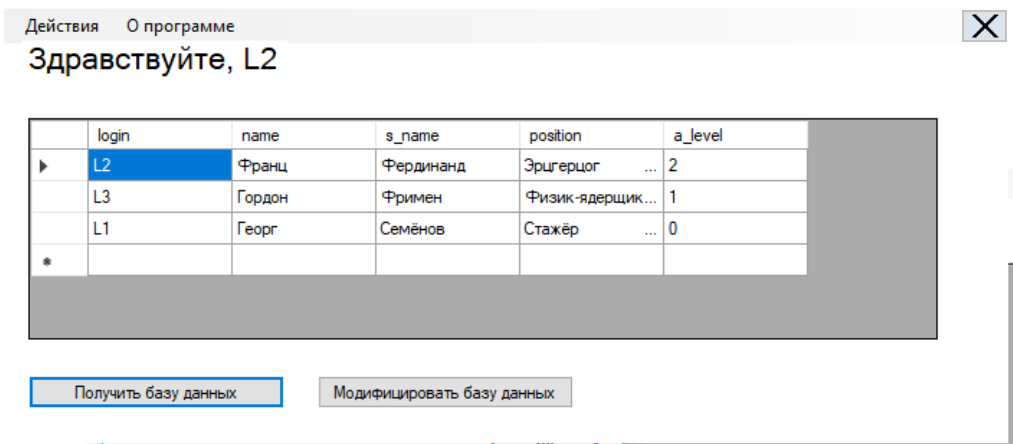
Демонстрация пользовательского и административного аккаунтов:

Пользователь:

Администратор:



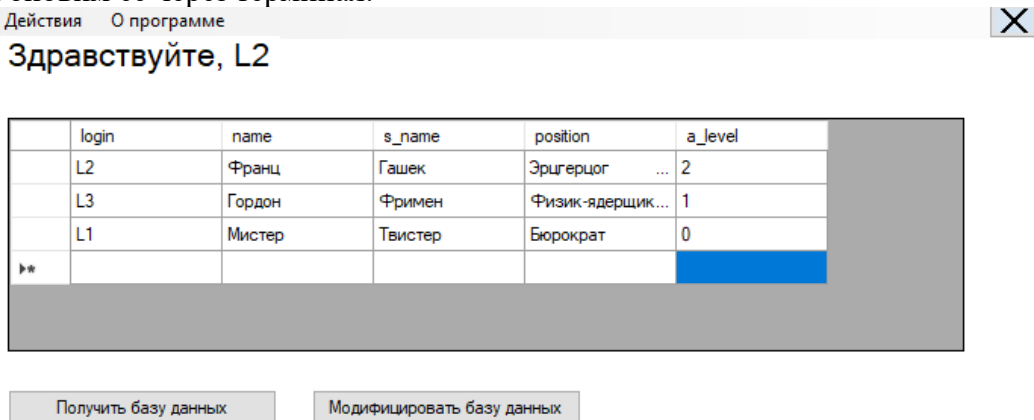
Если нажать Получить базу данных под администратором, то появиться кнопка Модифицировать базу данных. Она позволяет изменить данные в базе, путём модификации значений ячеек в окне терминала.



Изначальная база данных:

login	name	s_name	position	a_level
L2	Франц	Фердинанд	Эрцгерцог	2
L3	Гордон	Фримен	Физик-ядерщик	1
L1	Георг	Семёнов	Стажёр	0

Обновим её через терминал:



Обновлённая база данных:

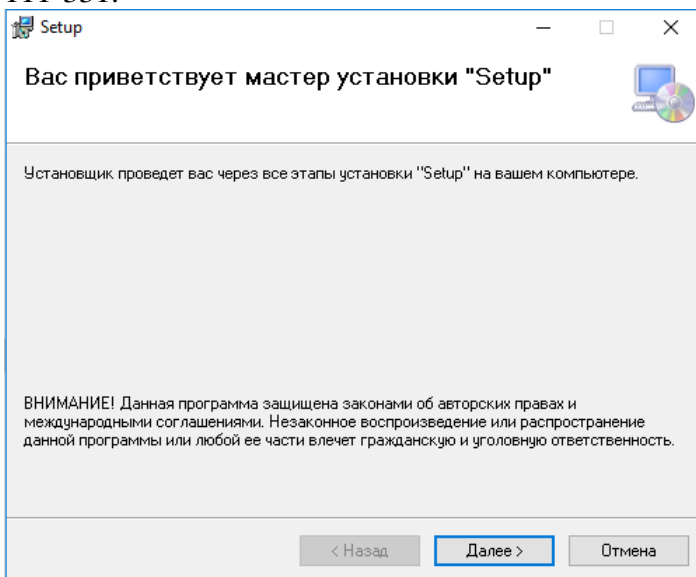
login	name	s_name	position	a_level
L3	Гордон	Фримен	Физик-ядерщик	1
L2	Франц	Гашек	Эрцгерцог	2
L1	Мистер	Твистер	Бюрократ	0

Также был разработан установщик, с применением ключа в качестве защиты от копирования.

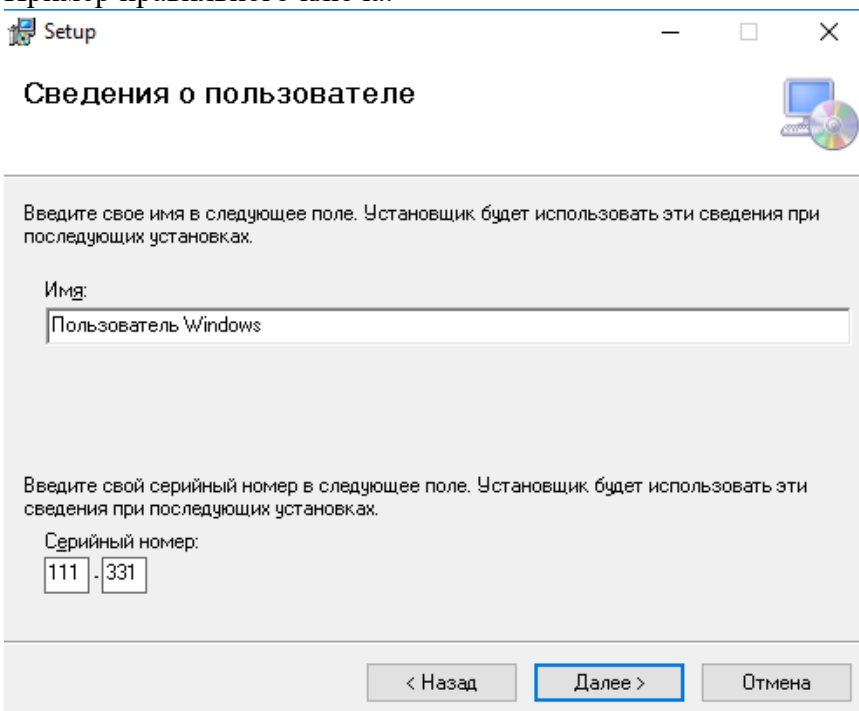
Ключом является код, состоящий из 6 цифр, сумма последних 3 цифр должна делиться на 7 без остатка.

Пример такого ключа:

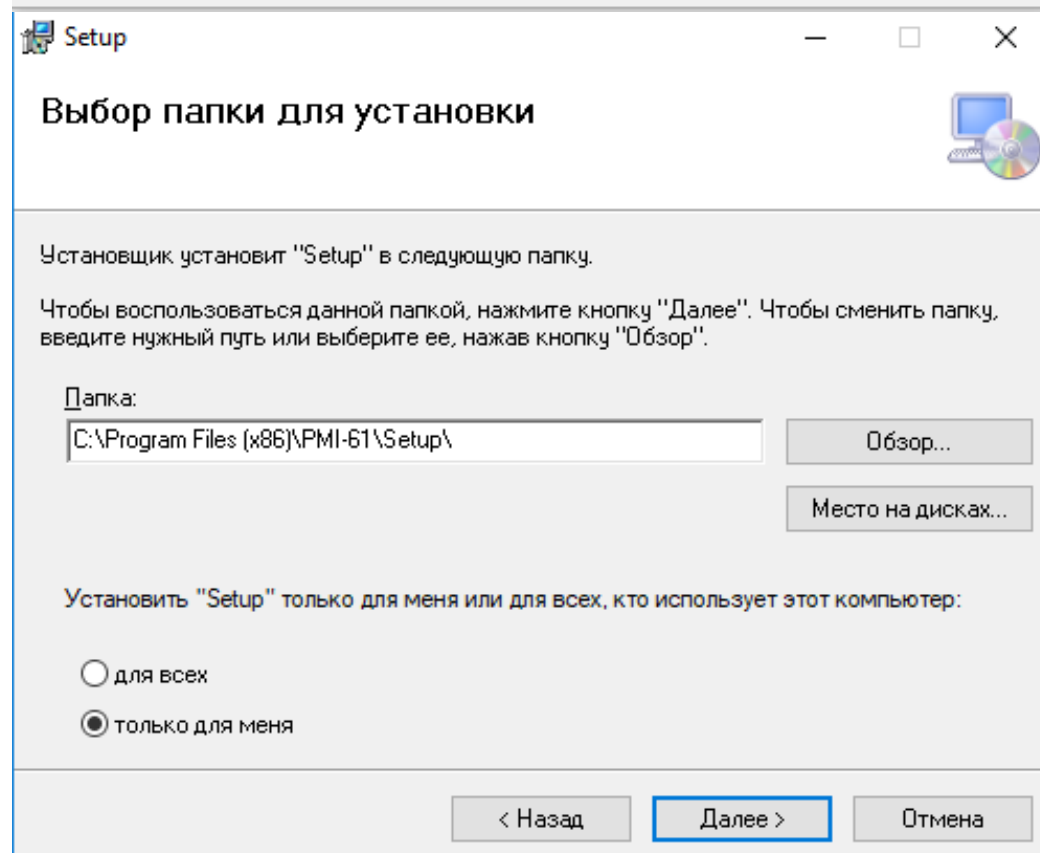
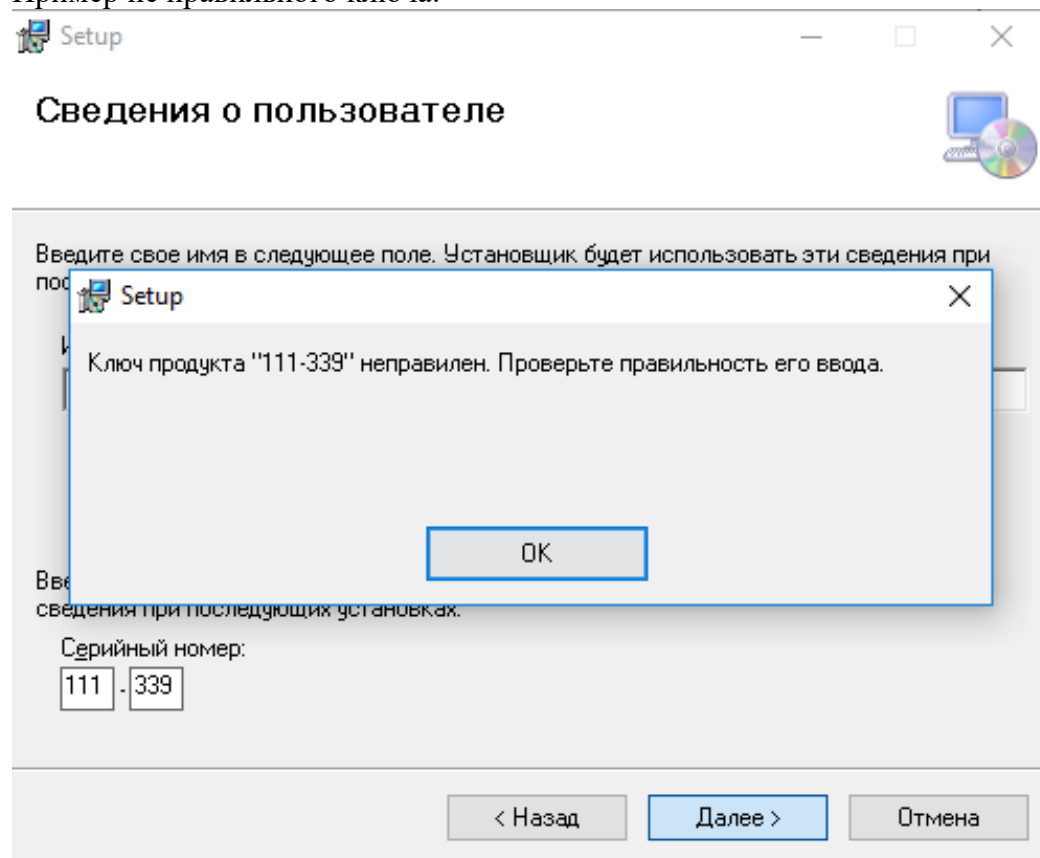
111-331.

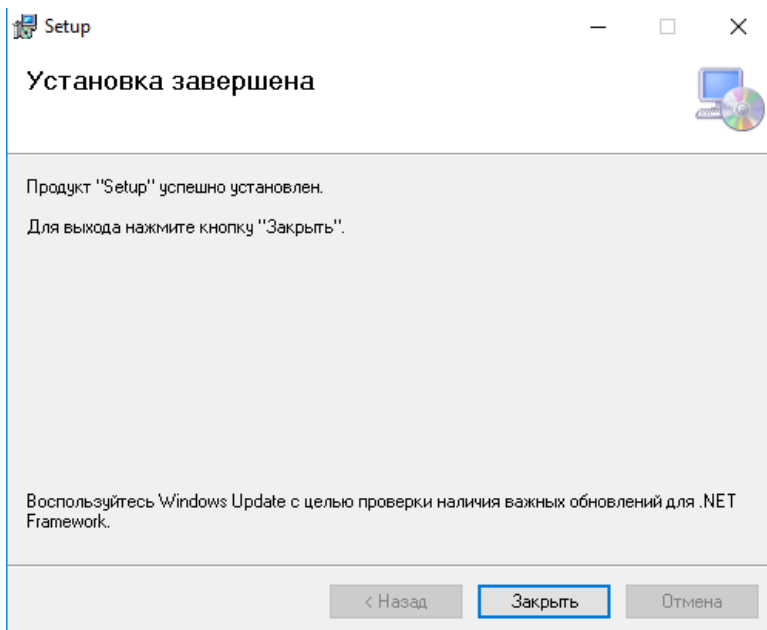


Пример правильного ключа:



Пример не правильного ключа:





Для запуска программы, необходимо перейти в папку C:\Program Files (x86)\PMI-61\Setup\ и запустить файл IS_lab4.exe.

4. Код программы

Crypto.cs

```
class Crypto
{
    public byte[] SHA_512(byte[] data) // генератор хэша на основе SHA512
    {
        byte[] res = new byte[1];

        SHA512 SHA = new SHA512Managed();

        res = SHA.ComputeHash(data);
        return res;
    }
}
```

DB_operation.cs

```
class DB_operation
{
    static public bool Authentication(string login, string password) // функция аутентификации пользователя
    {
        NpgsqlCommand db_op; // создаём экземпляр класса для выполнения SQL-запроса
        Crypto CR = new Crypto(); // экземпляр класса для шифрования
        var hash = CR.SHA_512(Encoding.Default.GetBytes(password)); // хеш введенного пароля, должен быть
        // равен тому, что в бд
        var con = new NpgsqlConnection(Param.DB_CONN_PARAMETERS); // задаём параметры соединения
        con.Open(); // соединяемся с БД

        db_op = new NpgsqlCommand("select pass_hash, a_level from pmib6702.passwords, pmib6702.login where
        passwords.n_pass = login.n_pass and login.login = (@login)", con); // динамический SQL
        db_op.Parameters.AddWithValue("login", login); // указываем параметр, по которому получаем данные из
        // запроса
        var reader = db_op.ExecuteReader(); // задаём экземпляр чтения данных

        if (!reader.Read())
            return false;

        var trueHash = reader.GetString(0); // берем строку из первого столбца результата
        var a_level = reader.GetInt32(1);
        con.Close(); // подключение больше не нужно

        string s_hash = "";
        for (int i = 0; i < hash.Length; i++)
```



```

        s_hash += hash[i].ToString("x");
trueHash = trueHash.Trim(' ');

for (int i = 0; i < trueHash.Length; i++) // сравниваем пароли
{
    if (trueHash[i] != s_hash[i]) return false;
}
Work_sess.Access = (Param.Access_level)a_level; // задаём уровень доступа
Work_sess.login = login; // задаём логин
return true;
}

static public bool Pesonal_data() // функция получения персональных данных
{
    NpgsqlCommand db_op;
    var con = new NpgsqlConnection(Param.DB_CONN_PARAMETERS);
    con.Open();

    db_op = new NpgsqlCommand(Param.DB_USR_L0, con); // динамический SQL
    db_op.Parameters.AddWithValue("login", Work_sess.login);
    var reader = db_op.ExecuteReader();

    if (!reader.Read())
        return false;

    Work_sess.name = reader.GetString(1); // сохраняем имя пользователя
    Work_sess.s_name = reader.GetString(2); // сохраняем фамилию пользователя
    Work_sess.pos = reader.GetString(3); // сохраняем должность пользователя
    con.Close();

    return true;
}

static public NpgsqlDataAdapter show_db() // получение всей базы данных
{
    NpgsqlDataAdapter db_op;
    string DB_type = "";

    var con = new NpgsqlConnection(Param.DB_CONN_PARAMETERS);
    con.Open();

    switch (Work_sess.Access) // выбираем тип запроса в зависимости уровня доступа
    {
        case Param.Access_level.l_0: // гость может просматривать только свои данные
            DB_type = Param.DB_USR_L0;
            break;
        case Param.Access_level.l_1: // пользователь и администратор могут просматривать всю БД
        case Param.Access_level.l_2:
            DB_type = Param.DB_USR_L1_2;
            break;
    }

    db_op = new NpgsqlDataAdapter(DB_type, con);
    switch (Work_sess.Access)
    {
        case Param.Access_level.l_0:
            db_op.SelectCommand.Parameters.AddWithValue("login", Work_sess.login);
            break;
    }
    con.Close();
    return db_op;
}
}
}

```

Param.cs

```

public static class Param
{
    static public string DB_CONN_PARAMETERS = "Host=students.ami.nstu.ru;Port=5432;Database=students;User Id=pmi-
b6702;Password=Etolys2;"; //параметр подключения к базе данных
    static public string DB_USR_L0 = "select * from pmib6702.personal_file where login = (@login)"; //запрос на получение
    всей базы данных для конкретного пользователя
    static public string DB_USR_L1_2 = "select * from pmib6702.personal_file"; //запрос на получение всей базы данных
    public enum Access_level { l_0, l_1, l_2 }; // тип данных для уровней допуска

    static public Form1 F1; // общий параметр для формы (необходим для блокировки формы, чтобы избежать двойного входа)
}

```

Work_sess.cs

```
public static class Work_sess
{
    public static string login; //логин
    public static string name; //имя
    public static string s_name; //фамилия
    public static string pos; //должность
    public static Param.Access_level Access; //уровень доступа
}
```

Form1.cs

```
public partial class Form1 : Form
{
    public Form1()
    {
        InitializeComponent();
    }
    string Log = "";
    private void Form1_Load(object sender, EventArgs e)
    {
        textBox2.PasswordChar = '*';
    }
    public string A_level_name(int i)
    {
        string s = "";
        if (i == 0)
            s = "гость";
        if (i == 1)
            s = "пользователь";
        if (i == 2)
            s = "администратор";

        return s;
    }
    private void Button1_Click_1(object sender, EventArgs e)
    {
        Log = textBox1.Text;
        if (Log == "")
            MessageBox.Show("Введите логин.");
        else
        {
            if (textBox2.Text == "")
                MessageBox.Show("Введите пароль.");
            else
            {
                if (DB_operation.Authentication(Log, textBox2.Text))
                {
                    label3.Text = "Добро пожаловать " + A_level_name(Convert.ToInt32(Work_sess.Access)) + " " + Log;

                    Form2 examp = new Form2();
                    examp.Show();
                    Param.F1 = this;
                    Visible = false;
                }
                else
                {
                    label3.Text = "Доступ запрещён.";
                    label4.Text = "Не верный логин или пароль.";
                }
            }
        }
    }
}
```

Pers_file.cs

```
public partial class Form2 : Form
{
    DataSet ds = new DataSet();
    DataTable dt = new DataTable();
    public Form2()
    {
```

```

        InitializeComponent();
    }

    private void Form2_MouseDown(object sender, MouseEventArgs e) // событие для перемещения формы
    {
        base.Capture = false;
        Message m = Message.Create(base.Handle, 0xa1, new IntPtr(2), IntPtr.Zero);
        this.WndProc(ref m);
    }

    private void Form2_Load(object sender, EventArgs e)
    {
        label1.Text = "Здравствуйте, " + Work_sess.login;
        button3.Visible = false; // блокировка кнопки обновления базы данных для гостя и пользователя
    }

    private void Button1_Click(object sender, EventArgs e) // получение базы данных
    {
        var db_op = DB_operation.show_db();
        ds.Reset();
        db_op.Fill(ds);
        dt = ds.Tables[0];
        dataGridView1.DataSource = dt; // вывод таблицы

        switch (Work_sess.Access)
        {
            case Param.Access_level.l_2:
                button3.Visible = true; // активация кнопки для обновления БД для администратора
                break;
        }
    }

    private void Button3_Click(object sender, EventArgs e) // обновление базы данных
    {
        NpgsqlDataAdapter db_op;

        var con = new NpgsqlConnection(Param.DB_CONN_PARAMETERS);
        con.Open();

        db_op = new NpgsqlDataAdapter(Param.DB_USR_L1_2, con);

        NpgsqlCommandBuilder cmd = new NpgsqlCommandBuilder(db_op);
        db_op.UpdateCommand = cmd.GetUpdateCommand();
        db_op.Update((DataTable) dataGridView1.DataSource); // выполняем обновление БД

        con.Close();
    }

    private void ЛичныеДанныеToolStripMenuItem_Click(object sender, EventArgs e) // вывод личных данных
    {
        string a_1 = "";
        switch (Work_sess.Access)
        {
            case Param.Access_level.l_0:
                a_1 = "чтение только своих персональных данных.";
                break;
            case Param.Access_level.l_1:
                a_1 = "чтение персональных данных всех пользователей.";
                break;
            case Param.Access_level.l_2:
                a_1 = "чтение и модификация персональных данных всех пользователей.";
                break;
        }
        DB_operation.Pesonal_data();
        MessageBox.Show("Имя: " + Work_sess.name + "\n" +
            "Фамилия: " + Work_sess.s_name + "\n" +
            "Должность: " + Work_sess.pos + "\n" +
            "Права доступа: " + a_1);
    }

    private void ВыходToolStripMenuItem_Click(object sender, EventArgs e) // выход из программы
    {
        Work_sess.login = "";
        Param.F1.Visible = true;
        Close();
    }

    private void ОПрограммеToolStripMenuItem_Click(object sender, EventArgs e)
    {
        MessageBox.Show("Приложение 'Терминал Базы Данных'\n" +
            "Разработано в рамках образовательного курса:\n" +
            "Информационная Безопасность\n" +
            "Разработчики: Ершов Пётр, Мамонова Елизавета, Цыденов Зана-Базар");
    }

    private void menuStrip1_MouseDown(object sender, MouseEventArgs e)
    {
        base.Capture = false;
        Message m = Message.Create(base.Handle, 0xa1, new IntPtr(2), IntPtr.Zero);
        this.WndProc(ref m);
    }

```

```

private void Button2_Click(object sender, EventArgs e)
{
    Work_sess.login = "";
    Param.F1.Visible = true;
    Close();
}
}

```

5. Выводы

В ходе выполненной лабораторной было разработано программное средство для демонстрации ролевого доступа на примере базы данных. В приложении были применены средства защиты информации, такие, как:

1. Хэширование паролей.
2. Динамический SQL.
3. Программное ограничение доступа к базе данных.
4. Использование ключа в установщике с целью ограничения копирования программы.

Приложение 1.

Логин	Пароль
L1	password
L2	Port_test_987456321
L3	123456789