**Практическая работа №4. Создание класса базы данных. Реализация основных операций с данными (CRUD) в консольном приложении.**

Выполнил студент группы ИСП21.1А Яковлев Алексей

Предметная область: Авторемонтные мастерские

Задание.

1. Создать проект PyCharm.
2. Создать в нем 5 модулей .py
3. В первом модуле описать класс, соответствующий предметной области по варианту. Класс должен содержать:

- конструктор с подключением к базе данных, запросы на создание таблиц (CREATETABLE);

-методы отключения базы данных, просмотра данных из таблиц, добавления данных, удаления данных, редактирования данных, поиска данных по критерию.

1. Остальные модули содержат методы и программный код, демонстрирующие работу с данными из базы данных.
2. Оформить отчет о выполненной работе:

- заголовок,

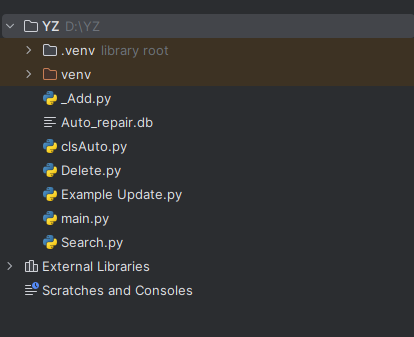
- кто выполнил,

- описание структуры проекта,

- программный код модулей,

- результаты выполнения модулей (скрины).

1. Создал проект в пайчарме и создал в нём 5 модулей



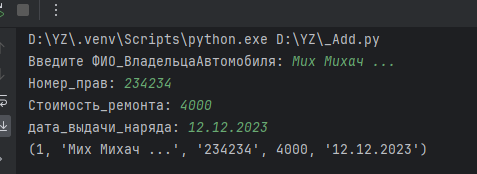
2. Первый модуль clsAuto

import sqlite3  
# создание класса БД  
class Auto\_repair\_shop:  
 # конструктор класса  
 def \_\_init\_\_(self):  
 # Подключение к БД  
 self.con = sqlite3.connect("Auto\_repair.db")  
 # Создание курсора  
 self.cur = self.con.cursor()  
 # Создание таблицы БД  
 self.cur.execute(  
 "CREATE TABLE IF NOT EXISTS person "  
 "(id INTEGER PRIMARY KEY,"  
 "ФИО\_ВладельцаАвтомобиля TEXT,"  
 "Номер\_прав TEXT,"  
 "Cтоимость\_ремонта INTEGER,"  
 "дата\_выдачи\_наряда TEXT)"  
 )  
 # сохранить изменения  
 self.con.commit()  
 def \_\_del\_\_(self):  
 # отключение от БД  
 self.con.close()  
 def view(self):  
 # просмотр всех записей в таблице БД  
 self.cur.execute("SELECT \* FROM person")  
 # список всех записей из таблицы  
 rows = self.cur.fetchall()  
 return rows  
 def insert(self,ФИО\_ВладельцаАвтомобиля,Номер\_прав ,Стоимость\_ремонта,дата\_выдачи\_наряда):  
 # добавить запись  
 self.cur.execute("INSERT INTO person "  
 "VALUES (NULL, ?, ?, ?, ?)",(ФИО\_ВладельцаАвтомобиля, Номер\_прав,Стоимость\_ремонта ,дата\_выдачи\_наряда))  
 self.con.commit()  
 def update(self, id, ФИО\_ВладельцаАвтомобиля,Номер\_прав,Стоимость\_ремонта,дата\_выдачи\_наряда):  
 # редактирование записи  
 self.cur.execute("UPDATE person SET "  
 "ФИО\_ВладельцаАвтомобиля=?, Номер\_прав=? "  
 "WHERE ID = ?",  
(ФИО\_ВладельцаАвтомобиля,Номер\_прав,Стоимость\_ремонта,дата\_выдачи\_наряда,  
 id,))  
 self.con.commit()  
 def delete(self, id):  
 # удаление записи  
 self.cur.execute("DELETE FROM person "  
 "WHERE ID=?", (id,))  
 self.con.commit()  
 def search(self, lastname):  
 self.cur.execute("SELECT ФИО\_ВладельцаАвтомобиля,Номер\_прав,Стоимость\_ремонта ,дата\_выдачи\_наряда FROM person "  
 "WHERE ФИО\_ВладельцаАвтомобиля=?", (lastname,))  
 rows = self.cur.fetchall()  
 return rows

2. Второй модуль \_Add.py

from clsAuto import Auto\_repair\_shop  
# создать объект базы данных  
database\_Auto = Auto\_repair\_shop()  
# логика  
# добавление записи  
def add\_command(ФИО\_ВладельцаАвтомобиля,Номер\_прав ,Стоимость\_ремонта ,дата\_выдачи\_наряда):  
 database\_Auto.insert(ФИО\_ВладельцаАвтомобиля,Номер\_прав,Стоимость\_ремонта,дата\_выдачи\_наряда)  
# просмотр всех записей  
def view\_command():  
 for row in database\_Auto.view():  
 print(row)  
# основная программа в консоли  
# добавление записи  
for i in range(1):  
 add\_command(input("Введите ФИО\_ВладельцаАвтомобиля: "),  
 input("Номер\_прав: "),  
 input("Стоимость\_ремонта: "),  
 input("дата\_выдачи\_наряда: ")  
)  
# просмотр всех записей  
view\_command()

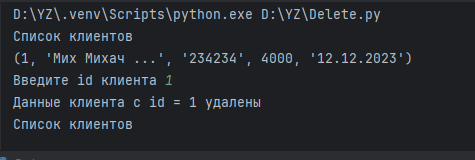
Результат выполнения:



3. Модуль Delete.py

from clsAuto import Auto\_repair\_shop  
# создать объект базы данных  
database\_Auto = Auto\_repair\_shop()  
# удаление по id студента  
def delete\_command(id):  
 database\_Auto.delete(id)  
 print(f"Данные клиента с id = {id} удалены")  
# просмотр всех записей  
def view\_command():  
 for row in database\_Auto.view():  
 print(row)  
# основная программа  
print("Список клиентов")  
view\_command()  
id\_delete = int(input("Введите id клиента "))  
delete\_command(id\_delete)  
print("Список клиентов")  
view\_command()

Результат выполнения:



4. Модуль Update.py

from clsAuto import Auto\_repair\_shop  
# создать объект базы данных  
database\_Auto = Auto\_repair\_shop()  
# удаление по id студента  
def update\_command(ФИО\_ВладельцаАвтомобиля, Номер\_прав ,Стоимость\_ремонта ,дата\_выдачи\_наряда):  
 database\_Auto .update(id, ФИО\_ВладельцаАвтомобиля, Номер\_прав,Стоимость\_ремонта ,дата\_выдачи\_наряда)  
 print(f"Данные клиента с id = {id} обновлены")  
# просмотр всех записей  
def view\_command():  
 for row in database\_Auto .view():  
 print(row)  
# основная программа  
print("Список клиетов")  
view\_command()  
id\_update = int(input("Введите id клиентов "))  
print("Укажите новые данные: ")  
ФИО\_ВладельцаАвтомобиля = input("ФИО")  
Номер\_прав = input("Номер\_прав")  
Стоимость\_ремонта = input("Стоимость\_ремонта ")  
дата\_выдачи\_наряда = input("дата\_выдачи\_наряда")  
  
update\_command(id\_update, ФИО\_ВладельцаАвтомобиля,Номер\_прав,Стоимость\_ремонта ,дата\_выдачи\_наряда)  
print("Список клиентов")  
view\_command()

5. Модуль search.py

from clsAuto import Auto\_repair\_shop  
# создать объект базы данных  
database\_Auto = Auto\_repair\_shop()  
# просмотр всех записей  
def view\_command():  
 for row in database\_Auto.view():  
 print(row)  
# поиск по фамилии  
def search\_command(ФИО\_ВладельцаАвтомобиля):  
 if len(database\_Auto.search(ФИО\_ВладельцаАвтомобиля)) > 0:  
 for row in database\_Auto.search(ФИО\_ВладельцаАвтомобиля):  
 print(row)  
 else:  
 print("Такого клиента нет")  
# основная программа в консоли  
# поиск по фамилии  
search\_command(input("ФИО клиента? "))