Студента группы ПОКС-22 Логвинов М.А

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 ТОВАРНЫЙ ЗАПАС для автоматизированного учета товарных запасов на складе. БД должна содержать таблицу Товары со следующей структурой записи: Код товара, Торговая марка, Тип, Цена, Количество на складе, Минимальный запас.
   БД должна обеспечивать получение информации о товарных запасах по марке товара

Вариант 13.

Текст программы:

1. ЗАДАЧА

```
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):  # описание кнопок, их размещение, а также верхняя плашка
        toolbar = tk.Frame(bg='#7b917b', bd=4)
        toolbar.pack(side=tk.TOP, fill=tk.X)

        self.add_img = tk.PhotoImage(file="add.png")
        self.btn_open_dialog = tk.Button(toolbar, text='Добавить товар', command=self.open_dialog,
bg='#7b917b', bd=1,
                                         compound=tk.TOP, image=self.add_img)
        self.btn_open_dialog.pack(side=tk.LEFT, padx='60')

        self.update_img = tk.PhotoImage(file="remo.png")
        btn_edit_dialog = tk.Button(toolbar, text="Редактировать", command=self.open_update_dialog,
bg='#7b917b',
```

```
                        bd=1, compound=tk.TOP, image=self.update_img)
        btn_edit_dialog.pack(side=tk.LEFT, padx='10')

        self.delete_img = tk.PhotoImage(file="delete.png")
        btn_delete = tk.Button(toolbar, text="Удалить товар", command=self.delete_records,
bg='#7b917b',
                               bd=1, compound=tk.TOP, image=self.delete_img)
        btn_delete.pack(side=tk.LEFT, padx='60')

        self.search_img = tk.PhotoImage(file="poisk.png")
        btn_search = tk.Button(toolbar, text="Поиск марки товара", command=self.open_search_dialog,
bg='#7b917b',
                               bd=1, compound=tk.TOP, image=self.search_img)
        btn_search.pack(side=tk.LEFT, padx='10')

        self.refresh_img = tk.PhotoImage(file="update.png")
        btn_refresh = tk.Button(toolbar, text="Обновить экран", command=self.view_records,
bg='#7b917b',
                                bd=1, compound=tk.TOP, image=self.refresh_img)
        btn_refresh.pack(side=tk.LEFT, padx='60')

        self.tree = ttk.Treeview(self, columns=('product_id', 'trademark', 'type', 'cost', 'quantity_goods',
                                                'min_stock'), height=25, show='headings')

        self.tree.column('product_id', width=80, anchor=tk.CENTER)
        self.tree.column('trademark', width=160, anchor=tk.CENTER)
        self.tree.column('type', width=120, anchor=tk.CENTER)
        self.tree.column('cost', width=100, anchor=tk.CENTER)
        self.tree.column('quantity_goods', width=140, anchor=tk.CENTER)
        self.tree.column('min_stock', width=180, anchor=tk.CENTER)

        self.tree.heading('product_id', text='ID товара')
        self.tree.heading('trademark', text='Торговая марка')
        self.tree.heading('type', text='Тип')
        self.tree.heading('cost', text='Цена')
        self.tree.heading('quantity_goods', text='Количество на складах')
        self.tree.heading('min_stock', text='Минимальный запас')

        self.tree.pack()

    def records(self, product_id, trademark, type, cost, quantity_goods, min_stock):
        self.db.insert_data(product_id, trademark, type, cost, quantity_goods, min_stock)
        self.view_records()

    def update_record(self, product_id, trademark, type, cost, quantity_goods, min_stock):  #
редактирование
        self.db.cur.execute("""UPDATE users SET product_id=?, trademark=?, type=?, cost=?,
quantity_goods=?,
        min_stock=? WHERE product_id=?""", (product_id, trademark, type, cost, quantity_goods,
min_stock,
```

```
                                    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):  # удаление записей
        for selection_item in self.tree.selection():
            self.db.cur.execute("""DELETE FROM users WHERE product_id=?""", (self.tree.set(selection_item,
'#1'),))
        self.db.con.commit()
        self.view_records()

    def search_records(self, trademark):  # поиск записей
        trademark = (trademark,)
        self.db.cur.execute("""SELECT * FROM users WHERE trademark>?""", trademark)
        [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):  # создание окна "Добавить товар"

    """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 товар')
        self.geometry('400x220+400+300')
        self.resizable(False, False)

        label_product_id = tk.Label(self, text='Товар')
        label_product_id.place(x=50, y=25)
        self.entry_product_id = ttk.Entry(self)
```

```
        self.entry_product_id.place(x=180, y=25)

        label_trademark = tk.Label(self, text='Торговая марка')
        label_trademark.place(x=50, y=50)
        self.entry_trademark = ttk.Entry(self)
        self.entry_trademark.place(x=180, y=50)

        label_type = tk.Label(self, text='Тип')
        label_type.place(x=50, y=75)
        self.combobox = ttk.Combobox(self, values=[u'Оптовый', u'Розничный'])
        self.combobox.current(0)
        self.combobox.place(x=180, y=75)

        label_cost = tk.Label(self, text='Цена')
        label_cost.place(x=50, y=100)
        self.entry_cost = ttk.Entry(self)
        self.entry_cost.place(x=180, y=100)

        label_quantity_goods = tk.Label(self, text='Количество на складе')
        label_quantity_goods.place(x=50, y=125)
        self.entry_quantity_goods = ttk.Entry(self)
        self.entry_quantity_goods.place(x=180, y=125)

        label_min_stock = tk.Label(self, text='Минимальный запас')
        label_min_stock.place(x=50, y=150)
        self.entry_min_stock = ttk.Entry(self)
        self.entry_min_stock.place(x=180, y=150)

        btn_cancel = ttk.Button(self, text='Закрыть', command=self.destroy)
        btn_cancel.place(x=300, y=190)

        self.btn_ok = ttk.Button(self, text='Добавить')
        self.btn_ok.place(x=220, y=190)
        self.btn_ok.bind('<Button-1>', lambda event: self.view.records(self.entry_product_id.get(),
                                                self.entry_trademark.get(),
                                                self.combobox.get(),
                                                self.entry_cost.get(),
                                                self.entry_quantity_goods.get(),
                                                self.entry_min_stock.get()))

        self.grab_set()
        self.focus_set()


class Update(Child):  # создание окна "Редактировать запись"
    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=190)
        btn_edit.bind('<Button-1>', lambda event: self.view.update_record(self.entry_product_id.get(),
                                                          self.entry_trademark.get(),
                                                          self.combobox.get(),
                                                          self.entry_cost.get(),
                                                          self.entry_quantity_goods.get(),
                                                          self.entry_min_stock.get()))
        self.btn_ok.destroy()


class Search(tk.Toplevel):  # создание окна "Поиск"
    def __init__(self):
        super().__init__()
        self.init_search()
        self.view = app

    def init_search(self):
        self.title("Поиск торговой марки")
        self.geometry("400x100+400+300")
        self.resizable(False, False)

        label_search = tk.Label(self, text="Поиск торговой марки")
        label_search.place(x=30, y=20)

        self.entry_search = ttk.Entry(self)
        self.entry_search.place(x=180, 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('tovar.db') as self.con:  # создание таблицы
            self.cur = self.con.cursor()
            self.cur.execute("""CREATE TABLE IF NOT EXISTS users (
                product_id INTEGER PRIMARY KEY AUTOINCREMENT,
                trademark TEXT NOT NULL,
                type INTEGER NOT NULL DEFAULT 1,
                cost INTEGER,
                quantity_goods INTEGER,
```

```
            min_stock INTEGER
            )""")

    def insert_data(self, product_id, trademark, type, cost, quantity_goods, min_stock):
        self.cur.execute("""INSERT INTO users(product_id, trademark, type, cost, quantity_goods,
min_stock)
        VALUES (?, ?, ?, ?, ?, ?)""", (product_id, trademark, type, cost, quantity_goods, min_stock))
        self.con.commit()


if __name__ == "__main__":  # запуск и параметры для главного окна
    root = tk.Tk()
    db = DB()
    app = Main(root)
    app.pack()
    root['bg'] = '#7b917b'
    root.title("Товарный запас")
    root.geometry("900x600")
    root.resizable(False, False)
    root.mainloop()
```

Протокол работы программы:

1. **См. работу**

Вывод:

В процессе выполнения практического занятия выработал навыки разработки многооконного приложения для работы с помощью однотабличной БД в IDE PyCharm Community. Были использованы языковые библиотеки tkinter, sqlite3 а также методы работы с приложением. Выполнены разработка, отладка, тестирование, оптимизация программного кода. Готовые программные коды выложены на GitHub