**ПРАКТИЧНА РОБОТА № 1**

Ремха Б. Т. 125-18-2

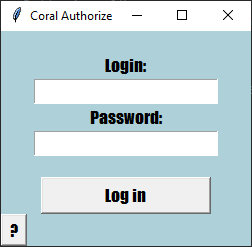
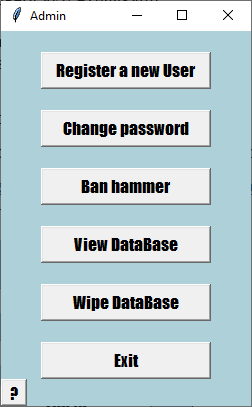
**Тема роботи:**Розмежування повноважень користувачів на основі парольноїаутентифікації.

**Мета роботи:**Розробка програми розмежування повноважень користувачівна основі парольної аутентифікації

Варiант 14

Наявність малих і великих літер, а також знаків арифметичних операцій.

<https://github.com/CoralHeavens/LoginToDatabase.git>

Приклад роботи.

Висновок: розмежування прав доступу – важлива складова программи.

from tkinter import \*  
from tkinter import messagebox  
import tkinter.ttk as ttk  
import tkinter as tk  
import os  
import sqlite3 as sql  
import sys  
from itertools import cycle  
from User import User  
  
root = Tk()  
Session = User('', '', True)  
arithmetic\_op = ['+', '-', '\_', '=', '/', '\*', '%', '^', '~']  
nameDB = 'user\_list'  
  
  
class Table(tk.Frame):  
 def \_\_init\_\_(self, parent=None, headings=tuple(), rows=tuple()):  
 super().\_\_init\_\_(parent)  
 table = ttk.Treeview(self, show="headings", selectmode="browse")  
 table["columns"] = headings  
 table["displaycolumns"] = headings  
 for head in headings:  
 table.heading(head, text=head, anchor=tk.CENTER)  
 table.column(head, anchor=tk.CENTER)  
 for row in rows:  
 table.insert('', tk.END, values=tuple(row))  
 scrolltable = tk.Scrollbar(self, command=table.yview)  
 table.configure(yscrollcommand=scrolltable.set)  
 scrolltable.pack(side=RIGHT, fill=tk.Y)  
 table.pack(expand=tk.YES, fill=tk.BOTH)  
  
  
def Login\_window():  
 root.title("Coral Authorize")  
 root.geometry(f"250x215+400+300")  
 root['bg'] = '#add0d9'  
 root.minsize(250, 215)  
 root.maxsize(250, 215)  
  
 blank\_print()  
  
 login\_text = Label(root,  
 text="Login:",  
 width=20,  
 bg='#add0d9',  
 font=('impact', 13))  
 login\_entry = Entry(root,  
 font=('impact', 13),  
 width=20, )  
 login\_text.pack()  
 login\_entry.pack()  
  
 password\_text = Label(root,  
 text="Password:",  
 width=20,  
 bg='#add0d9',  
 font=('impact', 13))  
 password\_entry = Entry(root,  
 show='♦',  
 font=('impact', 13),  
 width=20)  
 password\_text.pack()  
 password\_entry.pack()  
  
 blank\_print()  
  
 button\_login = Button(root,  
 text="Log in",  
 width=20,  
 font=('impact', 13),  
 command=lambda: login\_button(login\_entry.get(), password\_entry.get()))  
 button\_login.pack()  
  
 info(root)  
  
 root.mainloop()  
  
  
def blank\_print():  
 blank = Label(text='',  
 bg='#add0d9')  
 blank.pack()  
  
  
def login\_button(username, password):  
 if try\_user(username):  
 if compare\_pass(username, password):  
 create\_session(username)  
  
 else:  
 User.pass\_try -= 1  
 if User.pass\_try == 0:  
 sys.exit()  
 messagebox.showwarning(title='Error', message='Password is incorrect.\n'  
 'You have ' + str(User.pass\_try) + ' tries left.')  
 else:  
 messagebox.showwarning(title='Error', message='Unknown user.')  
  
  
def Admin\_panel():  
 root.destroy()  
 panelA = Tk()  
 panelA.title(Session.get\_username())  
 panelA.geometry(f"250x375+600+300")  
 panelA.minsize(250, 375)  
 panelA.maxsize(250, 375)  
 panelA['bg'] = '#add0d9'  
  
 blank\_print()  
  
 reg\_button = Button(panelA,  
 text="Register a new User",  
 width=20,  
 font=('impact', 13),  
 command=lambda: reg\_panel())  
 reg\_button.pack()  
  
 blank\_print()  
  
 change\_button = Button(panelA,  
 text="Change password",  
 width=20,  
 font=('impact', 13),  
 command=lambda: change\_password())  
 change\_button.pack()  
  
 blank\_print()  
  
 ban\_button = Button(panelA,  
 text="Ban hammer",  
 width=20,  
 font=('impact', 13),  
 command=lambda: ban())  
 ban\_button.pack()  
  
 blank\_print()  
  
 view\_button = Button(panelA,  
 text="View DataBase",  
 width=20,  
 font=('impact', 13),  
 command=lambda: view\_db())  
 view\_button.pack()  
  
 blank\_print()  
  
 wipe\_button = Button(panelA,  
 text="Wipe DataBase",  
 width=20,  
 font=('impact', 13),  
 command=lambda: wipe\_db())  
 wipe\_button.pack()  
  
 blank\_print()  
  
 exit\_button = Button(panelA,  
 text="Exit",  
 width=20,  
 font=('impact', 13),  
 command=lambda: exit())  
 exit\_button.pack()  
  
 info(panelA)  
  
 panelA.mainloop()  
  
  
def info(panel):  
  
 info\_button = Button(panel,  
 text="?",  
 width=2,  
 height=1,  
 font=('impact', 12),  
 command=lambda: get\_info())  
 info\_button.pack(side=LEFT)  
  
  
def get\_info():  
  
 messagebox.showinfo(title='Info', message="""Ремха Богдан 125-18-2\n  
 (c) CoralHeavens\n  
 Вариант 14\n  
 'Наявність малих і великих літер, а також знаків арифметичних операцій.'\n  
 remkha.b.t@gmail.com""")  
  
  
def User\_panel():  
 root.destroy()  
 panelU = Tk()  
 panelU.title(Session.get\_username())  
 panelU.geometry(f"250x150+600+300")  
 panelU['bg'] = '#add0d9'  
 panelU.minsize(250, 150)  
 panelU.maxsize(250, 150)  
  
 blank\_print()  
  
 change\_button = Button(panelU,  
 text="Change password",  
 width=20,  
 font=('impact', 13),  
 command=lambda: change\_password())  
 change\_button.pack()  
  
 blank\_print()  
  
 exit\_button = Button(panelU,  
 text="Exit",  
 width=20,  
 font=('impact', 13),  
 command=lambda: sys.exit())  
 exit\_button.pack()  
  
 info(panelU)  
  
 panelU.mainloop()  
  
  
def reg(username, password, role, registration1):  
 if password == '':  
  
 insert\_info(username, '', role)  
 registration1.destroy()  
 return  
  
 elif check\_pass(password=password):  
  
 insert\_info(username, password, role)  
 registration1.destroy()  
 return  
  
 else:  
  
 weak\_window = Tk()  
 weak\_window.title('Error')  
 weak\_window.geometry(f"300x60+800+360")  
 weak\_window['bg'] = '#add0d9'  
  
 messagebox.showwarning(title='Warning', message='Your password is too weak!')  
  
  
def ban():  
 ban\_window = Tk()  
 ban\_window.title("Registration")  
 ban\_window.geometry(f"300x300+800+350")  
 ban\_window['bg'] = '#add0d9'  
  
 blank\_print()  
  
 ban\_label = Label(ban\_window,  
 text="Enter username:",  
 width=20,  
 font=('impact', 16),  
 bg='#add0d9')  
 ban\_label.pack()  
  
 blank\_print()  
  
 ban\_enter = Entry(ban\_window,  
 font=('impact', 13),  
 width=20)  
 ban\_enter.pack()  
  
 blank\_print()  
  
 ban\_butt = Button(ban\_window,  
 text="BAN HAMMER",  
 width=20,  
 font=('impact', 13),  
 command=lambda: status\_0(ban\_enter.get(), ban\_window))  
 ban\_butt.pack()  
  
  
def status\_0(name, window):  
 con, cur = connect\_base()  
 cur.execute(f"""UPDATE user\_list SET status = {False} WHERE username = '{name}'""")  
 con.commit()  
 cur.close()  
 window.destroy()  
  
  
def reg\_panel():  
 registration = Tk()  
 registration.title("Registration")  
 registration.geometry(f"300x300+800+350")  
 registration['bg'] = '#add0d9'  
 registration.minsize(300, 300)  
 registration.maxsize(300, 300)  
  
 blank\_print()  
  
 username\_label = Label(registration,  
 text="Login:",  
 width=20,  
 font=('impact', 13),  
 bg='#add0d9')  
 username\_entry = Entry(registration,  
 font=('impact', 13),  
 width=20)  
 username\_label.pack()  
 username\_entry.pack()  
  
 blank\_print()  
  
 password\_label = Label(registration,  
 text="Password:",  
 width=20,  
 font=('impact', 13),  
 bg='#add0d9')  
 password\_entry = Entry(registration,  
 width=20,  
 font=('impact', 13),  
 show='♦')  
 password\_label.pack()  
 password\_entry.pack()  
  
 blank\_print()  
  
 role\_label = Label(registration,  
 text='Role:',  
 width=20,  
 font=('impact', 13),  
 bg='#add0d9')  
 role\_label.pack()  
  
 combo\_role = ttk.Combobox(registration,  
 state='readonly',  
 values=['User',  
 'Admin'])  
 combo\_role['width'] = 20  
 combo\_role['font'] = ('impact', 13)  
 combo\_role.current(0)  
 combo\_role.pack()  
  
 blank\_print()  
  
 reg\_button = Button(registration,  
 width=20,  
 text='Register',  
 font=('impact', 13),  
 command=lambda: reg(username\_entry.get(), password\_entry.get(), combo\_role.get(), registration))  
 reg\_button.pack()  
  
 registration.mainloop()  
  
  
def wipe\_db():  
 try:  
  
 os.remove('user\_list.db')  
 createDB()  
 messagebox.showwarning(title='New', message='Please, register an administrator.')  
 reg\_panel()  
  
 except PermissionError:  
  
 messagebox.showwarning(title='Error', message='Sorry, DataBase is opened in other program.'  
 '\nClose it before deleting.')  
  
  
def view\_db():  
 con, cur = connect\_base()  
 cur.execute("""SELECT \* FROM user\_list""")  
 content = (row for row in cur.fetchall())  
  
 db\_window = Tk()  
 db\_window.title("User List")  
 db\_table = Table(db\_window, headings=('Username', 'Password', 'Role', 'Status'), rows=content)  
 db\_table.pack(expand=tk.YES, fill=tk.BOTH)  
  
 db\_window.mainloop()  
  
  
def change\_password():  
  
 change = Tk()  
 change.title('Change password for ' + str(Session.get\_username()))  
 change.geometry(f"200x250+750+300")  
 change['bg'] = '#add0d9'  
 change.minsize(200, 250)  
 change.maxsize(200, 250)  
  
 blank\_print()  
 blank\_print()  
 blank\_print()  
  
 old\_pass\_label = Label(change,  
 text="Old password:",  
 width=20,  
 font=('impact', 13),  
 bg='#add0d9')  
 old\_pass\_entry = Entry(change,  
 show='♦',  
 font=('impact', 13),  
 width=20)  
 old\_pass\_label.pack()  
 old\_pass\_entry.pack()  
  
 blank\_print()  
  
 new\_pass\_label = Label(change,  
 text="New password:",  
 width=20,  
 font=('impact', 13),  
 bg='#add0d9')  
 new\_pass\_entry = Entry(change,  
 show='♦',  
 font=('impact', 13),  
 width=20)  
 new\_pass\_label.pack()  
 new\_pass\_entry.pack()  
  
 blank\_print()  
  
 change\_button = Button(change,  
 text='Confirm',  
 width=20,  
 font=('impact', 13),  
 command=lambda: confirm\_change(str(old\_pass\_entry.get()), str(new\_pass\_entry.get()), change))  
 change\_button.pack(side=BOTTOM)  
  
  
def confirm\_change(old, new, win):  
  
 name = Session.get\_username()  
 if compare\_pass(name, old):  
 con, cur = connect\_base()  
 new = (str(pass\_cipher(new)))  
  
 cur.execute("""UPDATE 'user\_list' SET password = (?) WHERE username = (?)""", (new, name))  
  
 con.commit()  
 cur.close()  
 win.destroy()  
  
 else:  
 messagebox.showwarning(title='Error', message='Password is incorrect.')  
 return  
  
  
def createDB():  
 con, cur = connect\_base()  
 cur.execute("""  
 CREATE TABLE IF NOT EXISTS """  
 + str(nameDB) +  
 """ ('username' STRING, 'password' STRING, 'role' STRING, 'status' BOOLEAN)""")  
 con.commit()  
 cur.close()  
  
  
def insert\_info(username, password, role):  
 password = pass\_cipher(password=password)  
  
 con, cur = connect\_base()  
 cur.execute("""INSERT INTO 'user\_list' VALUES (?, ?, ?, ?);""", (str(username), str(password), str(role), True))  
 con.commit()  
 cur.close()  
  
  
def check\_pass(password):  
 if check\_upper(password) and check\_lower(password) and check\_operation(password):  
 return True  
  
  
def check\_upper(password):  
 for x in range(len(password)):  
 if password[x].isupper():  
 return True  
 return False  
  
  
def check\_lower(password):  
 for x in range(len(password)):  
 if password[x].islower():  
 return True  
 return False  
  
  
def check\_operation(password):  
 for x in range(len(password)):  
 if password[x] in arithmetic\_op:  
 return True  
 return False  
  
  
def xor(message, key):  
 return bytes(a ^ b for a, b in zip(message, cycle(key)))  
  
  
def pass\_cipher(password):  
 key1 = b'this\_is\_key'  
 key2 = b'also\_here\_another'  
 result = xor(password.encode(), key1)  
  
 return xor(result, key2)  
  
  
def connect\_base():  
 con = sql.connect('user\_list.db')  
 cur = con.cursor()  
  
 return con, cur  
  
  
def compare\_pass(username, password):  
 check\_password = extract\_data('password', username)  
 password = pass\_cipher(password)  
  
 return str(password) == str(check\_password)  
  
  
def create\_session(username):  
 role = extract\_data('role', username)  
 status = extract\_data('status', username)  
  
 if not status:  
 messagebox.showwarning(title="Error", message="You've been blocked.\nContact Admin.")  
 sys.exit()  
  
 Session.set\_all(username, role, status)  
 if role == 'Admin':  
 Admin\_panel()  
 elif role == 'User':  
 User\_panel()  
  
  
def extract\_data(column, username):  
 con, cur = connect\_base()  
 cur.execute("""SELECT """ + str(column) + """ FROM user\_list WHERE username=(?)""", [username])  
 column\_raw = cur.fetchall()  
 new\_column = [i[0] for i in column\_raw]  
 new\_column = new\_column[0]  
 cur.close()  
  
 return new\_column  
  
  
def try\_user(username):  
 con, cur = connect\_base()  
 cur.execute("""SELECT username FROM user\_list""")  
 users\_raw = cur.fetchall()  
 con.close()  
  
 users = [i[0] for i in users\_raw]  
  
 for i in users:  
 if i == username:  
 return True  
 return False  
  
  
Login\_window()