Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Вятский государственный университет»

Колледж ВятГУ

**ОТЧЕТ**

**ПО ДОМАШНЕЙ КОНТРОЛЬНОЙ РАБОТЕ №9**

**«Работа с типизированными файлами»**

Выполнил: студент учебной группы

ИСПк-205-52-00

Митягин Д. С. Праздников А. А.

Храмов Д. Е. Панов А. А.

Преподаватель:

Сергеева Елизавета Григорьевна

Киров

2023

1. Цель работы: получить представление об организации хранения и использования информации посредством типизированных файлов, закрепить навыки создания пользовательского интерфейса.

2. Задание:

1. Сформировать структуру записи, с которой будет осуществляться работа, и согласовать ее с

преподавателем.

2. Разработать приложение, позволяющее извлекать набор записей из типизированного файла, визуализировать данный набор, а также позволяющее изменять данных и выполнять сохранение проделанных изменений.

3. Исходный код

import tkinter as tk

from tkinter import ttk

import sqlite3

class Main(tk.Frame):

def \_\_init\_\_(self, root):

super().\_\_init\_\_(root)

self.init\_main()

self.db = db

self.view\_records()

def init\_main(self):

toolbar = tk.Frame(bg='white', bd=2)

toolbar.pack(side=tk.TOP, fill=tk.X)

self.add\_img = tk.PhotoImage(file='add.png')

btn\_open\_dialog = tk.Button(toolbar, text='Добавить контакт', command=self.open\_dialog, bg='white',

bd=0, compound=tk.TOP, image=self.add\_img)

btn\_open\_dialog.pack(side=tk.LEFT)

self.update\_img = tk.PhotoImage(file='edit.png')

btn\_edit\_dialog = tk.Button(toolbar, text='Редактировать', bg='white', bd=0, image=self.update\_img,

compound=tk.TOP, command=self.open\_update\_dialog)

btn\_edit\_dialog.pack(side=tk.LEFT)

self.delete\_img = tk.PhotoImage(file='delete.png')

btn\_delete\_dialog = tk.Button(toolbar, text='Удалить', bg='white', bd=0, image=self.delete\_img,

compound=tk.TOP, command=self.delete\_records)

btn\_delete\_dialog.pack(side=tk.LEFT)

self.search\_img = tk.PhotoImage(file='search.png')

btn\_search = tk.Button(toolbar, text='Поиск', bg='white', bd=0, image=self.search\_img,

compound=tk.TOP, command=self.open\_search\_dialog)

btn\_search.pack(side=tk.LEFT)

self.refresh\_img = tk.PhotoImage(file='refresh.png')

btn\_refresh = tk.Button(toolbar, text='Обновить', bg='white', bd=0, image=self.refresh\_img,

compound=tk.TOP, command=self.view\_records)

btn\_refresh.pack(side=tk.LEFT)

self.tree = ttk.Treeview(self, columns=('ID', 'fio', 'category', 'phone'), height=15, show='headings')

self.tree.column('ID', width=30, anchor=tk.CENTER)

self.tree.column('fio', width=365, anchor=tk.CENTER)

self.tree.column('category', width=150, anchor=tk.CENTER)

self.tree.column('phone', width=100, anchor=tk.CENTER)

self.tree.heading('ID', text='№')

self.tree.heading('fio', text='Имя контакта')

self.tree.heading('category', text='Тип телефона')

self.tree.heading('phone', text='Телефон')

self.tree.pack(side=tk.LEFT)

scroll = tk.Scrollbar(self, command=self.tree.yview)

scroll.pack(side=tk.LEFT, fill=tk.Y)

self.tree.configure(yscrollcommand=scroll.set)

def records(self, fio, category, phone):

self.db.insert\_data(fio, category, phone)

self.view\_records()

def update\_record(self, fio, category, phone):

self.db.c.execute('''UPDATE phonebook SET fio=?, category=?, phone=? WHERE ID=?''',

(fio, category, phone, self.tree.set(self.tree.selection()[0], '#1')))

self.db.conn.commit()

self.view\_records()

def view\_records(self):

self.db.c.execute('''SELECT \* FROM phonebook''')

[self.tree.delete(i) for i in self.tree.get\_children()]

[self.tree.insert('', 'end', values=row) for row in self.db.c.fetchall()]

def delete\_records(self):

for selection\_item in self.tree.selection():

self.db.c.execute('''DELETE FROM phonebook WHERE id=? ''', (self.tree.set(selection\_item, '#1'),))

self.db.conn.commit()

self.view\_records()

def search\_records(self, fio):

fio = ('%' + fio + '%',)

self.db.c.execute('''SELECT \* FROM phonebook WHERE fio LIKE ?''', fio)

[self.tree.delete(i) for i in self.tree.get\_children()]

[self.tree.insert('', 'end', values=row) for row in self.db.c.fetchall()]

def open\_dialog(self):

Child()

def open\_update\_dialog(self):

Update()

def open\_search\_dialog(self):

Search()

class Child(tk.Toplevel):

def \_\_init\_\_(self):

super().\_\_init\_\_(root)

self.init\_child()

self.view = app

def init\_child(self):

self.title('Добавить контакт')

self.geometry('400x220+400+300')

self.resizable(False, False)

label\_fio = tk.Label(self, text='ФИО')

label\_fio.place(x=50, y=50)

label\_select = tk.Label(self, text='Категория')

label\_select.place(x=50, y=80)

label\_phone = tk.Label(self, text='Телефон:')

label\_phone.place(x=50, y=110)

self.entry\_fio = ttk.Entry(self)

self.entry\_fio.place(x=200, y=50)

self.entry\_phone = ttk.Entry(self)

self.entry\_phone.place(x=200, y=110)

self.combobox = ttk.Combobox(self, values=[u'Семья', u'Друзья', 'Работа'])

self.combobox.current(0)

self.combobox.place(x=200, y=80)

btn\_cancel = ttk.Button(self, text='Закрыть', command=self.destroy)

btn\_cancel.place(x=300, y=150)

self.btn\_ok = ttk.Button(self, text='Добавить')

self.btn\_ok.place(x=220, y=150)

self.btn\_ok.bind('<Button-1>', lambda event: self.view.records(self.entry\_fio.get(),

self.combobox.get(),

self.entry\_phone.get()))

self.grab\_set()

self.focus\_set()

class Update(Child):

def \_\_init\_\_(self):

super().\_\_init\_\_()

self.init\_edit()

self.view = app

self.db = db

self.default\_data()

def init\_edit(self):

self.title('Редактировать контакт')

btn\_edit = ttk.Button(self, text='Редактировать')

btn\_edit.place(x=205, y=150)

btn\_edit.bind('<Button-1>', lambda event: self.view.update\_record(self.entry\_fio.get(),

self.combobox.get(),

self.entry\_phone.get()))

self.btn\_ok.destroy()

def default\_data(self):

self.db.c.execute('''SELECT \* FROM phonebook WHERE id=?''',

(self.view.tree.set(self.view.tree.selection()[0], '#1'),))

row = self.db.c.fetchone()

self.entry\_fio.insert(0, row[1])

if row[2] != 'Семья':

self.combobox.current(1)

self.entry\_phone.insert(0, row[3])

class Search(tk.Toplevel):

def \_\_init\_\_(self):

super().\_\_init\_\_()

self.init\_search()

self.view = app

def init\_search(self):

self.title('Поиск')

self.geometry('300x100+400+300')

self.resizable(False, False)

label\_search = tk.Label(self, text='Поиск')

label\_search.place(x=50, y=20)

self.entry\_search = ttk.Entry(self)

self.entry\_search.place(x=105, y=20, width=150)

btn\_cancel = ttk.Button(self, text='Закрыть', command=self.destroy)

btn\_cancel.place(x=185, y=50)

btn\_search = ttk.Button(self, text='Поиск')

btn\_search.place(x=105, y=50)

btn\_search.bind('<Button-1>', lambda event: self.view.search\_records(self.entry\_search.get()))

btn\_search.bind('<Button-1>', lambda event: self.destroy(), add='+')

class DB:

def \_\_init\_\_(self):

self.conn = sqlite3.connect('phonebook.db')

self.c = self.conn.cursor()

self.c.execute(

'''CREATE TABLE IF NOT EXISTS phonebook (id integer primary key,fio text,

category text, phone text)''')

self.conn.commit()

def insert\_data(self, fio, category, phone):

self.c.execute('''INSERT INTO phonebook (fio, category, phone) VALUES (?, ?, ?) ''',

(fio, category, phone))

self.conn.commit()

if \_\_name\_\_ == "\_\_main\_\_":

root = tk.Tk()

db = DB()

app = Main(root)

app.pack()

root.title("Контакты")

root.geometry("665x450+300+200")

root.resizable(False, False)

root.mainloop()

4. Результат выполнения программы

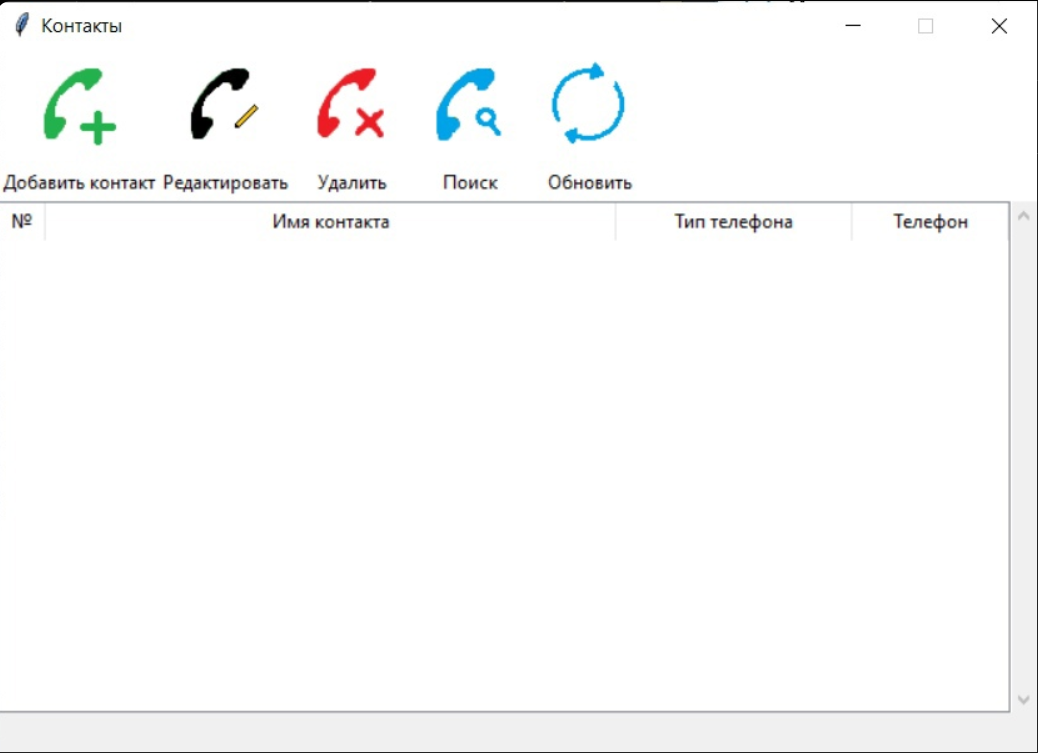


Рисунок 1 - Контакты

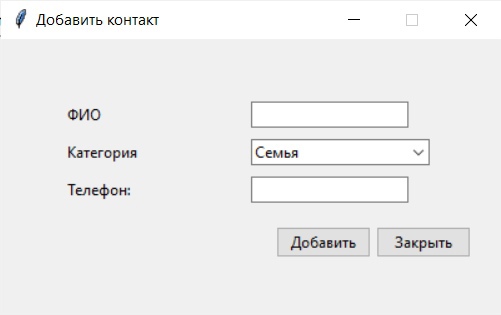


Рисунок 2 – Добавить контакт

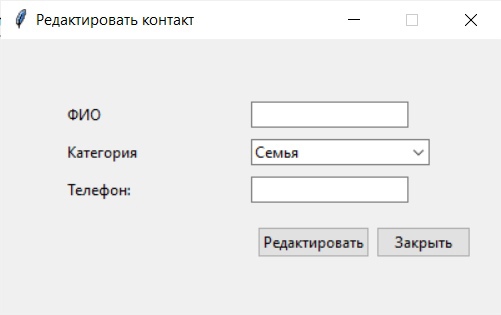


Рисунок 3 – Редактировать контакт

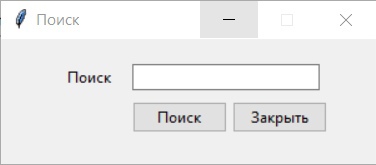


Рисунок 4 – Поиск

5. Вывод:

Для выполнения задания было была использована IDA PyCharm и библиотека tkinter, sqlite3.

Были использованы функции для добавления и редактирования контактов, обновления таблиц, поиска и удаления контактов.

Так же были применены функции вышеупомянутых библиотек.

Благодаря всему вышеперечисленному задание было успешно выполнено.