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 в IDE PyCharm Community

Постановка задачи

1.

Приложение СТРАХОВАЯ КОМПАНИЯ для некоторой организации. БД должна содержать таблицу Договор со следующей структурой записи: дата заключения, страховая сумма, вид страхования, тарифная ставка и филиал, в котором заключался договор. БД должна обеспечивать получение информации о договорах по страховой сумме.

Текст программы:

1.

```
# Приложение СТРАХОВАЯ КОМПАНИЯ для некоторой организации. БД должна содержать
таблицу Договор со следующей структурой
# записи: дата заключения, страховая сумма, вид страхования, тарифная ставка и
филиал, в котором заключался договор.
# БД должна обеспечивать получение информации о договорах по страховой сумме.
import tkinter as tk
from tkinter import ttk
import sqlite3 as sq
from tkcalendar import DateEntry
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='#a0dea0', bd=4)
        toolbar.pack(side=tk.TOP, fill=tk.X)
        self.add img = tk.PhotoImage(file="pres.gif")
        self.btn open dialog = tk.Button(toolbar, text='Добавить договор',
command=self.open dialog, bg='#5da130', bd=0,
                                     compound=tk.TOP, image=self.add img)
        self.btn open dialog.pack(side=tk.LEFT)
        self.update img = tk.PhotoImage(file="red.gif")
        btn edit dialog = tk.Button(toolbar, text="Редактировать",
command=self.open update dialog, bg='#5da130',
                                    bd=0, compound=tk.TOP,
image=self.update img)
        btn edit dialog.pack(side=tk.LEFT)
        self.delete img = tk.PhotoImage(file="bin.qif")
        btn delete = tk.Button(toolbar, text="Удалить договор",
command=self.delete records, bg='#5da130',
                                    bd=0, compound=tk.TOP,
image=self.delete img)
        btn delete.pack(side=tk.LEFT)
```

```
self.search img = tk.PhotoImage(file="sea.gif")
        btn_search = tk.Button(toolbar, text="Поиск договора",
command=self.open search dialog, bg='#5da130',
                               bd=0, compound=tk.TOP, image=self.search img)
       btn search.pack(side=tk.LEFT)
        self.refresh img = tk.PhotoImage(file="upd.gif")
        btn refresh = tk.Button(toolbar, text="Обновить экран",
command=self.view records, bg='#5da130',
                               bd=0, compound=tk.TOP, image=self.refresh img)
       btn refresh.pack(side=tk.LEFT)
        self.tree = ttk.Treeview(self, columns=('id','date', 'sum', 'insurance',
'bid', 'filial'), height=15, show='headings')
        self.tree.column('id', width=170, anchor=tk.CENTER)
        self.tree.column('date', width=180, anchor=tk.CENTER)
        self.tree.column('sum', width=180, anchor=tk.CENTER)
        self.tree.column('insurance', width=250, anchor=tk.CENTER)
        self.tree.column('bid', width=140, anchor=tk.CENTER)
        self.tree.column('filial', width=140, anchor=tk.CENTER)
       self.tree.heading('id', text='ID')
        self.tree.heading('date', text='Дата заключения')
        self.tree.heading('sum', text='CTpaxoBas cymma')
        self.tree.heading('insurance', text='Вид страхования')
        self.tree.heading('bid', text='Тарифная ставка')
        self.tree.heading('filial', text='Филиал')
        self.tree.pack()
    def records(self, id, date, sum, insurance, bid, filial):
        self.db.insert data( id, date, sum, insurance, bid, filial)
        self.view records()
    def update record(self, id, date, sum, insurance, bid, filial):
        self.db.cur.execute("""UPDATE users SET id =?, date=?, sum=?,
insurance=?, bid=? WHERE id=?""",
                            (id, date, sum, insurance, bid, filial,
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 users""")
        [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 users WHERE id=?""",
(self.tree.set(selection item, '#1'),))
        self.db.con.commit()
        self.view records()
    # def search records(self, user id):
    # user id = ("%" + user id + "%",)
         self.db.cur.execute("""SELECT * FROM users WHERE name LIKE ?""",
    #
user id)
         [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 search records(self, bid):
       bid = (bid,)
        self.db.cur.execute("""SELECT * FROM users WHERE bid >= ?""", bid)
```

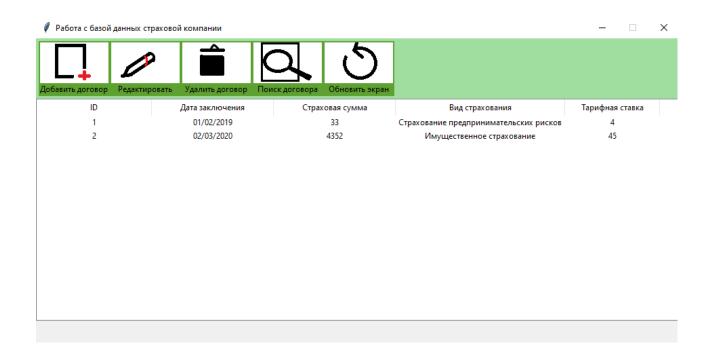
```
[self.tree.delete(i) for i in self.tree.get children()]
        [self.tree.insert('', 'end', values=row) for row in
self.db.cur.fetchall()]
    def open dialog(self):
        Child(root, app)
    def open update dialog(self):
        Update()
    def open search dialog(self):
class Child(tk.Toplevel):
    """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 договор')
        self.geometry('600x320+400+300')
        self.resizable(False, False)
        label description id = tk.Label(self, text='ID')
        label description id.place(x=50, y=15)
        self.entry description id = ttk.Entry(self)
        self.entry description id.place(x=210, y=15)
        label description = tk.Label(self, text='Дата заключения')
        label description.place (x=50, y=40)
        self.entry description = DateEntry(self, width = 12)
self.entry description.place(x=210, y=40)
        label name = tk.Label(self, text='CTpaxoBag cymma')
        label name.place(x=50, y=75)
        self.entry name = ttk.Entry(self)
        self.entry name.place(x=210, y=75)
        label sex = tk.Label(self, text='Вид страхования')
        label sex.place(x=50, y=100)
        self.combobox = ttk.Combobox(self, values=[u'Страхование
предпринимательских рисков', u'Имущественное страхование', u'Личное страхование', u'Страхование ответственности'], width=40)
        self.combobox.current(0)
        self.combobox.place(x=210, y=100)
        label_old = tk.Label(self, text='Тарифная ставка')
        label_old.place(x=50, y=125)
        self.entry old = ttk.Entry(self)
        self.entry old.place(x=210, y=125)
        label score = tk.Label(self, text='Филиал')
        label score.place(x=50, y=150)
        self.entry_score = ttk.Entry(self)
        self.entry score.place(x=210, y=150)
        btn cancel = ttk.Button(self, text='Закрыть', command=self.destroy)
        btn cancel.place (x=300, y=175)
        self.btn ok = ttk.Button(self, text='Добавить')
        self.btn ok.place(x=220, y=175)
        self.btn ok.bind('<Button-1>', lambda event:
```

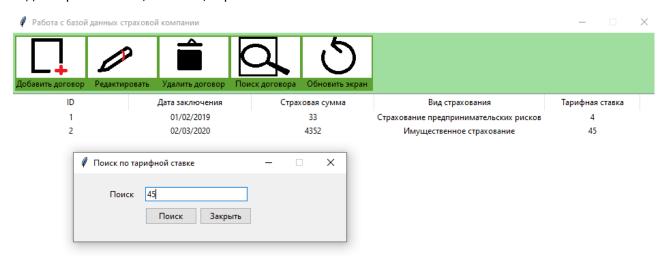
```
self.view.records(self.entry description id.get(),
self.entry description.get(),
self.entry name.get(),
self.combobox.get(),
self.entry old.get(),
self.entry score.get()))
        self.grab set()
        self.focus set()
class Update(Child):
    def __init__(self):
        super().__init__(root, 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=170)
        btn edit.bind('<Button-1>', lambda event:
self.view.update record(self.entry description id.get(),
self.entry description.get(),
self.entry name.get(),
self.combobox.get(),
self.entry old.get(),
self.entry score.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 по тарифной ставке")
        self.geometry("400x100+400+300")
        self.resizable(False, False)
        label search = tk.Label(self, text="Πομςκ")
        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):
```

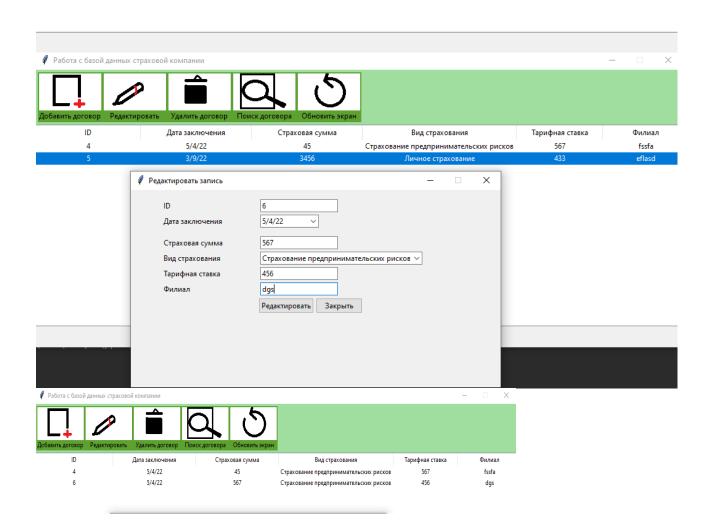
```
with sq.connect('company.db') as self.con:
            self.cur = self.con.cursor()
            self.cur.execute("""CREATE TABLE IF NOT EXISTS users (
                id INTEGER PRIMARY KEY AUTOINCREMENT,
                date DATETIME,
                sum INTEGER NOT NULL,
                insurance TEXT NOT NULL,
                bid INTEGER,
                filial TEXT
                ) " " " )
   def insert data(self,id, date,sum, insurance, bid, filial):
        self.cur.execute("""INSERT INTO users(id, date, sum, insurance, bid,
filial) VALUES (?, ?, ?, ?, ?, ?)""",
                             (id, date, sum, insurance, bid, filial))
        self.con.commit()
if __name__ == "__main ":
   root = tk.Tk()
   db = DB()
   app = Main(root)
   app.pack()
   root.title("Работа с базой данных страховой компании")
   root.geometry("950x450+300+200")
   root.resizable(False, False)
```

Протокол работы программы:

1.







Вывод: в процессе выполнения практического занятия выработала навыки составления программ с применением ткинтера, БД и знаний ООП в IDE PyCharm Community. Выполнены разработка кода, отладка, тестирование, оптимизация программногокода. Готовые программные коды выложены на GitHub.