

Практическое занятие №16

Тема: разработка многооконного приложения для работы с однотабличной БД в IDL PyCharm Community.

Цель: закрепить усвоенные знания, понятия, алгоритмы, основные принципы составления программ, работы с БД в IDL PyCharm Community.

Постановка задачи: Реализовать многооконное приложение для работы с однотабличной БД. Приложение должно обеспечивать функционал по вводу, поиску, удалению и редактированию данных в БД.

Приложение СДАЧА В АРЕНДУ ТОРГОВЫХ ПЛОЩАДЕЙ для некоторой организации. БД должна содержать таблицу Клиент со следующей структурой записи: ФИО клиента, код помещения, срок аренды, стоимость аренды за весь срок.

Тип алгоритма: циклический и линейный.

Текст программы:

```

"""Приложение СДАЧА В АРЕНДУ ТОРГОВЫХ ПЛОЩАДЕЙ для некоторой организации.
БД должна содержать таблицу Клиент со следующей структурой записи: ФИО
клиента,
код помещения, срок аренды, стоимость аренды за весь срок. """

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="#d7d8e0", bd=2)
        toolbar.pack(side=tk.TOP, fill=tk.X)

        self.add_img = tk.PhotoImage(file='plus.gif')

        btn_open_dialog = tk.Button(toolbar, text='Добавить позицию',
command=self.open_dialog, bg="#d7d8e0", bd=0,
compound=tk.TOP, image=self.add_img)
        btn_open_dialog.pack(side=tk.LEFT)

        self.update_img = tk.PhotoImage(file='edit.gif')
        btn_edit_dialog = tk.Button(toolbar, text='Редактировать',
bg="#d7d8e0", 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.gif')

```

```

        btn_delete = tk.Button(toolbar, text='Удалить запись', bg="#d7d8e0",
                                bd=0, image=self.delete_img,
                                compound=tk.TOP,
                                command=self.delete_records)
        btn_delete.pack(side=tk.LEFT)

        self.search_img = tk.PhotoImage(file='search.gif')
        btn_search = tk.Button(toolbar, text='Поиск записи', bg="#d7d8e0",
                                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.gif')
        btn_refresh = tk.Button(toolbar, text='Обновить таблицу',
                                bg="#d7d8e0", 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=('fio', 'id', 'date', 'cash'),
                                height=15, show='headings')
        self.tree.column('fio', width=200, anchor=tk.CENTER)
        self.tree.column('id', width=130, anchor=tk.CENTER)
        self.tree.column('date', width=130, anchor=tk.CENTER)
        self.tree.column('cash', width=200, anchor=tk.CENTER)

        self.tree.heading('fio', text='ФИО')
        self.tree.heading('id', text='Код помещения')
        self.tree.heading('date', text='Срок аренды')
        self.tree.heading('cash', text='Стоимость за весь срок')

        self.tree.pack()

    def records(self, fio, id, date, cash):
        self.db.insert_data(fio, id, date, cash)
        self.view_records()

    def update_record(self, fio, id, date, cash):
        self.db.c.execute('UPDATE klient SET fio=?, id=?, date=?, cash=?
WHERE id=?',
                        (fio, id, date, cash,
self.tree.set(self.tree.selection()[0], '#2'))
        self.db.conn.commit()
        self.view_records()

    def view_records(self):
        self.db.c.execute('SELECT * FROM klient')
        [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 klient WHERE id=?',
(self.tree.set(selection_item, '#2'),))

            self.db.conn.commit()
            self.view_records()

    def search_records(self, fio):
        fio = ('%' + fio + '%')
        self.db.c.execute('SELECT * FROM klient 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_dilog(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')
        self.geometry('400x220+400+300')
        self.resizable(False, False)

        label_name = tk.Label(self, text='ФИО:')
        label_name.place(x=50, y=30)

        label_kod = tk.Label(self, text='Код помещения:')
        label_kod.place(x=50, y=60)

        label_date = tk.Label(self, text='Срок аренды:')
        label_date.place(x=50, y=90)

        label_cash = tk.Label(self, text='Стоимость за весь срок:')
        label_cash.place(x=50, y=120)

        self.entry_name = ttk.Entry(self)
        self.entry_name.place(x=200, y=30)

        self.entry_kod = ttk.Entry(self)
        self.entry_kod.place(x=200, y=60)

        self.entry_date = ttk.Entry(self)
        self.entry_date.place(x=200, y=90)

        self.entry_cash = ttk.Entry(self)
        self.entry_cash.place(x=200, y=120)

        btn_cancel = ttk.Button(self, text='Заккрыть', command=self.destroy)
        btn_cancel.place(x=300, y=170)

        self.btn_ok = ttk.Button(self, text='Добавить')
        self.btn_ok.place(x=220, y=170)
        self.btn_ok.bind('<Button-1>', lambda event:
self.view.records(self.entry_name.get(),
self.entry_kod.get(),
self.entry_date.get(),
self.entry_cash.get()))

```

```

        self.grab_set()
        self.focus_set()

class Update(Child):
    def __init__(self):
        super().__init__()
        self.init_edit()
        self.view = app

    def init_edit(self):
        self.title('Редактировать')
        btn_edit = ttk.Button(self, text='Редактировать')
        btn_edit.place(x=205, y=170)
        btn_edit.bind('<Button-1>', lambda event:
self.view.update_record(self.entry_name.get(),
self.entry_kod.get(),
self.entry_date.get(),
self.entry_cash.get()))
        self.btn_ok.destroy()

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='Заккрыть', 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('klient.db')
        self.c = self.conn.cursor()
        self.c.execute(
            """CREATE TABLE IF NOT EXISTS klient (fio text, id integer
primary key, date blob, cash real)""")
        self.conn.commit()

    def insert_data(self, fio, id, date, cash):
        self.c.execute('INSERT INTO klient (fio, id, date, cash) VALUES (?,
?, ?, ?)', (fio, id, date, cash))
        self.conn.commit()

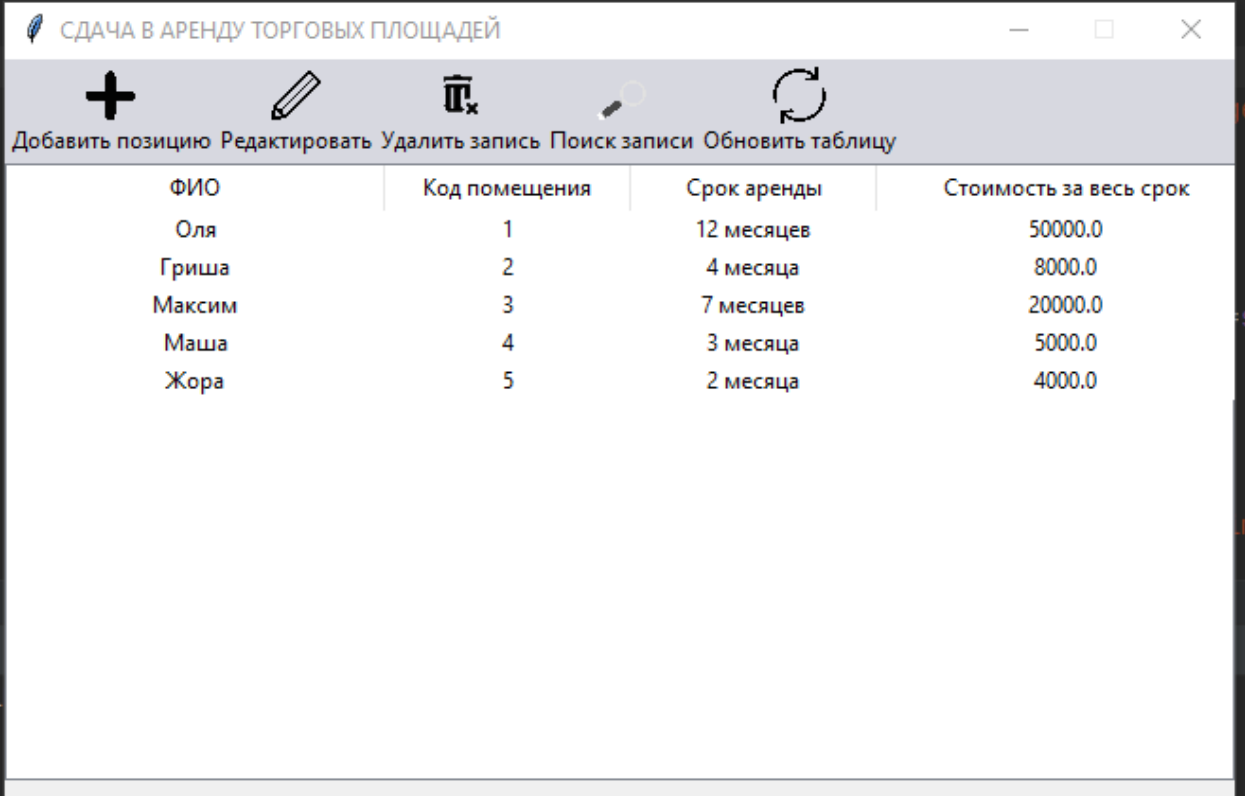
```

```

if __name__ == "__main__":
    root = tk.Tk()
    db = DB()
    app = Main(root)
    app.pack()
    root.title("СДАЧА В АРЕНДУ ТОРГОВЫХ ПЛОЩАДЕЙ")
    root.geometry("650x450+300+200")
    root.resizable(False, False)
    root.mainloop()

```

Протокол работы программы:



ФИО	Код помещения	Срок аренды	Стоимость за весь срок
Оля	1	12 месяцев	50000.0
Гриша	2	4 месяца	8000.0
Максим	3	7 месяцев	20000.0
Маша	4	3 месяца	5000.0
Жора	5	2 месяца	4000.0

Вывод: в процессе практического занятия закрепила усвоенные знания, понятия, алгоритмы, основные принципы составления программ, разработки многооконного приложения для работы с однотабличной БД в IDL PyCharmCommunity.

Выполнены разработка кода, отладка, тестирование, оптимизация программного кода. Готовые программные коды выложены на GitHub