Студентки группы ПОКС-21 Бондарцевой Е.И.

Практическое занятие №16

Вариант №3

Тема: Разработка многооконного приложения для работы с однотабличной БД в IDE PyCharm Community.

Цели практического занятия: закрепить усвоенные знания, понятия, алгоритмы, основные принципы составления программ, работы с БД в IDE PyCharm Community.

Постановка задачи.

Приложение КАФЕДРА для автоматизации работы отдела кадров ВУЗа. Таблица Преподавательский состав должна содержать следующие данные: Табельный номер, Фамилия И.О., Дата рождения, Должность, Ученая степень, Нагрузка, Зарплата. БД должна обеспечивать получение информации о преподавателях по должности.

Текст программы:

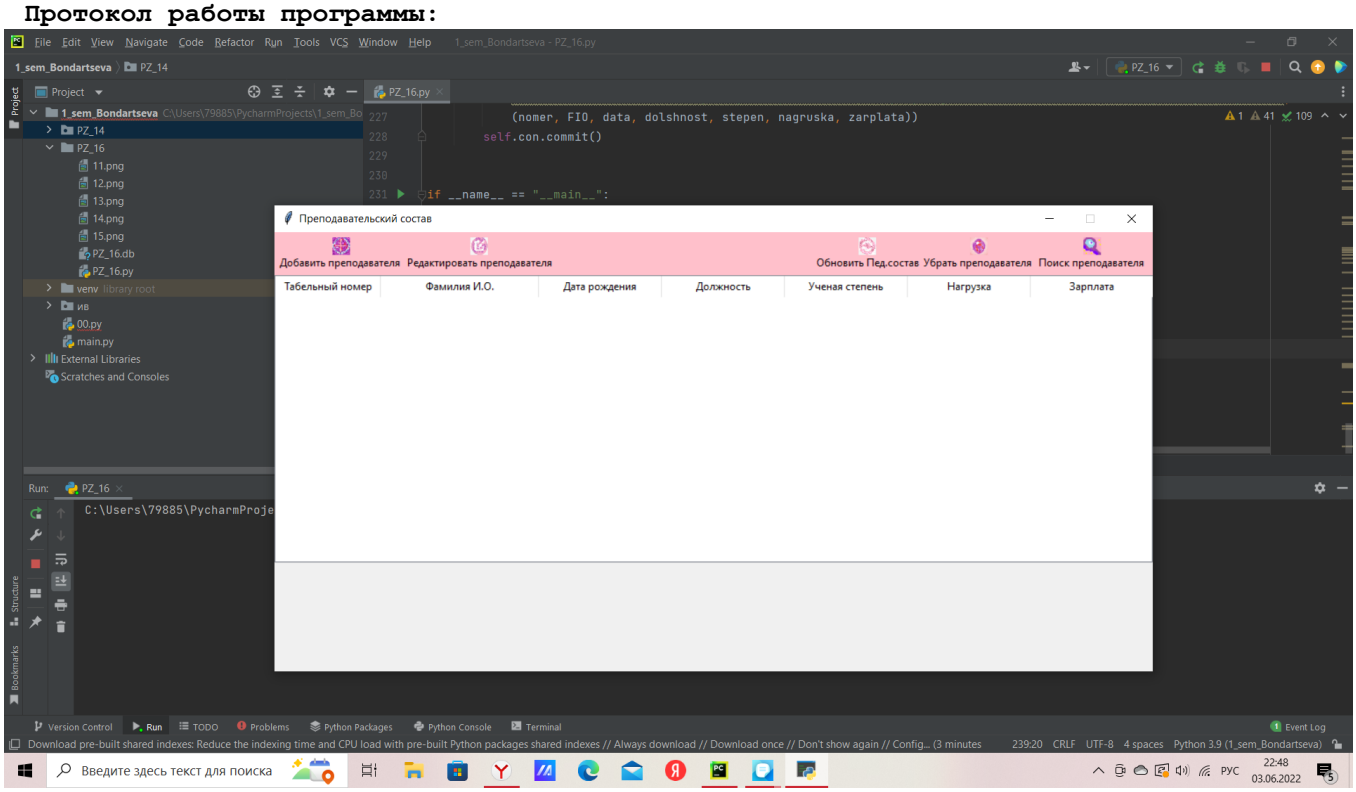
```
# Практическая №16
import tkinter as tk
from tkinter import ttk
import sqlite3 as sq


class Main(tk.Frame):
    """Класс для главного окна"""

    def __init__(self, root):
        super().__init__(root)
        self.init_main()
        self.db = db
        self.view_records()

    def init_main(self):
        toolbar = tk.Frame(bg='#FFC0CB', bd=4)
        toolbar.pack(side=tk.TOP, fill=tk.X)

        self.add_img = tk.PhotoImage(file="11.png")
        self.btn_open_dialog = tk.Button(toolbar, text='Добавить преподавателя ', command=self.open_dialog, bg='#FFC0CB',
                                         bd=0,
                                         compound=tk.TOP, image=self.add_img)
        self.btn_open_dialog.pack(side=tk.LEFT)

        self.search_img = tk.PhotoImage(file="14.png")
        btn_search = tk.Button(toolbar, text="Поиск преподавателя ", command=self.open_search_dialog, bg='#FFC0CB',
                               bd=0, compound=tk.TOP, image=self.search_img)
        btn_search.pack(side=tk.RIGHT)

        self.update_img = tk.PhotoImage(file="12.png")
        btn_edit_dialog = tk.Button(toolbar, text="Редактировать преподавателя", command=self.open_update_dialog,
                                    bg='#FFC0CB',
                                    bd=0, compound=tk.TOP, image=self.update_img)
        btn_edit_dialog.pack(side=tk.LEFT)

        self.delete_img = tk.PhotoImage(file="15.png")
        btn_delete = tk.Button(toolbar, text="Убрать преподавателя ", command=self.delete_records, bg='#FFC0CB',
                               bd=0, compound=tk.TOP, image=self.delete_img)
        btn_delete.pack(side=tk.RIGHT)

        self.refresh_img = tk.PhotoImage(file="13.png")
        btn_refresh = tk.Button(toolbar, text="Обновить Пед.состав", command=self.view_records, bg='#FFC0CB',
                                bd=0, compound=tk.TOP, image=self.refresh_img)
        btn_refresh.pack(side=tk.RIGHT)

        self.tree = ttk.Treeview(self, columns=(
        'nomer', 'FIO', 'data', 'dolshnost', 'stepen', 'nagruska', 'zarplata'), height=15,
                                 show='headings')

        self.tree.column('nomer', width=120, anchor=tk.CENTER)
        self.tree.column('FIO', width=180, anchor=tk.CENTER)
        self.tree.column('data', width=140, anchor=tk.CENTER)
        self.tree.column('dolshnost', width=140, anchor=tk.CENTER)
        self.tree.column('stepen', width=140, anchor=tk.CENTER)
        self.tree.column('nagruska', width=140, anchor=tk.CENTER)
```

```
        self.tree.column('zarplata', width=140, anchor=tk.CENTER)

        self.tree.heading('nomer', text='Табельный номер')
        self.tree.heading('FIO', text='Фамилия И.О.')
        self.tree.heading('data', text='Дата рождения')
        self.tree.heading('dolshnost', text='Должность')
        self.tree.heading('stepen', text='Ученая степень')
        self.tree.heading('nagruska', text='Нагрузка')
        self.tree.heading('zarplata', text='Зарплата')

        self.tree.pack()

    def records(self, nomer, FIO, data, dolshnost, stepen, nagruska, zarplata):
        self.db.insert_data( nomer, FIO, data, dolshnost, stepen, nagruska, zarplata)
        self.view_records()

    def update_record(self,  nomer, FIO, data, dolshnost, stepen, nagruska, zarplata):
        self.db.cur.execute("""UPDATE sostav SET nomer=?, FIO=?, data=?, dolshnost=?, stepen=?, nagruska=?, zarplata=? WHERE nomer=?""" ,
                            (nomer, FIO, data, dolshnost, stepen, nagruska, zarplata,
                             self.tree.set(self.tree.selection()[0], '#1')))
        self.db.con.commit()
        self.view_records()

    def view_records(self):
        self.db.cur.execute("""SELECT * FROM sostav""")
        [self.tree.delete(i) for i in self.tree.get_children()]
        [self.tree.insert('', 'end', values=row) for row in self.db.cur.fetchall()]

    def delete_records(self):
        for selection_item in self.tree.selection():
            self.db.cur.execute("""DELETE FROM sostav WHERE nomer=?""", (self.tree.set(selection_item, '#1'),))
        self.db.con.commit()
        self.view_records()

    def search_records(self, dolshnost):
        dolshnost = ("%" + dolshnost + "%",)
        self.db.cur.execute("""SELECT * FROM disciplina WHERE dolshnost LIKE ?""", dolshnost)
        [self.tree.delete(i) for i in self.tree.get_children()]
        [self.tree.insert('', 'end', values=row) for row in self.db.cur.fetchall()]

    def open_dialog(self):
        Child(root, app)

    def open_update_dialog(self):
        Update()

    def open_search_dialog(self):
        Search()


class Child(tk.Toplevel):
    """Класс для дочернего окна"""

    def __init__(self, root, app):
        super().__init__(root)
        self.init_child()
        self.view = app

    def init_child(self):
        self.title('Добавить дисциплину')
        self.geometry('400x270+400+300')
        self.resizable(False, False)

        label_description = tk.Label(self, text='Табельный номер')
        label_description.place(x=35, y=25)
        self.entry_score1 = ttk.Entry(self)
        self.entry_score1.place(x=170, y=25)

        label_name = tk.Label(self, text='Фамилия И.О')
        label_name.place(x=45, y=50)
        self.entry_name1 = ttk.Entry(self)
        self.entry_name1.place(x=170, y=50)

        label_spec = tk.Label(self, text='Дата рождения ')
        label_spec.place(x=45, y=75)
        self.entry_name2 = ttk.Entry(self)
        self.entry_name2.place(x=170, y=75)

        label_lecs = tk.Label(self, text='Должность')
        label_lecs.place(x=50, y=100)
        self.entry_score2 = ttk.Entry(self)
        self.entry_score2.place(x=170, y=100)

        label_score = tk.Label(self, text='Учебная степень')
        label_score.place(x=40, y=125)
        self.entry_score3 = ttk.Entry(self)
```

```
        self.entry_score3.place(x=170, y=125)

        label_score = tk.Label(self, text='Нагрузка')
        label_score.place(x=55, y=150)
        self.entry_score4 = ttk.Entry(self)
        self.entry_score4.place(x=170, y=150)

        label_otch = tk.Label(self, text='Зарплата')
        label_otch.place(x=55, y=175)
        self.entry_name3 = ttk.Entry(self)
        self.entry_name3.place(x=170, y=175)

        btn_cancel = ttk.Button(self, text='Закрыть', command=self.destroy)
        btn_cancel.place(x=300, y=210)

        self.btn_ok = ttk.Button(self, text='Добавить')
        self.btn_ok.place(x=220, y=210)
        self.btn_ok.bind('<Button-1>', lambda event: self.view.records(self.entry_score1.get(), self.entry_name1.get(),
                                                       self.entry_name2.get(), self.entry_score2.get(),
                                                       self.entry_score3.get(), self.entry_score4.get(),
                                                       self.entry_name3.get()))
        self.grab_set()
        self.focus_set()


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

    def init_edit(self):
        self.title("Редактировать дисциплину")
        btn_edit = ttk.Button(self, text="Редактировать")
        btn_edit.place(x=205, y=210)
        btn_edit.bind('<Button-1>', lambda event: self.view.update_record(self.entry_score1.get(), self.entry_name1.get(),
                                                       self.entry_name2.get(), self.entry_score2.get(),
                                                       self.entry_score3.get(), self.entry_score4.get(),
                                                       self.entry_name3.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 дисциплины")
        self.geometry("400x140+400+300")
        self.resizable(False, False)

        label_search = tk.Label(self, text="Поиск дисциплин, по должности")
        label_search.place(x=5, y=20)

        self.entry_search = ttk.Entry(self)
        self.entry_search.place(x=220, 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):
        with sq.connect('PZ_16.db') as self.con:
            self.cur = self.con.cursor()
            self.cur.execute("""CREATE TABLE IF NOT EXISTS sostav (
                nomer INTEGER PRIMARY KEY AUTOINCREMENT,
                FIO TEXT NOT NULL,
                data INTEGER NOT NULL DEFAULT 1,
                dolshnost TEXT NOT NULL,
                stepen TEXT NOT NULL,
                nagruska TEXT NOT NULL,
                zarplata INTEGER NOT NULL DEFAULT 1
                )""")

    def insert_data(self, nomer, FIO, data, dolshnost, stepen, nagruska, zarplata):
        self.cur.execute(
            """INSERT INTO sostav(nomer, FIO, data, dolshnost, stepen, nagruska, zarplata) VALUES (?, ?, ?, ?, ?, ?, ?)""",
            (nomer, FIO, data, dolshnost, stepen, nagruska, zarplata))
```

```
        self.con.commit()


if __name__ == "__main__":
    root = tk.Tk()
    db = DB()
    app = Main(root)
    app.pack()
    root.title("Преподавательский состав")
    root.geometry("1000x500+300+200")
    root.resizable(False, False)
    root.mainloop()
```

Протокол работы программы:



Вывод: я закрепила усвоенные знания, понятия, алгоритмы, основные принципы составления программ, работы с БД в IDE PyCharm Community.