Семерунин Владимир ПОКС-22

Практическая работа №16

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 ПРОКАТ АВТОМОБИЛЕЙ для некоторой организации. БД должна содержать таблицу Клиент со следующей структурой записи: ФИО клиента, марка авто, срок проката, сумма, предоплата (да/нет). БД должна обеспечивать получение информации по сумме

Текст программы:

```python
# V23
# Приложение ПРОКАТ АВТОМОБИЛЕЙ для некоторой организации. БД должна
# содержать таблицу Клиент со следующей структурой записи: ФИО клиента, марка авто,
# срок проката, сумма, предоплата (да/нет).
# БД должна обеспечивать получение информации по сумме.
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='#f09ca4', bd=4)
        toolbar.pack(side=tk.TOP, fill=tk.X)

        self.add_img = tk.PhotoImage(file="img/добавить.png")
        self.btn_open_dialog = tk.Button(toolbar, text='Добавить клиента', command=self.open_dialog, bg='#C0C0C0', bd=0,
                                         compound=tk.TOP,
                                         image=self.add_img)

        self.btn_open_dialog.pack(side=tk.LEFT, padx=2)

        self.update_img = tk.PhotoImage(file="img/редактировать.png")
        btn_edit_dialog = tk.Button(
            toolbar,
            text="Редактировать",
            command=self.open_update_dialog,
            bg='#C0C0C0',
            bd=0, compound=tk.TOP,
            image=self.update_img
        )
```

```
        btn_edit_dialog.pack(side=tk.LEFT, padx=2)

        self.delete_img = tk.PhotoImage(file="img/удалить.png")
        btn_delete = tk.Button(toolbar, text="Удалить запись", command=self.delete_records, bg='#C0C0C0',
                               bd=0, compound=tk.TOP, image=self.delete_img)
        btn_delete.pack(side=tk.LEFT, padx=2)

        self.search_img = tk.PhotoImage(file="img/поиск.png")
        btn_search = tk.Button(toolbar, text="Поиск записи", command=self.open_search_dialog, bg='#C0C0C0',
                               bd=0, compound=tk.TOP, image=self.search_img)
        btn_search.pack(side=tk.LEFT, padx=2)

        self.fresh_img = tk.PhotoImage(file="img/обновить.png")
        btn_fresh = tk.Button(toolbar, text="Обновить экран", command=self.view_records, bg='#C0C0C0',
                              bd=0, compound=tk.TOP, image=self.fresh_img)
        btn_fresh.pack(side=tk.LEFT, padx=2)

        self.tree = ttk.Treeview(
                                 self, columns=('klient_id', 'name', 'marka', 'prokat', 'sum', 'oplata'),
                                 height=15, show='headings')

        self.tree.column('klient_id', width=70, anchor=tk.CENTER)
        self.tree.column('name', width=180, anchor=tk.CENTER)
        self.tree.column('marka', width=140, anchor=tk.CENTER)
        self.tree.column('prokat', width=140, anchor=tk.CENTER)
        self.tree.column('sum', width=140, anchor=tk.CENTER)
        self.tree.column('oplata', width=140, anchor=tk.CENTER)

        self.tree.heading('klient_id', text='ID')
        self.tree.heading('name', text='Фамилия клиента')
        self.tree.heading('marka', text='Марка машины')
        self.tree.heading('prokat', text='Срок проката(мес)')
        self.tree.heading('sum', text='Сумма($)')
        self.tree.heading('oplata', text='Предоплата')

```

```
        self.tree.pack()

    def records(self, klient_id, name, marka, prokat, sum, oplata):
        self.db.insert_data(klient_id, name, marka, prokat, sum, oplata)
        self.view_records()

    def update_record(self, klient_id, name, marka, prokat, sum, oplata):
        self.db.cur.execute(
            """UPDATE klient SET klient_id=?, name=?, marka=?, prokat=?, sum=?, oplata=? WHERE klient_id=?""",
            (klient_id, name, marka, prokat, sum, oplata, 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 klient""")
        [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 klient WHERE klient_id = ?""",
                (self.tree.set(selection_item, '#1'),)
            )
        self.db.con.commit()
        self.view_records()

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

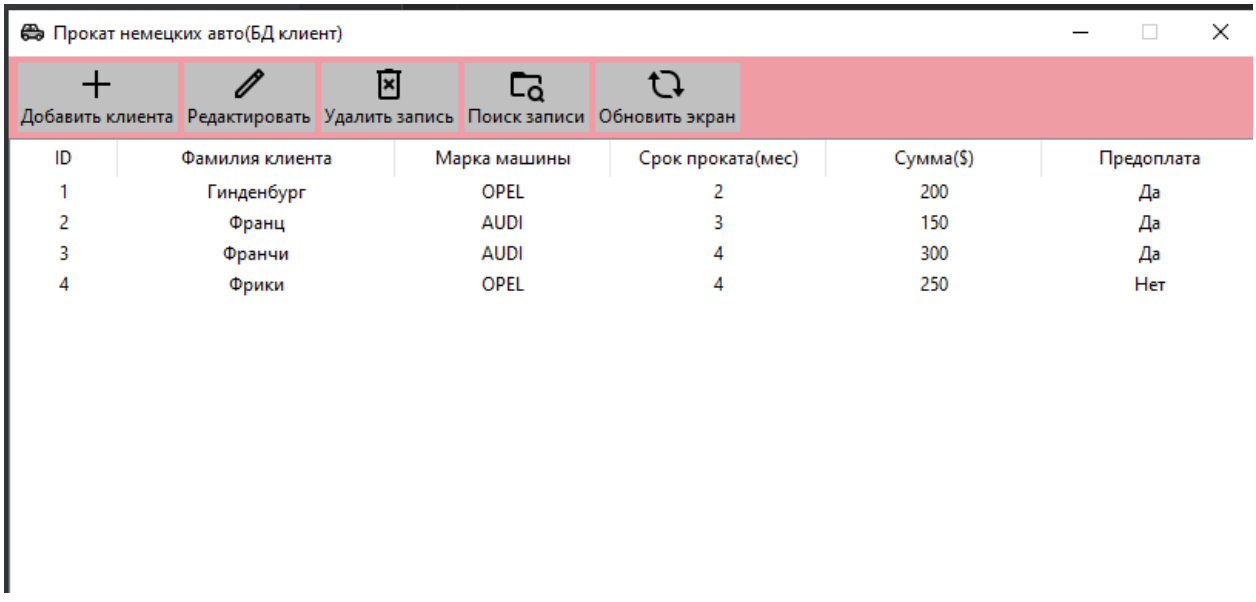
```
110
111     def open_update_dialog(self):
112         self.db.cur.execute("SELECT * FROM klient WHERE klient_id=?",
113                             self.tree.set([n for n in self.tree.selection()][0], '#1'))
114         Update(self.db.cur.fetchall()[0])
115
116     def open_search_dialog(self):
117         Search()
118
119
120 class Child(tk.Toplevel):
121     """Класс для дочернего окна"""
122     entry_delete = {}
123
124     def __init__(self, root, app):
125         super().__init__(root)
126         self.init_child()
127         self.view = app
128
129     def init_child(self):
130         self.title('Добавить клиента')
131         self.geometry('400x220+400+300')
132         self.resizable(False, False)
133
134         self.label_id = tk.Label(self, text='№ Клиента')
135         self.label_id.place(x=50, y=1)
136         self.entry_klient_id = ttk.Entry(self)
137         self.entry_klient_id.place(x=170, y=1)
138
139         label_description = tk.Label(self, text='Фамилия клиента')
140         label_description.place(x=50, y=25)
141         self.entry_name = ttk.Entry(self)
142         self.entry_name.place(x=170, y=25)
143
144         label_name = tk.Label(self, text='Марка авто')
```

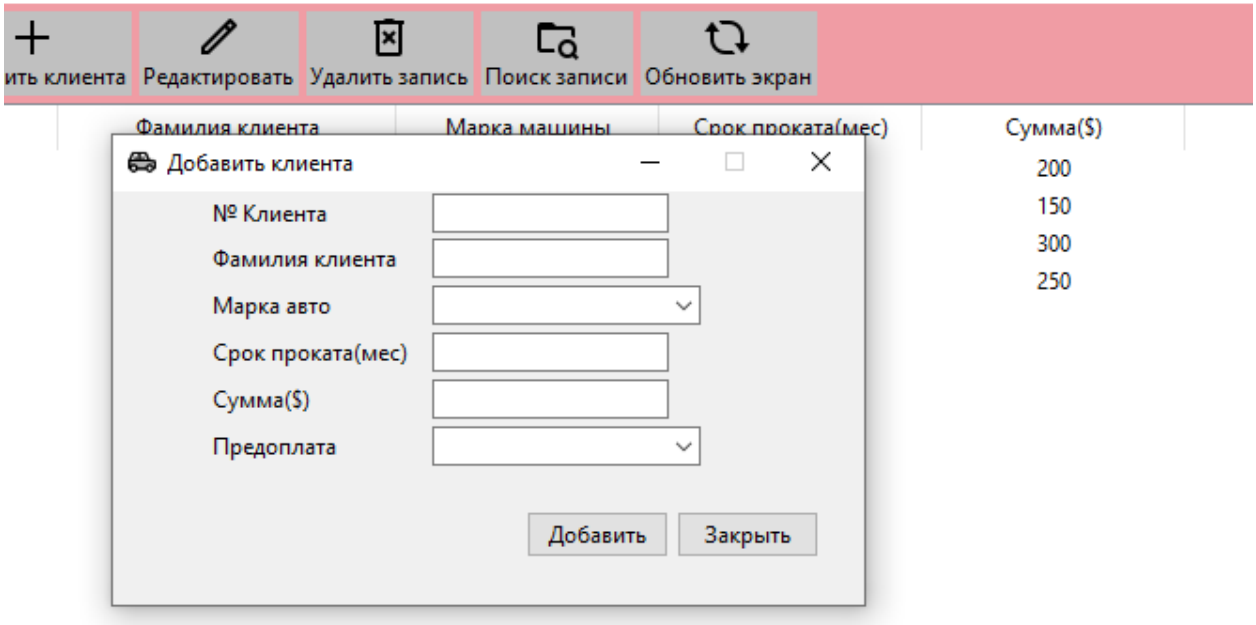
```
        label_name.place(x=50, y=50)
        self.combobox_marka = ttk.Combobox(self, values=[u'BMW', u'AUDI', u'MERCEDES-BENZ', u'VOLKSWAGEN', u'PORSCHE',
                                                          u'OPEL', u'Bitter', u'GUMPERT', u'ISDERA',
                                                          u'KEINATH', u'Adler'])
        self.combobox_marka.place(x=170, y=50)

        label_sex = tk.Label(self, text='Срок проката(мес)')
        label_sex.place(x=50, y=75)
        self.entry_prokat = ttk.Entry(self)
        self.entry_prokat.place(x=170, y=75)

        label_old = tk.Label(self, text='Сумма($)')
        label_old.place(x=50, y=100)
        self.entry_sum = ttk.Entry(self)
        self.entry_sum.place(x=170, y=100)

        label_old = tk.Label(self, text='Предоплата')
        label_old.place(x=50, y=125)
        self.combobox_oplata = ttk.Combobox(self, values=[u'Да', u'Нет'])
        self.combobox_oplata.place(x=170, y=125)

        btn_cancel = ttk.Button(self, text='Закрыть', 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_klient_id.get(),
                                                                       self.entry_name.get(),
                                                                       self.combobox_marka.get(),
                                                                       self.entry_prokat.get(),
                                                                       self.entry_sum.get(),
                                                                       self.combobox_oplata.get()))
        self.grab_set()
        self.focus_set()
```

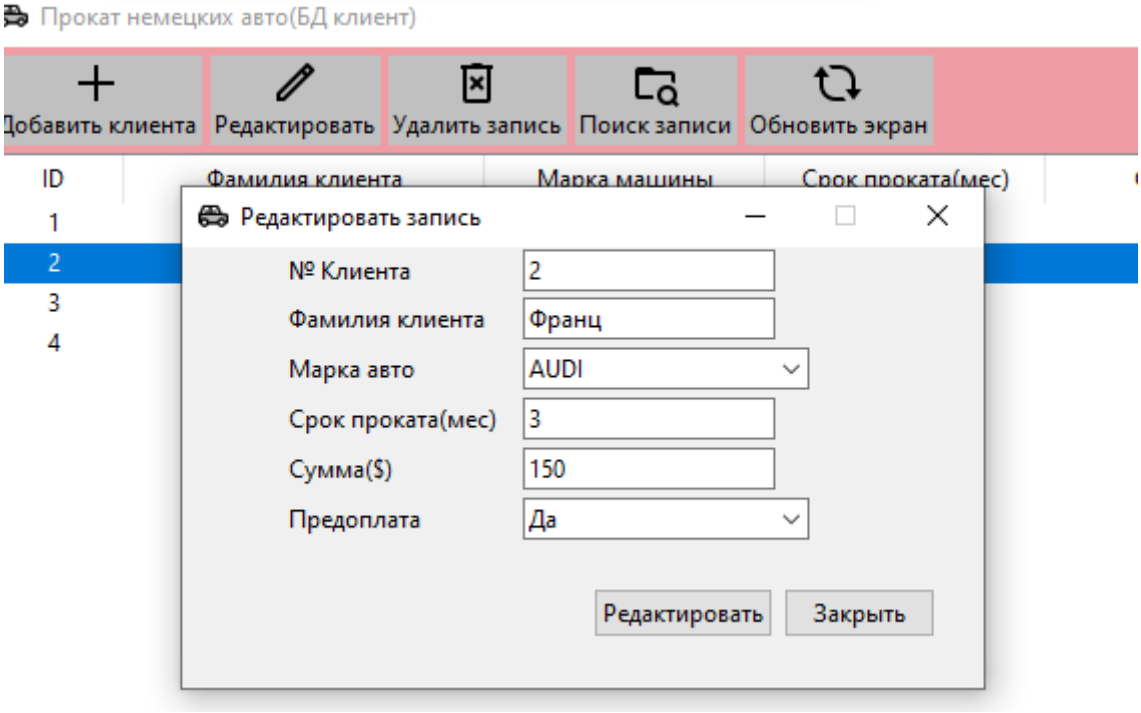
```
class Update(Child):
    def __init__(self, zm):
        super().__init__(root, app)
        self.view = app
        self.entry_klient_id.insert(0, zm[0])
        self.entry_name.insert(0, zm[1])
        self.combobox_marka.insert(0, zm[2])
        self.entry_prokat.insert(0, zm[3])
        self.entry_sum.insert(0, zm[4])
        self.combobox_oplata.insert(0, zm[5])

        self.title("Редактировать запись")
        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_klient_id.get(),
                            self.entry_name.get(),
                            self.combobox_marka.get(),
                            self.entry_prokat.get(),
                            self.entry_sum.get(),
                            self.combobox_oplata.get()
                        )
                    )
        self.btn_ok.destroy()


class Search(tk.Toplevel):
    def __init__(self):
        super().__init__()
        self.view = app

        self.title("Поиск")
        self.geometry("250x150+400+300")
        self.configure(bg='#C0C0C0')
        self.resizable(False, False)

        self.label_search = tk.Label(self, text="Сумма\nпроката\n(>)", bg='#C0C0C0')
        self.label_search.place(x=10, y=20)
```

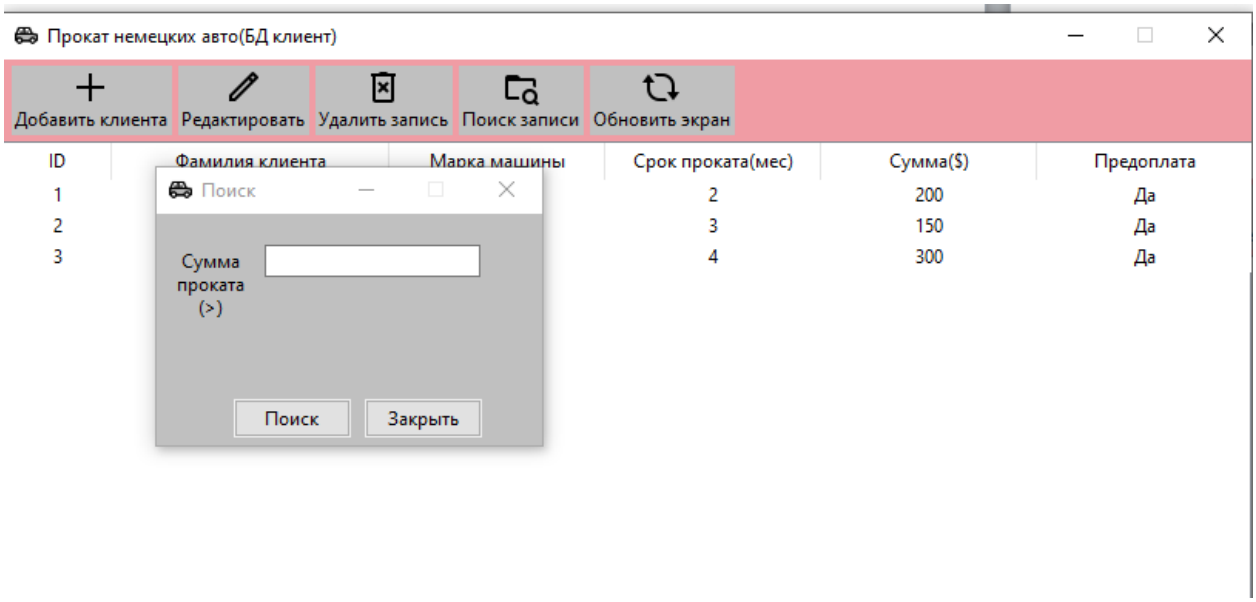
```
        self.entry_search = ttk.Entry(self)
        self.entry_search.place(x=70, y=20, width=140)

        btn_cancel = ttk.Button(self, text="Закрыть", command=self.destroy)
        btn_cancel.place(x=135, y=120)

        btn_search = ttk.Button(self, text="Поиск")
        btn_search.place(x=50, y=120)
        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('klient.db') as self.con:
            self.cur = self.con.cursor()
            self.cur.execute("""CREATE TABLE IF NOT EXISTS klient (
                        klient_id INTEGER PRIMARY KEY AUTOINCREMENT,
                        name TEXT NOT NULL,
                        marka TEXT NOT NULL,
                        prokat INTEGER,
                        sum INTEGER,
                        oplata TEXT NOT NULL)""")

    def insert_data(self, klient_id, name, marka, prokat, sum, oplata):
        self.cur.execute(
            """INSERT INTO klient (klient_id, name, marka, prokat, sum, oplata) VALUES (?,?, ?, ?, ?, ?)""",
            (klient_id, name, marka, prokat, sum, oplata))
        self.con.commit()


if __name__ == "__main__":
    root = tk.Tk()
    db = DB()
    app = Main(root)
    app.pack()
    root.title("Прокат немецких авто(БД клиент) ")
    root.geometry("810x350+300+200")
    root.iconphoto(True, tk.PhotoImage(file="img/машинка.png"))
```

Протокол работы:




ить клиента
Редактировать
Удалить запись
Поиск записи
Обновить экран
Фамилия клиента
Марка машины
Срок проката(мес)
Сумма($)
200
150
300
250
Добавить клиента
№ Клиента
Фамилия клиента
Марка авто
Срок проката(мес)
Сумма($)
Предоплата
Добавить
Закрыть


Прокат немецких авто(БД клиент)
Добавить клиента
Редактировать
Удалить запись
Поиск записи
Обновить экран
ID
Фамилия клиента
Марка машины
Срок проката(мес)
1
2
3
4
Редактировать запись
№ Клиента
2
Фамилия клиента
Франц
Марка авто
AUDI
Срок проката(мес)
3
Сумма($)
150
Предоплата
Да
Редактировать
Закрыть



Вывод: В ходе выполнения практической работы закрепил усвоенные знания, понятия, алгоритмы, основные принципы составления программ, работы с БД в IDE PyCharm Community.