Сделал Мезерный Владимир ПОКС-22

**Практическая работа № 16**

**Тема:**Разработка многооконного приложения для

работы с однотабличной БД в I**DE PyCharm Community.**

**Цель:** закрепить усвоенные знания, понятия, алгоритмы, основные принципы составления

программ, работы с БД в **IDE PyCharm Community.**

**Постановка задачи:**

Приложение ТОРГОВАЯ ФИРМА для автоматизированного контроля продаж

товаров торговой фирмы. БД должна содержать таблицу Продажа товаров со следующей

структурой записи: Дата продажи, Товар, Сумма, Скидка, Филиал, Менеджер.

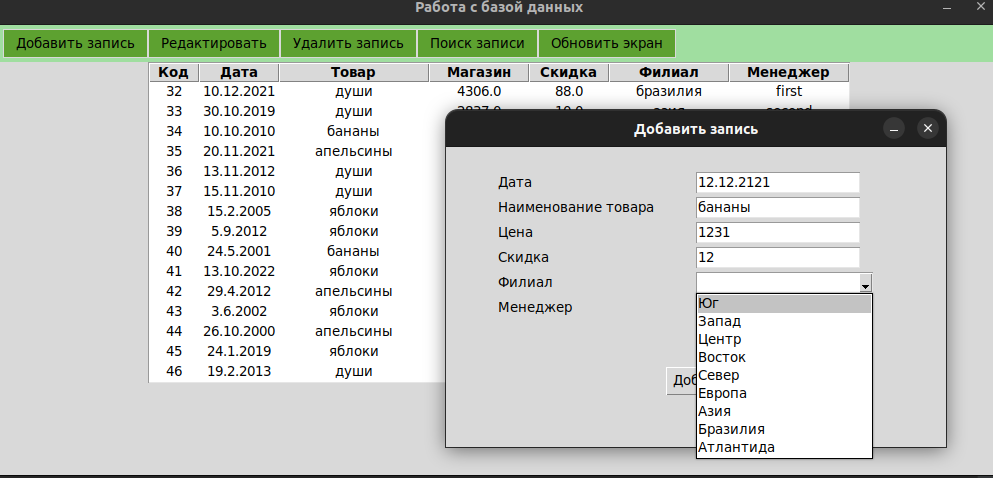
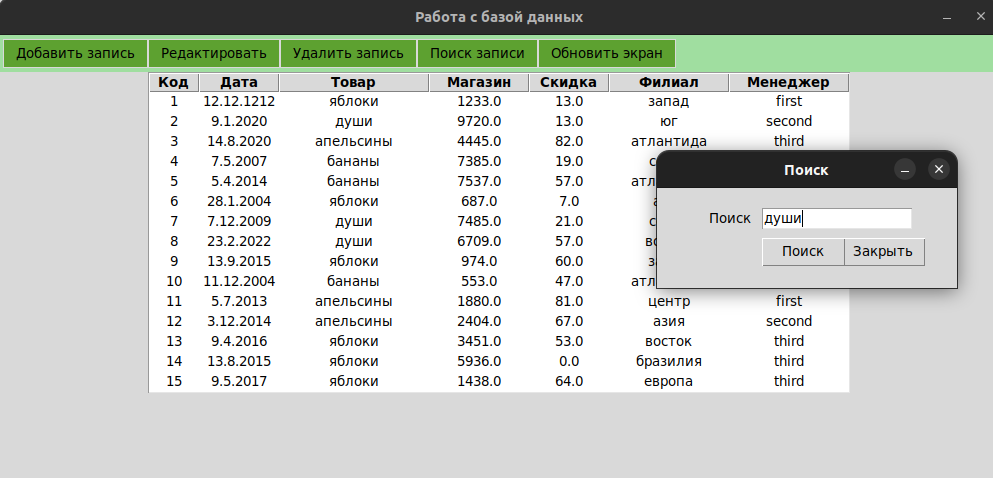
БД должна обеспечивать получение информации о продаже товаров по

наименованию товара.

**Код:**

# Приложение ТОРГОВАЯ ФИРМА для автоматизированного контроля продаж  
# товаров торговой фирмы. БД должна содержать таблицу Продажа товаров со следующей  
# структурой записи: Дата продажи, Товар, Сумма, Скидка, Филиал, Менеджер.  
# БД должна обеспечивать получение информации о продаже товаров по  
# наименованию товара.  
  
import tkinter as tk  
from tkinter import ttk  
import sqlite3 as sq  
  
  
class Main(tk.Frame):  
 *"""Main window class"""* def \_\_init\_\_(self, roottt):  
 super().\_\_init\_\_(roottt)  
 self.btn\_open\_dialog = None  
 self.tree = None  
 self.init\_main()  
 self.db = db  
 self.view\_records()  
  
 def init\_main(self): # Main window  
 toolbar = tk.Frame(bg='#a0dea0', bd=4)  
 toolbar.pack(side=tk.TOP, fill=tk.X)  
  
 self.btn\_open\_dialog = tk.Button(toolbar, text='Добавить запись', command=self.open\_dialog, bg='#5da130', bd=0,  
 compound=tk.TOP)  
 self.btn\_open\_dialog.pack(side=tk.LEFT)  
  
 btn\_edit\_dialog = tk.Button(toolbar, text="Редактировать", command=self.open\_update\_dialog, bg='#5da130',  
 bd=0, compound=tk.TOP)   
 btn\_edit\_dialog.pack(side=tk.LEFT)  
  
 btn\_delete = tk.Button(toolbar, text="Удалить запись", command=self.delete\_records, bg='#5da130',  
 bd=0, compound=tk.TOP)  
 btn\_delete.pack(side=tk.LEFT)  
  
 btn\_search = tk.Button(toolbar, text="Поиск записи", command=self.open\_search\_dialog, bg='#5da130',  
 bd=0, compound=tk.TOP)  
 btn\_search.pack(side=tk.LEFT)  
  
 btn\_refresh = tk.Button(toolbar, text="Обновить экран", command=self.view\_records, bg='#5da130',  
 bd=0, compound=tk.TOP)  
 btn\_refresh.pack(side=tk.LEFT)  
  
 self.tree = ttk.Treeview(self, columns=( # Tree making  
 'id', 'data', 'tovar', 'cost', 'skidka', 'filial', 'manager'), height=15,  
 show='headings')  
  
 self.tree.column('id', width=50, anchor=tk.CENTER)  
 self.tree.column('data', width=80, anchor=tk.CENTER)  
 self.tree.column('tovar', width=150, anchor=tk.CENTER)  
 self.tree.column('cost', width=100, anchor=tk.CENTER)  
 self.tree.column('skidka', width=80, anchor=tk.CENTER)  
 self.tree.column('filial', width=120, anchor=tk.CENTER)  
 self.tree.column('manager', width=120, anchor=tk.CENTER)  
  
 self.tree.heading('id', text='Код')  
 self.tree.heading('data', text='Дата')  
 self.tree.heading('tovar', text='Товар')  
 self.tree.heading('cost', text='Магазин')  
 self.tree.heading('skidka', text='Скидка')  
 self.tree.heading('filial', text='Филиал')  
 self.tree.heading('manager', text='Менеджер')  
  
 self.tree.pack(side=tk.BOTTOM)  
  
 def records(self, data, tovar, cost, skidka, filial, manager): # inserting data  
 self.db.insert\_data(data, tovar, cost, skidka, filial, manager)  
 self.view\_records()  
  
 def update\_record(self, data, tovar, cost, skidka, filial, manager): # updating data  
 self.db.cur.execute("""UPDATE tovarDB SET data=?, tovar=?, cost=?, skidka=?, filial=?, manager=?   
 WHERE id=?""", (data, tovar.lower(), cost, skidka, filial.lower(), manager.lower(),  
 self.tree.set(self.tree.selection()[0], '#1')))  
 self.db.con.commit()  
 self.view\_records()  
  
 def view\_records(self): # Watch data  
 self.db.cur.execute("""SELECT \* FROM tovarDB""")  
 [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): # delete data  
 for selection\_item in self.tree.selection():  
 self.db.cur.execute("""DELETE FROM tovarDB WHERE id=?""", (self.tree.set(selection\_item, '#1'),))  
 self.db.con.commit()  
 self.view\_records()  
  
 def search\_records(self, tovar\_name): # search data  
 self.db.cur.execute(f"""SELECT \* FROM tovarDB WHERE tovar LIKE '%{tovar\_name.lower()}%'""")  
 [self.tree.delete(i) for i in self.tree.get\_children()]  
 [self.tree.insert('', 'end', values=row) for row in self.db.cur.fetchall()]  
  
 @staticmethod  
 def open\_dialog():  
 Child(root, app)  
  
 @staticmethod  
 def open\_update\_dialog():  
 Update()  
  
 @staticmethod  
 def open\_search\_dialog():  
 Search()  
  
  
class Child(tk.Toplevel):  
 *"""Дочка"""* def \_\_init\_\_(self, roott, appp):  
 super().\_\_init\_\_(roott)  
 self.btn\_ok = None  
 self.entry\_manager = None  
 self.entry\_filial = None  
 self.entry\_skidka = None  
 self.var = None  
 self.entry\_bid = None  
 self.entry\_name\_shop = None  
 self.entry\_name\_tovar = None  
 self.entry\_id = None  
 self.init\_child()  
 self.view = appp  
  
 def init\_child(self): # adding new  
 self.title('Добавить запись')  
 self.geometry('500x300+400+300')  
 self.resizable(False, False)  
  
 label\_data = tk.Label(self, text='Дата')  
 label\_data.place(x=50, y=25)  
 self.entry\_id = ttk.Entry(self)  
 self.entry\_id.place(x=250, y=25)  
  
 label\_name\_tovar = tk.Label(self, text='Наименование товара')  
 label\_name\_tovar.place(x=50, y=50)  
 self.entry\_name\_tovar = ttk.Entry(self)  
 self.entry\_name\_tovar.place(x=250, y=50)  
  
 label\_name\_shop = tk.Label(self, text='Цена')  
 label\_name\_shop.place(x=50, y=75)  
 self.entry\_name\_shop = ttk.Entry(self)  
 self.entry\_name\_shop.place(x=250, y=75)  
  
 label\_skidka = tk.Label(self, text='Скидка')  
 label\_skidka.place(x=50, y=100)  
 self.entry\_skidka = ttk.Entry(self)  
 self.entry\_skidka.place(x=250, y=100)  
  
 label\_filial = tk.Label(self, text='Филиал')  
 label\_filial.place(x=50, y=125)  
 self.entry\_filial = ttk.Combobox(self, values=[u'Юг', u'Запад', u'Центр', u'Восток', u'Север',  
 u'Европа', u'Азия', u'Бразилия', u'Атлантида'])  
 self.entry\_filial.place(x=250, y=125)  
  
 label\_manager = tk.Label(self, text='Менеджер')  
 label\_manager.place(x=50, y=150)  
 self.entry\_manager = ttk.Entry(self)  
 self.entry\_manager.place(x=250, 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\_id.get(),  
 self.entry\_name\_tovar.get(),  
 self.entry\_name\_shop.get(),  
 self.entry\_skidka.get(),  
 self.entry\_filial.get(),  
 self.entry\_manager.get()))  
 self.grab\_set()  
 self.focus\_set()  
  
  
class Update(Child): # Update class window  
 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=150, y=220)  
 btn\_edit.bind('<Button-1>', lambda event: self.view.update\_record(self.entry\_id.get(),  
 self.entry\_name\_tovar.get(),  
 self.entry\_name\_shop.get(),  
 self.entry\_skidka.get(),  
 self.entry\_filial.get(),  
 self.entry\_manager.get()))  
 self.btn\_ok.destroy()  
  
  
class Search(tk.Toplevel): # Search window  
 def \_\_init\_\_(self):  
 super().\_\_init\_\_()  
 self.entry\_search = None  
 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: # working with db  
 def \_\_init\_\_(self):  
 with sq.connect('torgFirm.db') as self.con:  
 self.cur = self.con.cursor()  
 # create new DB  
 self.cur.execute("""CREATE TABLE IF NOT EXISTS tovarDB (   
 id INTEGER PRIMARY KEY AUTOINCREMENT,  
 data DATE NOT NULL,  
 tovar TEXT NOT NULL,  
 cost FLOAT NOT NULL,  
 skidka FLOAT,  
 filial TEXT NOT NULL,  
 manager TEXT NOT NULL  
 )""")  
  
 def insert\_data(self, data, tovar, cost, skidka, filial, manager): # inserting data  
 self.cur.execute(  
 """INSERT INTO tovarDB (data, tovar, cost, skidka, filial, manager) VALUES (?,   
 ?, ?, ?, ?, ?)""",  
 (data, tovar.lower(), cost, skidka, filial.lower(), manager.lower()))  
 self.con.commit()  
  
  
if \_\_name\_\_ == "\_\_main\_\_": # Starting  
 root = tk.Tk()  
 db = DB()  
 app = Main(root)  
 app.pack()  
 root.title("Работа с базой данных")  
 root.geometry("1000x450+300+200")  
 root.resizable(False, False)  
 root.mainloop()

**Протокол работы:**



**Вывод:**в процессе выполнения практического занятия закрепил усвоенные знания, понятия, алгоритмы, основные принципы составления программ, работы с БД в **IDE PyCharm Community.**