Студент группы ПОКС-21 Иванков Михаил

Практическое занятие №16

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

Цель: Закрепить усвоенные знания, понятия, алгоритмы, основные принципы составления программ, работы с БД вIDE PyCharm Community.

Постановка задачи №1.

Приложение АПТЕКА для автоматизации работы аптечных пунктов. Таблица Лекарственные Средства должна содержать следующую информацию: Код, Название препарата, Применение, Количество, Цена, Страна-производитель. БД должна обеспечить получение информации о лекарственных средствах по названию.

Текст программы №1:

```
#  Приложение АПТЕКА для автоматизации работы аптечных пунктов.
#  Таблица Лекарственные Средства должна содержать следующую информацию:
#  Код, Название препарата, Применение, Количество, Цена, Страна-производитель.
#  БД должна обеспечить получение информации о лекарственных средствах по названию.
import tkinter as tk
from tkinter import ttk
import sqlite3 as sq


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='#00a8f3', bd=4)
        toolbar.pack(side=tk.TOP, fill=tk.X)

        self.add_img = tk.PhotoImage(file="11.gif")
        self.btn_open_dialog = tk.Button(toolbar, text='Добавить препарат',
command=self.open_dialog, bg='#00a8f3',
                                         bd=0, compound=tk.TOP, image=self.add_img)
        self.btn_open_dialog.pack(side=tk.LEFT)

        self.update_img = tk.PhotoImage(file="12.gif")
        btn_edit_dialog = tk.Button(toolbar, text="Редактировать",
command=self.open_update_dialog, bg='#00a8f3',
                                    bd=0, compound=tk.TOP, image=self.update_img)
        btn_edit_dialog.pack(side=tk.LEFT)

        self.delete_img = tk.PhotoImage(file="13.gif")
        btn_delete = tk.Button(toolbar, text="Удалить запись",
command=self.delete_records, bg='#00a8f3',
                               bd=0, compound=tk.TOP, image=self.delete_img)
        btn_delete.pack(side=tk.LEFT)

        self.search_img = tk.PhotoImage(file="14.gif")
        btn_search = tk.Button(toolbar, text="Поиск записи",
command=self.open_search_dialog, bg='#00a8f3',
                               bd=0, compound=tk.TOP, image=self.search_img)
```

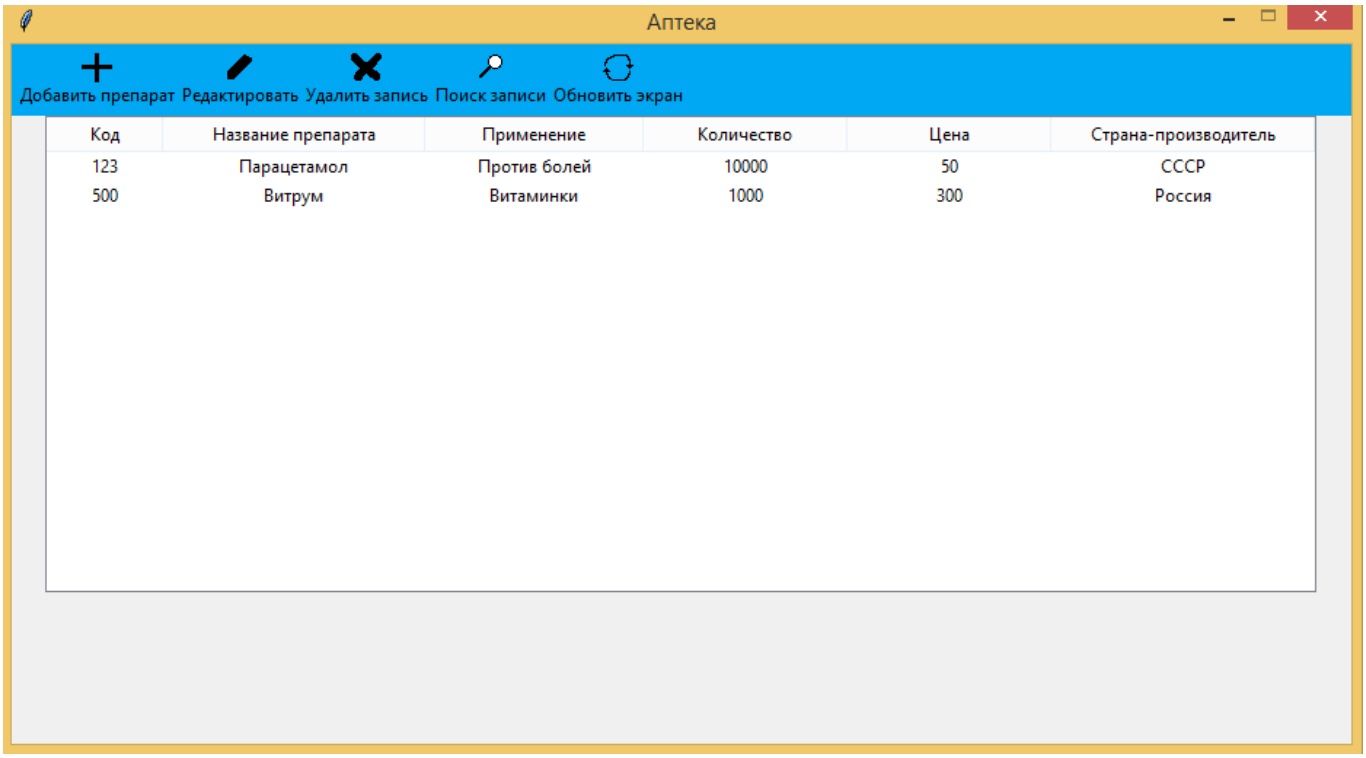
```
        btn_search.pack(side=tk.LEFT)

        self.refresh_img = tk.PhotoImage(file="15.gif")
        btn_refresh = tk.Button(toolbar, text="Обновить экран",
command=self.view_records, bg='#00a8f3',
                                bd=0, compound=tk.TOP, image=self.refresh_img)
        btn_refresh.pack(side=tk.LEFT)

        self.tree = ttk.Treeview(self, columns=('user_id', 'title', 'Application',
'quantity', 'price', 'country'),
                                 height=15, show='headings')

        self.tree.column('user_id', width=80, anchor=tk.CENTER)
        self.tree.column('title', width=180, anchor=tk.CENTER)
        self.tree.column('Application', width=150, anchor=tk.CENTER)
        self.tree.column('quantity', width=140, anchor=tk.CENTER)
        self.tree.column('price', width=140, anchor=tk.CENTER)
        self.tree.column('country', width=180, anchor=tk.CENTER)

        self.tree.heading('user_id', text='Код')
        self.tree.heading('title', text='Название препарата')
        self.tree.heading('Application', text='Применение')
        self.tree.heading('quantity', text='Количество')
        self.tree.heading('price', text='Цена')
        self.tree.heading('country', text='Страна-производитель')

        self.tree.pack()

    def records(self, user_id, title, Application, quantity, price, country):
        self.db.insert_data(user_id, title, Application, quantity, price, country)
        self.view_records()

    def update_record(self, user_id, title, Application, quantity, price, country):
        self.db.cur.execute("""UPDATE susers SET user_id=?, title=?, Application=?,
quantity=?, price=?,country=?
        WHERE user_id=?""",
                            (user_id, title, Application, quantity, price, country,
                             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 susers""")
        [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 susers WHERE user_id=?""",
(self.tree.set(selection_item, '#1'),))
        self.db.con.commit()
        self.view_records()

    def search_records(self, title):
        title = (title,)
        self.db.cur.execute("""SELECT * FROM susers WHERE title>=?""", title)
        [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):

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

    def init_child(self):
        self.title('Добавить препарат')
        self.geometry('400x320+400+600')
        self.resizable(False, False)

        label_user_id = tk.Label(self, text='Код')
        label_user_id.place(x=30, y=25)
        self.entry_user_id = ttk.Entry(self)
        self.entry_user_id.place(x=160, y=25)

        label_name = tk.Label(self, text='Название препарата')
        label_name.place(x=30, y=50)
        self.entry_title = ttk.Entry(self)
        self.entry_title.place(x=160, y=50)

        label_second = tk.Label(self, text='Применение')
        label_second.place(x=30, y=75)
        self.entry_Application = ttk.Entry(self)
        self.entry_Application.place(x=160, y=75)

        label_quantity = tk.Label(self, text='Количество')
        label_quantity.place(x=30, y=100)
        self.entry_quantity = ttk.Entry(self)
        self.entry_quantity.place(x=160, y=100)

        label_old = tk.Label(self, text='Цена')
        label_old.place(x=30, y=125)
        self.entry_price = ttk.Entry(self)
        self.entry_price.place(x=160, y=125)

        label_prof = tk.Label(self, text='Страна-производитель')
        label_prof.place(x=30, y=150)
        self.entry_country = ttk.Entry(self)
        self.entry_country.place(x=160, y=150)

        btn_cancel = ttk.Button(self, text='Закрыть', command=self.destroy)
        btn_cancel.place(x=300, y=220)

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

self.entry_title.get(),

self.entry_Application.get(),

self.entry_quantity.get(),

self.entry_price.get(),

self.entry_country.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 запись")
        btn_edit = ttk.Button(self, text="Редактировать")
        btn_edit.place(x=205, y=220)
        btn_edit.bind('<Button-1>', lambda event:
self.view.update_record(self.entry_user_id.get(),

self.entry_title.get(),

self.entry_Application.get(),

self.entry_quantity.get(),

self.entry_price.get(),

self.entry_country.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="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('test.db') as self.con:
            self.cur = self.con.cursor()
            self.cur.execute("""CREATE TABLE IF NOT EXISTS susers (
                user_id INTEGER,
                title TEXT NOT NULL,
                Application TEXT NOT NULL,
                quantity INTEGER NOT NULL DEFAULT 1,
                price INTEGER,
                country TEXT NOT NULL
                )""")

    def insert_data(self, user_id, title, Application, quantity, price, country):
        self.cur.execute("""INSERT INTO susers(user_id, title, Application, quantity,
price, country)
```

```
        VALUES (?, ?, ?, ?, ?, ?)""", (user_id, title, Application, quantity, price,
country))
        self.con.commit()


if __name__ == "__main__":
    root = tk.Tk()
    db = DB()
    app = Main(root)
    app.pack()
    root.title("Аптека")
    root.geometry("920x480")
    root.resizable(False, False)
    root.mainloop()
```

Протокол работы программы №1:



Студент группы ПОКС-21 Иванков Михаил

Вывод: В процессе выполнения 16 практического занятия я закрепил усвоенные знания, понятия, алгоритмы, основные принципы составления программ, приобрел навыки составления программ с БД в IDE PyCharm Community.
Было использовано ООП, а также работа в DBeaver.

Выполнены разработка кода, отладка, тестирование, оптимизация программного кода.

Готовый программный код выложен на GitHub.