**Практическая работа №5. Создание класса базы данных. Реализация основных операций с данными (CRUD) в настольном приложении.**

Выполнил: Кайнов Михаил

Предметная область: География

Ход работы:

1. Модуль database.py

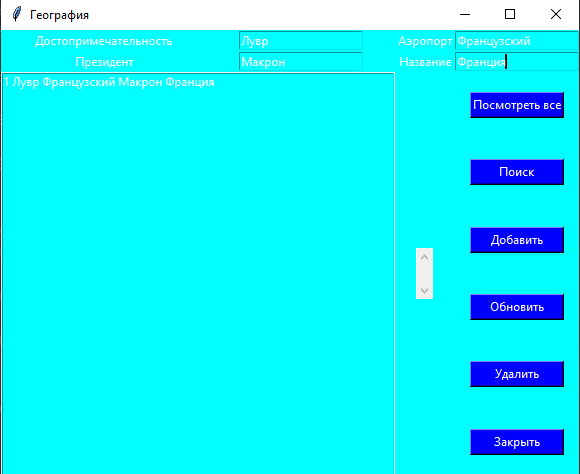
import sqlite3  
  
class DB:  
 def \_\_init\_\_(self):  
 self.conn = sqlite3.connect("geography.db")  
 self.cur = self.conn.cursor()  
 self.cur.execute(  
 "CREATE TABLE IF NOT EXISTS geogr (id INTEGER PRIMARY KEY, dost TEXT, air TEXT, prezident TEXT, name TEXT)")  
 self.conn.commit()  
  
 def \_\_del\_\_(self):  
 self.conn.close()  
  
 def view(self):  
 self.cur.execute("SELECT \* FROM geogr")  
 rows = self.cur.fetchall()  
 return rows  
  
 def insert(self, dost, air, prezident, name):  
 self.cur.execute("INSERT INTO geogr VALUES (NULL,?,?,?,?)", (dost, air, prezident, name))  
 self.conn.commit()  
  
 def update(self, id, dost, air, prezident, name):  
 self.cur.execute("UPDATE geogr SET dost=?, air=?, prezident=?, name=? WHERE id=?", (dost, air, prezident, name, id,))  
 self.conn.commit()  
  
 def delete(self, id):  
 self.cur.execute("DELETE FROM geogr WHERE id=?", (id,))  
 self.conn.commit()  
  
 def search(self, name=""):  
 self.cur.execute("SELECT \* FROM geogr WHERE name=?", (name,))  
 rows = self.cur.fetchall()  
 return rows

1. Модуль main.py

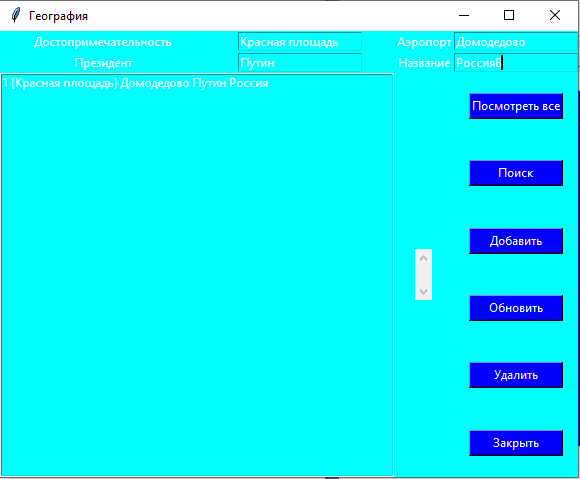
from tkinter import \*  
from tkinter import messagebox  
from database import DB  
  
db = DB()  
  
def get\_selected\_row(event):  
 global selected\_tuple  
 index = list1.curselection()[0]  
 selected\_tuple = list1.get(index)  
 e1.delete(0, END)  
 e1.insert(END, selected\_tuple[1])  
 e2.delete(0, END)  
 e2.insert(END, selected\_tuple[2])  
 e3.delete(0, END)  
 e3.insert(END, selected\_tuple[3])  
 e4.delete(0, END)  
 e4.insert(END, selected\_tuple[4])  
  
def view\_command():  
 list1.delete(0, END)  
 for row in db.view():  
 list1.insert(END, row)  
  
def search\_command():  
 list1.delete(0, END)  
 for row in db.search(product\_text.get()):  
 list1.insert(END, row)  
  
def add\_command():  
 db.insert(product\_text.get(), price\_text.get(), comment\_text.get(), mine\_text.get())  
 view\_command()  
  
def delete\_command():  
 db.delete(selected\_tuple[0])  
 view\_command()  
  
def update\_command():  
 db.update(selected\_tuple[0], product\_text.get(), price\_text.get(), comment\_text.get(), mine\_text.get())  
 view\_command()  
  
window = Tk()  
window.title("География")  
  
window.configure(bg='aqua')  
window.option\_add('\*TButton\*background', 'aqua')  
window.option\_add('\*TButton\*foreground', 'aqua')  
window.option\_add('\*TEntry\*background', 'aqua')  
window.option\_add('\*TEntry\*foreground', 'aqua')  
window.option\_add('\*TListbox\*background', 'aqua')  
window.option\_add('\*TListbox\*foreground', 'aqua')  
  
def on\_closing():  
 if messagebox.askokcancel("", "Закрыть программу?"):  
 window.destroy()  
  
window.protocol("WM\_DELETE\_WINDOW", on\_closing)  
  
l1 = Label(window, text="Достопримечательность", bg='aqua', fg='#FFFFFF')  
l1.grid(row=0, column=0)  
  
l2 = Label(window, text="Аэропорт", bg='aqua', fg='#FFFFFF')  
l2.grid(row=0, column=2)  
  
l3 = Label(window, text="Президент", bg='aqua', fg='#FFFFFF')  
l3.grid(row=1, column=0)  
  
l4 = Label(window, text="Название", bg='aqua', fg='#FFFFFF')  
l4.grid(row=1, column=2)  
  
product\_text = StringVar()  
e1 = Entry(window, textvariable=product\_text, bg='aqua', fg='#FFFFFF')  
e1.grid(row=0, column=1)  
  
price\_text = StringVar()  
e2 = Entry(window, textvariable=price\_text, bg='aqua', fg='#FFFFFF')  
e2.grid(row=0, column=3)  
  
comment\_text = StringVar()  
e3 = Entry(window, textvariable=comment\_text, bg='aqua', fg='#FFFFFF')  
e3.grid(row=1, column=1)  
  
mine\_text = StringVar()  
e4 = Entry(window, textvariable=mine\_text, bg='aqua', fg='#FFFFFF')  
e4.grid(row=1, column=3)  
  
list1 = Listbox(window, height=25, width=65, bg='aqua', fg='#FFFFFF')  
list1.grid(row=2, column=0, rowspan=6, columnspan=2)  
  
sb1 = Scrollbar(window)  
sb1.grid(row=2, column=2, rowspan=6)  
  
list1.configure(yscrollcommand=sb1.set)  
sb1.configure(command=list1.yview)  
  
list1.bind('<<ListboxSelect>>', get\_selected\_row)  
  
b1 = Button(window, text="Посмотреть все", width=12, command=view\_command, bg='blue', fg='#FFFFFF')  
b1.grid(row=2, column=3)  
  
b2 = Button(window, text="Поиск", width=12, command=search\_command, bg='blue', fg='#FFFFFF')  
b2.grid(row=3, column=3)  
  
b3 = Button(window, text="Добавить", width=12, command=add\_command, bg='blue', fg='#FFFFFF')  
b3.grid(row=4, column=3)  
  
b4 = Button(window, text="Обновить", width=12, command=update\_command, bg='blue', fg='#FFFFFF')  
b4.grid(row=5, column=3)  
  
b5 = Button(window, text="Удалить", width=12, command=delete\_command, bg='blue', fg='#FFFFFF')  
b5.grid(row=6, column=3)  
  
b6 = Button(window, text="Закрыть", width=12, command=on\_closing, bg='blue', fg='#FFFFFF')  
b6.grid(row=7, column=3)  
  
view\_command()  
  
window.mainloop()

1. Результат:

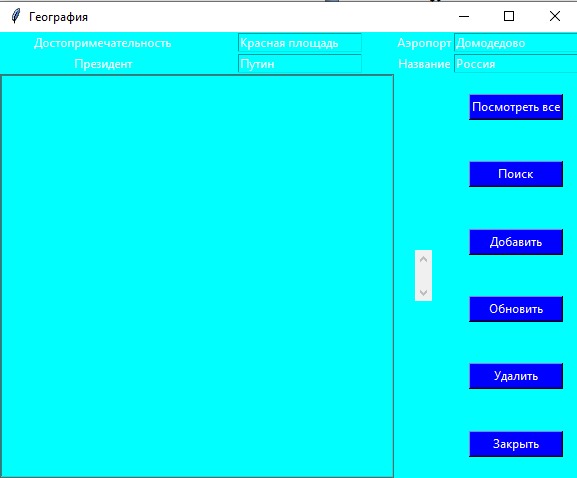
Добавление:



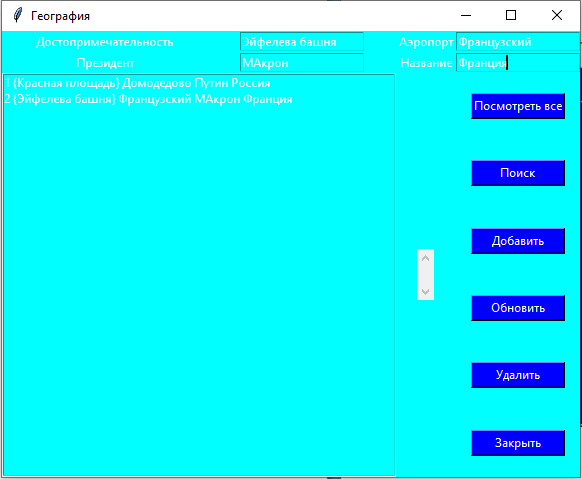
Обновление:

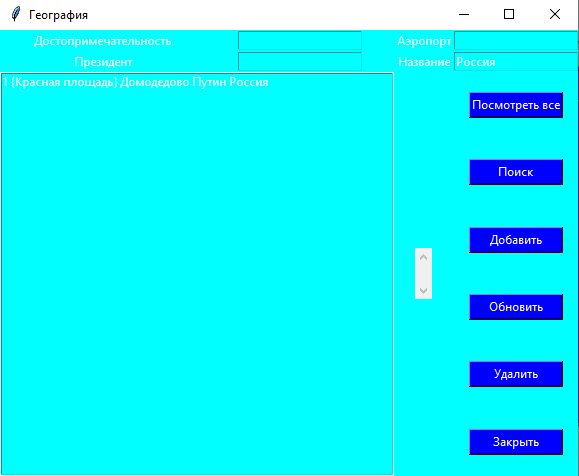


Удаление:

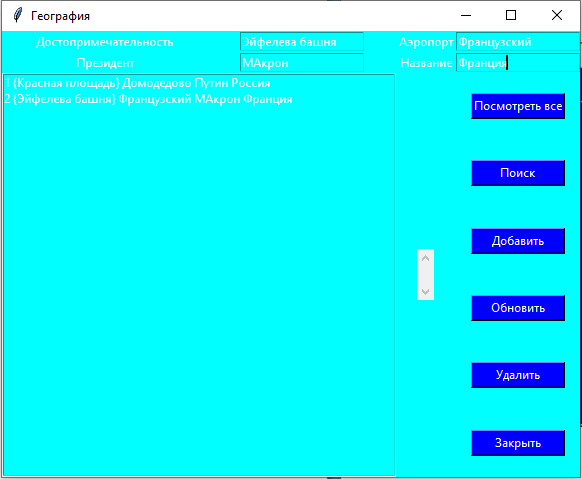


Поиск:





Посмотреть все:



Закрыть:

