**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ**

**ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ**

**УНИВЕРСИТЕТ ИМЕНИ П.О.СУХОГО**

Факультет автоматизированных и информационных систем

Кафедра «Информационные технологии»

Дисциплина «скриптованные языки программирования»

ОТЧЕТ

Лабораторная работа № 4

База данных в текстовом файле

Выполнил студент

Группы ИТД-21

Чайдаков И.М.

Принял преподаватель

Малиновский И. Л.

Гомель 2024

База данных в текстовом файле

**Цель работы:** Научиться использовать базы данных из внешних файлов. Научиться считывать из них данные и записывать в них новые. Создавать программы используя предоставляемые ими данные.

**Задание 1:** Создать программу на в которой будет организован список словарей с ключами соответствующие пункту "атрибуты". Программа должна:

1. выбрать файл для работы

2. Инициализировать базу данных

3. Добавлять записи

4. Выводить на экран все записи в виде таблицы

5. Удалять записи по номеру

6. Добавлять запись в базу данных

7. Поиск по одному пол.

8. Поиск по двум полям.

9. Осуществлять поиск в соответствии с запросами указанными в пункте "Основные функции".

Результаты поиска выводить на экран в виде таблицы.

Атрибуты: ФИО, Название места работы, Должность, Даты мед. осмотров, Название специалиста проводившего осмотр, результаты мед. осмотра.

Основные функции: Удалить все записи по заданному месту работы, заменить должность для заданного работника, вывести все сведения о работниках прошедших мед. осмотр в определенную дату.

**Ход выполнения задания:**

**1.** Первым делом подготавливаем файл с базой данным в расширении *json*. В него записываем базовые данные работников которыми позже будет управлять (Приложение А).

**2.** Описываем код нашей программы (Приложение B) и запускаем программу встречая в консоли пользовательское меню (рисунок 1).

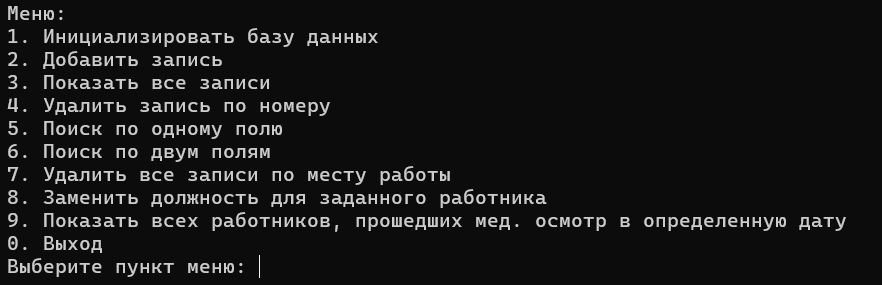


Рисунок 1 — Меню программы.

**3.** Выберем третий пункт и осмотрим наши данные из текстового файла (рисунок 2).

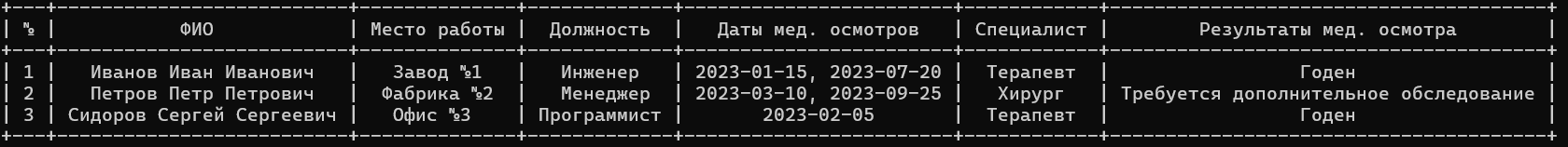


Рисунок 3 — Таблица данных.

**4.** Выведем сортировку по полю должности с маркировкой инженера с помощью пятого пункта (рисунок 4).

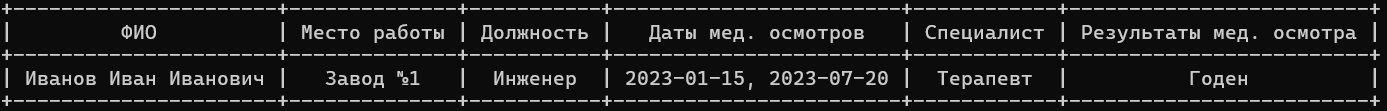


Рисунок 4 — Поиск по одному полю.

**5.** Выведем сортировку по полю специалиста с маркировкой хирурга и ФИО с маркировкой Петрова с шестым пунктом (рисунок 5).

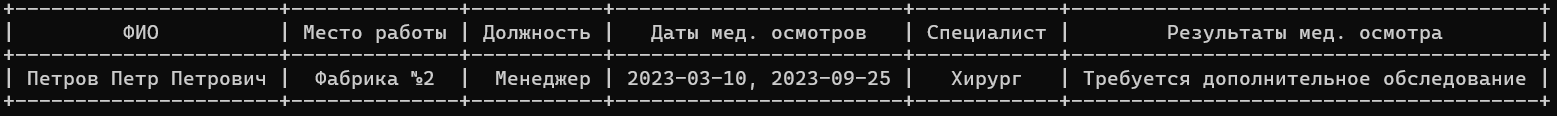


Рисунок 5 — Поиск по двум полям.

**6.** Создадим еще один объект и запишем его в файл со вторым пунктом (рисунок 6).

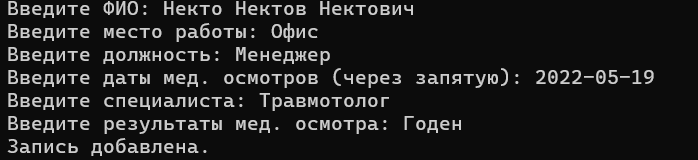


Рисунок 6 — Создание нового работника.

**7.** Далее, с четвертым пунктом, удалим работника под номером один. Теперь в списке три объекта один из которым мы сами создали. Попробуем сменить должность созданного нами сотрудника с восьмым пунктом (Рисунок 7).

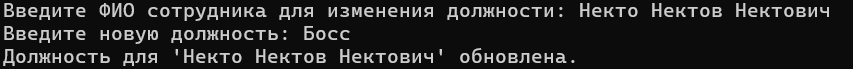


Рисунок 7 — Смена должности.

**Вывод:** Базы данных хранимые в текстовых файлах позволяют нам работать с заранее заданным набором параметров и переносить данные в последующие итерации программы. Это крайне полезный инструмент широко используемый в современной разработке приложений.

**ПРИЛОЖЕНИЕ А**

***Json*-файл**

[

    {

        "ФИО": "Петров Петр Петрович",

        "Место работы": "Фабрика №2",

        "Должность": "Менеджер",

        "Даты мед. осмотров": [

            "2023-03-10",

            "2023-09-25"

        ],

        "Специалист": "Хирург",

        "Результаты мед. осмотра": "Требуется дополнительное обследование"

    },

    {

        "ФИО": "Сидоров Сергей Сергеевич",

        "Место работы": "Офис №3",

        "Должность": "Программист",

        "Даты мед. осмотров": [

            "2023-02-05"

        ],

        "Специалист": "Терапевт",

        "Результаты мед. осмотра": "Годен"

    },

    {

        "ФИО": "Некто Нектов Нектович",

        "Место работы": "Офис",

        "Должность": "Босс",

        "Даты мед. осмотров": [

            "2022-05-19"

        ],

        "Специалист": "Травмотолог",

        "Результаты мед. осмотра": "Годен"

    }

]

**ПРИЛОЖЕНИЕ B**

**Код программы**

import json

from prettytable import PrettyTable  # type: ignore

FILE\_NAME = "employees.json"

def load\_data():

    try:

        with open(FILE\_NAME, 'r', encoding='utf-8') as file:

            return json.load(file)

    except (FileNotFoundError, json.JSONDecodeError):

        return []

def save\_data(data):

    with open(FILE\_NAME, 'w', encoding='utf-8') as file:

        json.dump(data, file, ensure\_ascii=False, indent=4)

def add\_record(data):

    fio = input("Введите ФИО: ")

    workplace = input("Введите место работы: ")

    position = input("Введите должность: ")

    med\_exam\_dates = input("Введите даты мед. осмотров (через запятую): ").split(", ")

    specialist = input("Введите специалиста: ")

    exam\_results = input("Введите результаты мед. осмотра: ")

    record = {

        "ФИО": fio,

        "Место работы": workplace,

        "Должность": position,

        "Даты мед. осмотров": med\_exam\_dates,

        "Специалист": specialist,

        "Результаты мед. осмотра": exam\_results

    }

    data.append(record)

    save\_data(data)

    print("Запись добавлена.")

def display\_search\_results(results):

    if results:

        table = PrettyTable()

        table.field\_names = ["ФИО", "Место работы", "Должность", "Даты мед. осмотров", "Специалист", "Результаты мед. осмотра"]

        for record in results:

            table.add\_row([

                record["ФИО"],

                record["Место работы"],

                record["Должность"],

                ", ".join(record["Даты мед. осмотров"]) if isinstance(record["Даты мед. осмотров"], list) else record["Даты мед. осмотров"],

                record["Специалист"],

                record["Результаты мед. осмотра"]

            ])

        print(table)

    else:

        print("По запросу записи не найдены.")

def initialize\_database():

    return []

data = load\_data()

while True:

    print("\nМеню:")

    print("1. Инициализировать базу данных")

    print("2. Добавить запись")

    print("3. Показать все записи")

    print("4. Удалить запись по номеру")

    print("5. Поиск по одному полю")

    print("6. Поиск по двум полям")

    print("7. Удалить все записи по месту работы")

    print("8. Заменить должность для заданного работника")

    print("9. Показать всех работников, прошедших мед. осмотр в определенную дату")

    print("0. Выход")

    choice = input("Выберите пункт меню: ")

    match choice:

        case "1":

            data = initialize\_database()

            save\_data(data)

            print("База данных инициализирована (очищена).")

        case "2":

            add\_record(data)

        case "3":

            if data:

                table = PrettyTable()

                table.field\_names = ["№", "ФИО", "Место работы", "Должность", "Даты мед. осмотров", "Специалист", "Результаты мед. осмотра"]

                for idx, record in enumerate(data, start=1):

                    table.add\_row([

                        idx,

                        record["ФИО"],

                        record["Место работы"],

                        record["Должность"],

                        ", ".join(record["Даты мед. осмотров"]) if isinstance(record["Даты мед. осмотров"], list) else record["Даты мед. осмотров"],

                        record["Специалист"],

                        record["Результаты мед. осмотра"]

                    ])

                print(table)

            else:

                print("База данных пуста.")

        case "4":

            try:

                index = int(input("Введите номер записи для удаления: ")) - 1

                if 0 <= index < len(data):

                    removed = data.pop(index)

                    save\_data(data)

                    print(f"Запись под номером {index + 1} удалена.")

                else:

                    print("Неверный номер записи.")

            except ValueError:

                print("Пожалуйста, введите корректный номер записи.")

        case "5":

            field = input("Введите поле для поиска (ФИО, Место работы, Должность, Даты мед. осмотров, Специалист, Результаты мед. осмотра): ")

            value = input("Введите значение для поиска: ")

            if field == "Даты мед. осмотров":

                results = [record for record in data if value in record.get(field, [])]

            else:

                results = [record for record in data if record.get(field) == value]

            display\_search\_results(results)

        case "6":

            field1 = input("Введите первое поле для поиска: ")

            value1 = input("Введите значение для первого поля: ")

            field2 = input("Введите второе поле для поиска: ")

            value2 = input("Введите значение для второго поля: ")

            results = [record for record in data if record.get(field1) == value1 and record.get(field2) == value2]

            display\_search\_results(results)

        case "7":

            workplace = input("Введите место работы для удаления записей: ")

            new\_data = [record for record in data if record.get("Место работы") != workplace]

            if len(new\_data) < len(data):

                data = new\_data

                save\_data(data)

                print(f"Все записи для места работы '{workplace}' удалены.")

            else:

                print(f"Записей с местом работы '{workplace}' не найдено.")

        case "8":

            fio = input("Введите ФИО сотрудника для изменения должности: ")

            new\_position = input("Введите новую должность: ")

            updated = False

            for record in data:

                if record.get("ФИО") == fio:

                    record["Должность"] = new\_position

                    updated = True

            if updated:

                save\_data(data)

                print(f"Должность для '{fio}' обновлена.")

            else:

                print(f"Сотрудник с ФИО '{fio}' не найден.")

        case "9":

            exam\_date = input("Введите дату мед. осмотра для поиска (в формате ГГГГ-ММ-ДД): ")

            results = [record for record in data if exam\_date in record.get("Даты мед. осмотров", [])]

            display\_search\_results(results)

        case "0":

            break

        case \_:

            print("Неверный ввод. Попробуйте снова.")