

Федеральное агентство связи
Ордена трудового Красного Знамени федеральное государственное
бюджетное
образовательное учреждение высшего образования
«Московский технический университет связи и информатики»

Отчет по лабораторной работе
«МЕТОДЫ ПАРОЛЬНОЙ ЗАЩИТЫ. РАЗРАБОТКА ПРОГРАММНОЙ
ПАРОЛЬНОЙ ЗАЩИТЫ»
по дисциплине «Информационная безопасность»
Вариант 1

Выполнил: студент группы БВТ1904

Абакаров Гасан Гаджирабаданович

Проверила:

Магомедова Дженнет Исламутдиновна

Москва, 2020

Цель работы: изучение технологии аутентификации пользователя на основе пароля.

Ход работы:

1. Ознакомиться с теоретической частью данной работы.
2. Составить программу, представляющую собой форму доступа к определённым информационным ресурсам на основе пароля.
3. Составить отчет по проделанной работе.
4. Защитить работу.

Постановка задачи:

Разработать программу, представляющую собой форму доступа к определённым информационным ресурсам на основе пароля.

- В качестве информационного ресурса использовать любой файл или приложение.

- Доступ к ресурсу должен быть разрешен только санкционированным пользователям. Для этого в программе должны храниться имена пользователей и их пароли. При попытке доступа пользователя к ресурсу проверяется наличие его идентификатора ID (имени) в системе и соответствие введенного пароля паролю, который хранится в системе.

- В системе должна храниться следующая информация о пользователе: ID или имя пользователя, пароль, ФИО, дата рождения, место рождения (город), номер телефона.

- Пользователь должен иметь возможность поменять пароль

- Длина пароля (количество символов): 6, Используемые символы: Латиница (строчные буквы), Дополнительные средства защиты: при смене пароля: проверка на отсутствие повторяющихся символов

Листинг программы:

```
#программа составлена на языке python3.9
#(https://www.python.org/downloads/release/python-390/)
#в ходе создания программы был использован
#генератор форм page (http://page.sourceforge.net/)
#использовались библиотека для работы с базой данных sqlite
#и библиотека для работы с оконным интерфейсом tkinter

table = '''CREATE TABLE table_name(id text primary key, pin text, surname
text, name text, patronymic text, bright_data text, bright_place text, phone
text)'''

import sys, os
import time
import sqlite3 as sq
import tkinter as tk
import tkinter.ttk as ttk
from tkinter import messagebox
from sqlite3 import OperationalError as o_err
from sqlite3 import IntegrityError as i_err
con=sq.connect('users.db')
cur=con.cursor()
cur.execute('''select * from sqlite_master where name = "table_name"''')
f=cur.fetchall()

if len(f)>0:
    f=f[0][4]
    if f!=table:
        cur.execute('DROP table table_name')
        cur.execute(table)
    else:
        cur.execute(table)

del f

con.commit()

abc='abcdefghijklmnopqrstuvwxyz'
```

```

def callback(text):
    p=''
    for i in text.get().lower():
        if i in abc:
            p+=i
    if len(p)>6:
        p=p[:6]
    text.set(p)

class database():
    def __getitem__(a, b):
        cur.execute("select * from table_name where id = "+repr(b))
        j=cur.fetchall()
        if len(j)!=0:
            return j[0]
        else:
            return []

    def add(a, c):
        try:
            cur.execute('insert into table_name values'+repr(c))
        except i_err:
            return []
        con.commit()
        return c

    def __delitem__(a, b):
        cur.execute('delete from table_name where id = '+repr(b))
        con.commit()

database = database()

exists=os.path.exists
toast=messagebox.showinfo

def delete(top):

```

```

        b = top.b

        if messagebox.askyesno('', 'Вы уверены, что хотите удалить
аккаунт?')==False:

            return

        database.__delitem__(b[0])

        top.toplevel.destroy()

def update(user, top):
    b=database[user]

    ij=''

    ij+='Имя пользователя: '+b[0]+'\\nФамилия: '+b[2]+'\\nИмя: '+b[3]+'\\
nОтчество: '+b[4]+'\\nДата рождения: '+b[5]+'\\nМесто рождения: '+b[6]+'\\
nТелефон: '+b[7]

    top.input.delete('0.0', '10000000000000000000.1000000000000000')

    top.input.insert(tk.INSERT, ij)

def change_password(top):
    l=window_2(tk.Toplevel(root))

    l.toplevel.title('Обновить данные')

    l.Entry1.bind("<Key>", lambda e: ctrlEvent(e))

    user = l.user = top.b[0]

    l.password = top.b[1]

    b=top.b

    l.Entry1.insert('0',b[0])
    l.Entry2.insert('0',b[1])
    l.Entry3.insert('0',b[2])
    l.Entry4.insert('0',b[3])
    l.Entry5.insert('0',b[4])
    l.Entry6.insert('0',b[5])
    l.Entry7.insert('0',b[6])
    l.Entry8.insert('0',b[7])

    l.postcode = lambda *a: update(user, top)

def save(top, user='', password='', post=None):
    a1=top.Entry1.get()
    a2=top.code_text.get()
    a3=top.Entry3.get()
    a4=top.Entry4.get()

```

```

a5=top.Entry5.get()
a6=top.Entry6.get()
a7=top.Entry7.get()
a8=top.Entry8.get()
if a1=='':
    toast('', 'Поле "Имя пользователя" является обязательной для
заполнения')
    return
elif a2=='':
    toast('', 'Поле "Пароль" является обязательной для заполнения')
    return
elif len(a2)!=6:
    toast('', 'Пароль не может быть короче 6 символов')
    return
else:
    if password == a2:
        toast('', 'Вы уже использовали этот пароль, введите другой')
        return
    if user == a1:
        database.__delitem__(a1)
        k=database.add((a1, a2, a3, a4, a5, a6, a7, a8))
    if not k:
        toast('', 'Пользователь с именем '+repr(a1)+' уже существует')
        return
con.commit()
top.toplevel.destroy()
if callable(post):
    post()
return

def login(top):
    a1=top.Entry1.get()
    a2=top.Entry2.get()
    if a1=='':
        toast('', 'Поле "Имя пользователя" является обязательной для
заполнения')
    elif a2=='':
        toast('', 'Поле "Пароль" является обязательной для заполнения')

```

```

else:
    time.sleep(1)
    b=database[a1]
    if len(b)==0:
        toast('', 'Пользователь с именем '+repr(a1)+' не существует, или
        пароль неверный')
    elif b[1]!=a2:
        toast('', 'Пользователь с именем '+repr(a1)+' не существует, или
        пароль неверный')
    else:
        l=window_3(tk.Toplevel(root))
        ij=''
        ij+='Имя пользователя: '+b[0]+'\\nФамилия: '+b[2]+'\\nИмя:
        '+b[3]+'\\nОтчество: '+b[4]+'\\nДата рождения: '+b[5]+'\\nМесто рождения:
        '+b[6]+'\\nТелефон: '+b[7]
        l.input.insert(tk.INSERT,ij)
        l.b=b
        #      print(b)

```

```

class window_3:
    def __init__(self, top=None):
        '''This class configures and populates the toplevel window.
        top is the toplevel containing window.'''
        _bgcolor = '#d9d9d9' # X11 color: 'gray85'
        _fgcolor = '#000000' # X11 color: 'black'
        _compcolor = '#d9d9d9' # X11 color: 'gray85'
        _ana1color = '#d9d9d9' # X11 color: 'gray85'
        _ana2color = '#ececec' # Closest X11 color: 'gray92'

        self.toplevel = top

        top.geometry("600x450+62+444")
        top.minsize(1, 1)
        top.maxsize(1905, 1050)
        top.resizable(1, 1)
        top.title("New Toplevel")
        top.configure(highlightcolor="black")

```

```

self.Button1 = tk.Button(top)
self.Button1.place(relx=1.2, rely=0.711, height=38, width=205)
self.Button1.configure(activebackground="#f9f9f9")
self.Button1.configure(font="-family {DejaVu Sans} -size 10 -weight
normal -slant roman -underline 0 -overstrike 0")
self.Button1.configure(text='''удалить аккаунт''')

self.menubar = tk.Menu(top, font="TkMenuFont", bg=_bgcolor, fg=_fgcolor)
top.configure(menu = self.menubar)

self.Label1 = tk.Label(top)
self.Label1.place(relx=0.017, rely=0.022, height=28, width=134)
self.Label1.configure(activebackground="#f9f9f9")
self.Label1.configure(font="-family {DejaVu Sans} -size 12 -weight
normal -slant roman -underline 0 -overstrike 0")
self.Label1.configure(text='''Ваши данные''')

self.input = tk.Text(top, )
self.input.place(relx=0.033, rely=0.089, relheight=0.782,
relwidth=0.91)
self.input.configure(background="white")
self.input.configure(font="-family {DejaVu Sans} -size 12")
self.input.configure(selectbackground="blue")
self.input.configure(selectforeground="white")
self.input.configure(wrap="word")
self.input.bind("<Key>", lambda e: ctrlEvent(e))

self.Button2 = tk.Button(top)
self.Button2.place(relx=0.033, rely=0.889, height=38, width=165)
self.Button2.configure(activebackground="#f9f9f9")
self.Button2.configure(command=lambda: delete(self))
self.Button2.configure(font="-family {DejaVu Sans} -size 10 -weight
normal -slant roman -underline 0 -overstrike 0")
self.Button2.configure(text='''удалить аккаунт''')

self.Button3 = tk.Button(top)
self.Button3.place(relx=0.333, rely=0.889, height=38, width=165)
self.Button3.configure(activebackground="#f9f9f9")
self.Button3.configure(command=lambda: change_password(self))

```



```

        self.Button3.configure(font="-family {DejaVu Sans} -size 10 -weight
normal -slant roman -underline 0 -overstrike 0")
        self.Button3.configure(text='''обновить данные''')

def ctrlEvent(e):
    if (e.state == 20 and e.keysym == 'c'): return
    return "break"

class window_1:
    def __init__(self, top=None):
        '''This class configures and populates the toplevel window.
        top is the toplevel containing window.'''
        _bgcolor = '#d9d9d9' # X11 color: 'gray85'
        _fgcolor = '#000000' # X11 color: 'black'
        _compcolor = '#d9d9d9' # X11 color: 'gray85'
        _analogcolor = '#d9d9d9' # X11 color: 'gray85'
        _ana2color = '#ecec' # Closest X11 color: 'gray92'
        font9 = "-family {DejaVu Sans} -size 12"

        self.toplevel = top

        top.geometry("284x353+59+554")
        top.minsize(1, 1)
        top.maxsize(1905, 1050)
        top.resizable(1, 1)
        top.title("Вход")

        self.Label1 = tk.Label(top)
        self.Label1.place(relx=0.211, rely=0.057, height=33, width=165)
        self.Label1.configure(activebackground="#f9f9f9")
        self.Label1.configure(font="-family {DejaVu Sans} -size 12 -weight
normal -slant roman -underline 0 -overstrike 0")
        self.Label1.configure(text='''Имя пользователя''')

        self.Entry1 = tk.Entry(top)
        self.Entry1.place(relx=0.07, rely=0.142, height=30, relwidth=0.866)
        self.Entry1.configure(background="white")
        self.Entry1.configure(font="TkFixedFont")

```

```

self.Label2 = tk.Label(top)
self.Label2.place(relx=0.211, rely=0.283, height=33, width=165)
self.Label2.configure(activebackground="#f9f9f9")
self.Label2.configure(font=font9)
self.Label2.configure(text='''Пароль''')

self.password = tk.StringVar()
self.password.trace("w", lambda *a: callback(self.password))

self.Entry2 = tk.Entry(top, textvariable=self.password, show='*')
self.Entry2.place(relx=0.07, rely=0.368, height=30, relwidth=0.866)
self.Entry2.configure(background="white")
self.Entry2.configure(font="TkFixedFont")

self.Button1 = tk.Button(top)
self.Button1.place(relx=0.07, rely=0.538, height=38, width=245)
self.Button1.configure(font=font9, command=lambda: login(self))
self.Button1.configure(text='''Войти''')

self.Button2 = tk.Button(top)
self.Button2.place(relx=0.07, rely=0.793, height=38, width=245)
self.Button2.configure(activebackground="#f9f9f9")
self.Button2.configure(font="-family {DejaVu Sans} -size 12 -weight
normal -slant roman -underline 0 -overstrike 0")
self.Button2.configure(text='''Регистрация''', command=registration )

def registration():
    window_2(tk.Toplevel(root))

class window_2:
    def __init__(self, top=None):
        '''This class configures and populates the toplevel window.
            top is the toplevel containing window.'''
        _bgcolor = '#d9d9d9' # X11 color: 'gray85'
        _fgcolor = '#000000' # X11 color: 'black'
        _compcolor = '#d9d9d9' # X11 color: 'gray85'
        _analogcolor = '#d9d9d9' # X11 color: 'gray85'

```

```

_ana2color = '#ececec' # Closest X11 color: 'gray92'
font9 = "-family {Liberation Sans} -size 13"
self.postcode = None
self.password = ''
self.toplevel = top
self.user=''
top.geometry("412x490+489+465")
top.minsize(1, 1)
top.maxsize(1905, 1050)
top.resizable(1, 1)
top.title("Регистрация")
top.configure(highlightcolor="black")

self.Label1 = tk.Label(top)
self.Label1.place(relx=0.218, rely=0.122, height=28, width=84)
self.Label1.configure(activebackground="#f9f9f9")
self.Label1.configure(font="-family {Liberation Sans} -size 13 -
weight normal -slant roman -underline 0 -overstrike 0")
self.Label1.configure(text='' 'Пароль' '')

self.Label2 = tk.Label(top)
self.Label2.place(relx=0.024, rely=0.041, height=28, width=164)
self.Label2.configure(activebackground="#f9f9f9")
self.Label2.configure(font="-family {Liberation Sans} -size 13 -
weight normal -slant roman -underline 0 -overstrike 0")
self.Label2.configure(text='' 'Имя пользователя' '')

self.Label3 = tk.Label(top)
self.Label3.place(relx=0.194, rely=0.265, height=28, width=94)
self.Label3.configure(activebackground="#f9f9f9")
self.Label3.configure(font="-family {Liberation Sans} -size 13 -
weight normal -slant roman -underline 0 -overstrike 0")
self.Label3.configure(text='' 'Фамилия' '')

self.Label4 = tk.Label(top)
self.Label4.place(relx=0.267, rely=0.347, height=28, width=64)
self.Label4.configure(activebackground="#f9f9f9")

```

```

        self.Label4.configure(font="-family {Liberation Sans} -size 13 -
weight normal -slant roman -underline 0 -overstrike 0")
        self.Label4.configure(text='''Имя''')

        self.Label5 = tk.Label(top)
        self.Label5.place(relx=0.194, rely=0.429, height=28, width=94)
        self.Label5.configure(activebackground="#f9f9f9")
        self.Label5.configure(font="-family {Liberation Sans} -size 13 -
weight normal -slant roman -underline 0 -overstrike 0")
        self.Label5.configure(text='''Отчество''')

        self.Label6 = tk.Label(top)
        self.Label6.place(relx=0.097, rely=0.571, height=28, width=134)
        self.Label6.configure(activebackground="#f9f9f9")
        self.Label6.configure(font="-family {Liberation Sans} -size 13 -
weight normal -slant roman -underline 0 -overstrike 0")
        self.Label6.configure(text='''Дата рождения''')

        self.Label7 = tk.Label(top)
        self.Label7.place(relx=0.073, rely=0.653, height=28, width=144)
        self.Label7.configure(activebackground="#f9f9f9")
        self.Label7.configure(font="-family {Liberation Sans} -size 13 -
weight normal -slant roman -underline 0 -overstrike 0")
        self.Label7.configure(text='''Место рождения''')

        self.Label8 = tk.Label(top)
        self.Label8.place(relx=0.194, rely=0.735, height=28, width=94)
        self.Label8.configure(activebackground="#f9f9f9")
        self.Label8.configure(font="-family {Liberation Sans} -size 13 -
weight normal -slant roman -underline 0 -overstrike 0")
        self.Label8.configure(text='''Телефон''')

        self.Button1 = tk.Button(top)
        self.Button1.place(relx=0.097, rely=0.857, height=38, width=335)
        self.Button1.configure(activebackground="#f9f9f9")
        self.Button1.configure(font="-family {Liberation Sans} -size 13 -
weight normal -slant roman -underline 0 -overstrike 0")
        self.Button1.configure(text='''Сохранить''', command=lambda:
save(self, self.user, self.password, self.postcode))

```

```

self.Entry1 = tk.Entry(top)
self.Entry1.place(relx=0.437, rely=0.041,height=30, relwidth=0.5)
self.Entry1.configure(background="white")
self.Entry1.configure(font=font9)

self.code_text = tk.StringVar()
self.code_text.trace("w", lambda *a: callback(self.code_text))

self.Entry2 = tk.Entry(top, show='*', textvariable=self.code_text)
self.Entry2.place(relx=0.437, rely=0.122,height=30, relwidth=0.5)
self.Entry2.configure(background="white")
self.Entry2.configure(font="-family {Liberation Sans} -size 13 -
weight normal -slant roman -underline 0 -overstrike 0")
self.Entry2.configure(selectbackground="blue")
self.Entry2.configure(selectforeground="white")

self.Entry3 = tk.Entry(top)
self.Entry3.place(relx=0.437, rely=0.265,height=30, relwidth=0.5)
self.Entry3.configure(background="white")
self.Entry3.configure(font="-family {Liberation Sans} -size 13 -
weight normal -slant roman -underline 0 -overstrike 0")
self.Entry3.configure(selectbackground="blue")
self.Entry3.configure(selectforeground="white")

self.Entry4 = tk.Entry(top)
self.Entry4.place(relx=0.437, rely=0.347,height=30, relwidth=0.5)
self.Entry4.configure(background="white")
self.Entry4.configure(font="-family {Liberation Sans} -size 13 -
weight normal -slant roman -underline 0 -overstrike 0")
self.Entry4.configure(selectbackground="blue")
self.Entry4.configure(selectforeground="white")

self.Entry5 = tk.Entry(top)
self.Entry5.place(relx=0.437, rely=0.429,height=30, relwidth=0.5)
self.Entry5.configure(background="white")
self.Entry5.configure(font="-family {Liberation Sans} -size 13 -
weight normal -slant roman -underline 0 -overstrike 0")
self.Entry5.configure(selectbackground="blue")
self.Entry5.configure(selectforeground="white")

```

```

self.menubar = tk.Menu(top,font="TkMenuFont",bg=_bgcolor,fg=_fgcolor)
top.configure(menu = self.menubar)

self.Entry6 = tk.Entry(top)
self.Entry6.place(relx=0.437, rely=0.571,height=30, relwidth=0.5)
self.Entry6.configure(background="white")
self.Entry6.configure(font="-family {Liberation Sans} -size 13 -
weight normal -slant roman -underline 0 -overstrike 0")
self.Entry6.configure(selectbackground="blue")
self.Entry6.configure(selectforeground="white")

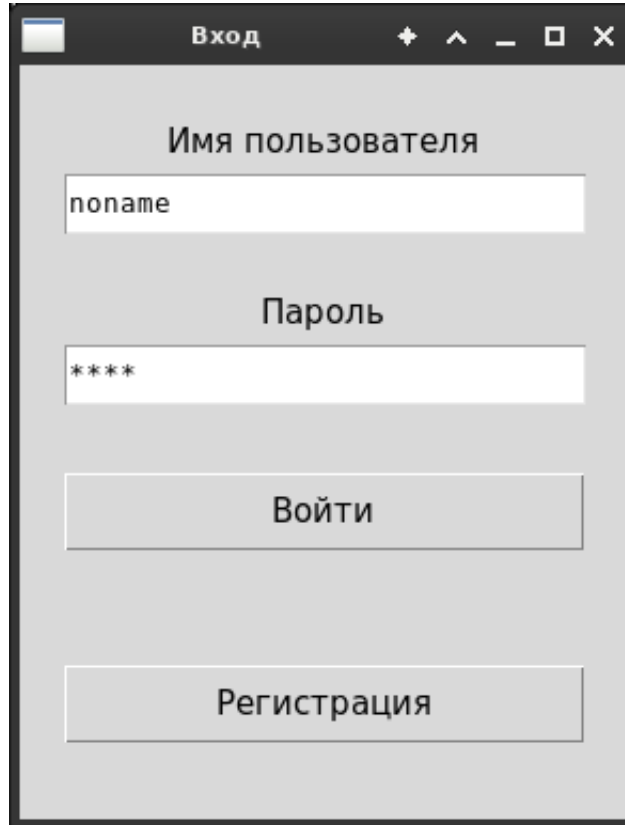
self.Entry7 = tk.Entry(top)
self.Entry7.place(relx=0.437, rely=0.653,height=30, relwidth=0.5)
self.Entry7.configure(background="white")
self.Entry7.configure(font="-family {Liberation Sans} -size 13 -
weight normal -slant roman -underline 0 -overstrike 0")
self.Entry7.configure(selectbackground="blue")
self.Entry7.configure(selectforeground="white")

self.Entry8 = tk.Entry(top)
self.Entry8.place(relx=0.437, rely=0.735,height=30, relwidth=0.5)
self.Entry8.configure(background="white")
self.Entry8.configure(font="-family {Liberation Sans} -size 13 -
weight normal -slant roman -underline 0 -overstrike 0")
self.Entry8.configure(selectbackground="blue")
self.Entry8.configure(selectforeground="white")

if __name__ == '__main__':
    root = tk.Tk()
    window_1(root).toplevel.mainloop()

```

Результат выполнения программы:



The screenshot shows a login window titled "Вход". It contains two text input fields: the first is labeled "Имя пользователя" and contains the text "popame"; the second is labeled "Пароль" and contains four asterisks "****". Below the password field are two buttons: "Войти" (Login) and "Регистрация" (Registration).

Рис 1. Окно входа (ввод данных)

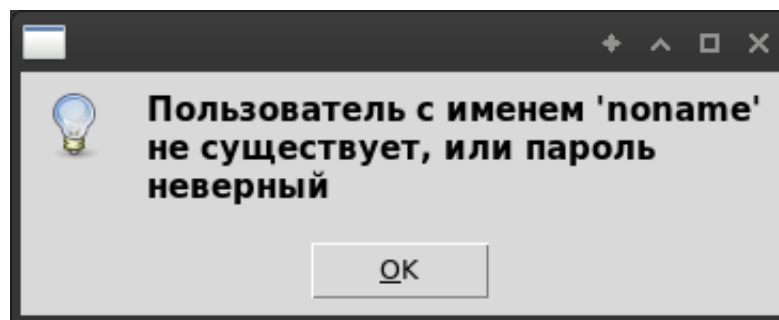


Рис 2. Ошибка при попытке входа с неправильным паролем

Регистрация

Имя пользователя

Пароль

Фамилия

Имя

Отчество

Дата рождения

Место рождения

Телефон

Рис 3.Окно регистрации

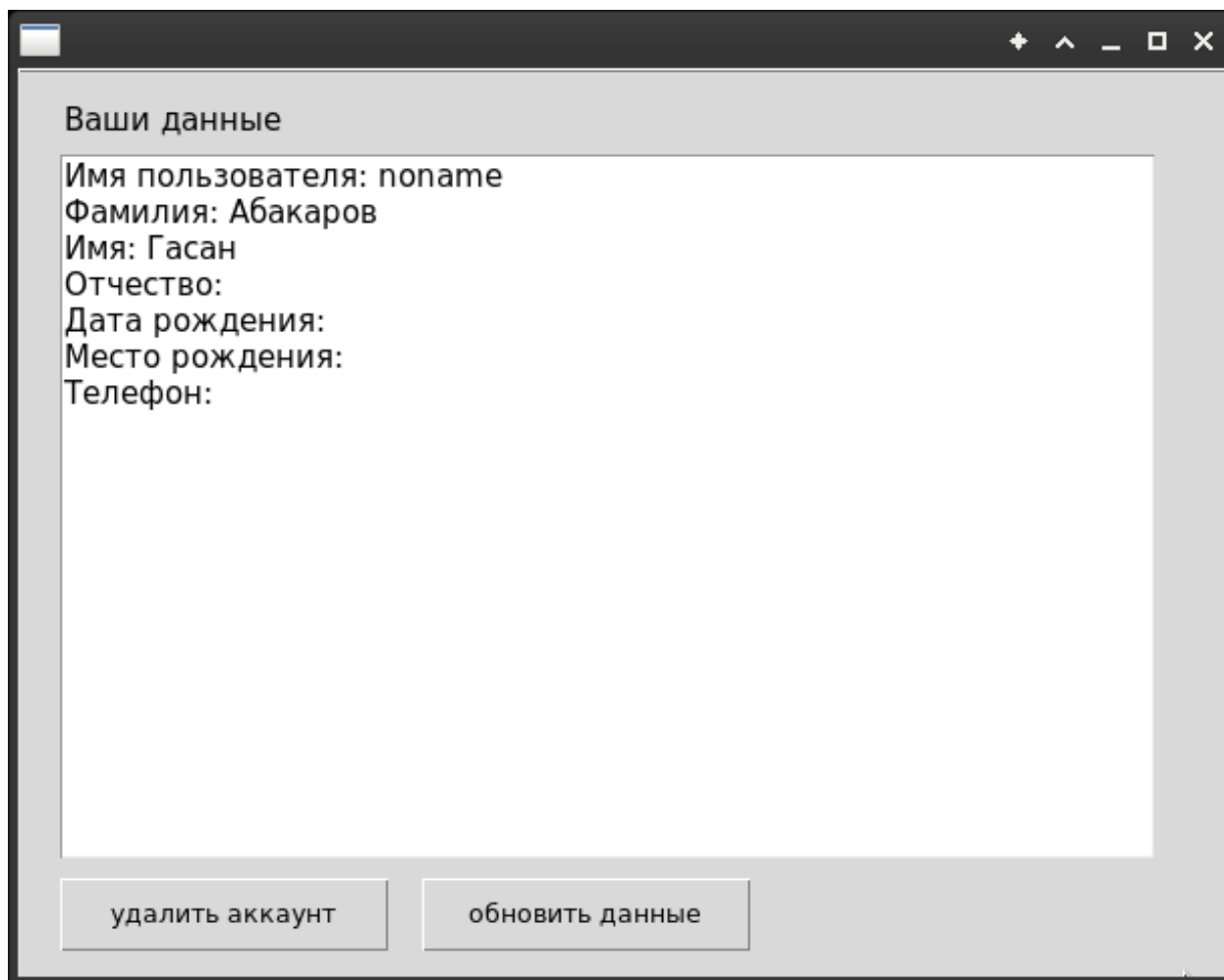


Рис 4. Окно при попытке входа с правильным паролем

Обновить данные

Имя пользователя: noname

Пароль: *****

Фамилия: Абакаров

Имя: Гасан

Отчество:

Дата рождения:

Место рождения:

Телефон:

Сохранить

Рис 5. Окно для смены пароля и других данных (поле «Имя пользователя» в режиме чтения)

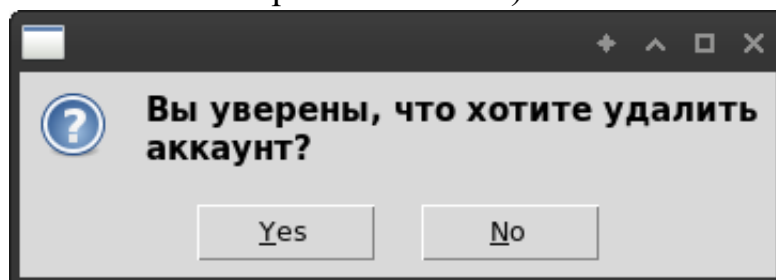


Рис 7. Окно предупреждения при попытке удалить аккаунт

Выводы: я изучил технологии аутентификации пользователя на основе пароля.

Контрольные вопросы

1. Дать определение аутентификации. Привести примеры.
2. Дать определение идентификации в информационных системах.
3. Дать определение авторизации пользователя.

4. Дать определение пароля.

Ответы:

1. Аутентификация — процедура проверки подлинности, например:

- проверка подлинности пользователя (пароли, биометрические данные, библиографические данные и т.д.)
- подтверждение подлинности электронного письма путём проверки цифровой подписи письма по открытому ключу отправителя;
- проверка контрольной суммы файла на соответствие сумме, заявленной автором этого файла.

2. Идентификация в информационных системах — процедура, в результате выполнения которой субъект идентификации получает уникальное имя (идентификатор), однозначно идентифицирующий этого субъекта в информационной системе.

3. Авторизация — предоставление определённому лицу или группе лиц прав на выполнение определённых действий; а также процесс проверки (подтверждения) данных прав при попытке выполнения этих действий.

4. Пароль - набор знаков, предназначенный для авторизации или аутентификации. Используются для защиты информации от несанкционированного доступа.