Lista de Exercícios

1. Sistema de Gerenciamento de Alunos

Crie um programa que gerencie informações sobre alunos de uma escola. As informações devem ser armazenadas em um arquivo e cada registro deve conter:

- **Campos da struct:**
    - int matricula: Número de matrícula do aluno.
    - char nome[50]: Nome do aluno.
    - char curso[30]: Nome do curso que o aluno está matriculado.
    - int idade: Idade do aluno.
- **Funcionalidades do programa:**

1. Incluir um novo aluno.
2. Alterar informações de um aluno (nome, curso ou idade).
3. Excluir um aluno pelo número de matrícula.
4. Consultar informações de um aluno.
5. Listar todos os alunos cadastrados.

---

2. Cadastro de Funcionários

Implemente um programa que gerencie os dados de funcionários de uma empresa. O programa deve usar um arquivo para persistir os dados e cada registro deve conter:

- **Campos da struct:**
    - int id: Identificador único do funcionário.
    - char nome[50]: Nome completo do funcionário.
    - char cargo[30]: Cargo que o funcionário ocupa.
    - int salario: Salário do funcionário (valor inteiro representando em reais).
- **Funcionalidades do programa:**

1. Cadastrar um novo funcionário.
2. Alterar dados de um funcionário (cargo ou salário).
3. Excluir um funcionário pelo ID.
4. Consultar informações de um funcionário pelo ID.
5. Listar todos os funcionários cadastrados.

---

3. Gerenciamento de Livros

Desenvolva um programa para gerenciar um catálogo de livros em uma biblioteca. As informações devem ser salvas em um arquivo e cada registro deve conter:

- **Campos da struct:**
    - int codigo: Código único do livro.
    - char titulo[50]: Título do livro.
    - char autor[50]: Nome do autor do livro.

- int ano: Ano de publicação.

- **Funcionalidades do programa:**

1. Adicionar um novo livro ao catálogo.
2. Alterar os dados de um livro (título, autor ou ano de publicação).
3. Excluir um livro pelo código.
4. Consultar informações de um livro pelo código.
5. Listar todos os livros cadastrados.

---

4. Registro de Veículos

Crie um programa que registre informações sobre veículos em uma concessionária. O programa deve usar um arquivo para armazenar os dados, e cada registro deve conter:

- **Campos da struct:**
  - int id: Identificador único do veículo.
  - char modelo[50]: Modelo do veículo (ex.: "Fiat Uno").
  - char placa[10]: Placa do veículo.
  - int ano: Ano de fabricação do veículo.
- **Funcionalidades do programa:**

1. Adicionar um novo veículo ao sistema.
2. Alterar os dados de um veículo (modelo, placa ou ano).
3. Excluir um veículo pelo ID.
4. Consultar as informações de um veículo pela placa.
5. Listar todos os veículos cadastrados.

---

5. Controle de Filmes

Desenvolva um programa para gerenciar um acervo de filmes de uma locadora. Os dados devem ser armazenados em um arquivo e cada registro deve conter:

- **Campos da struct:**
  - int id: Identificador único do filme.
  - char titulo[50]: Título do filme.
  - char genero[20]: Gênero do filme (ex.: "Ação", "Comédia").
  - int ano: Ano de lançamento.
- **Funcionalidades do programa:**

1. Adicionar um novo filme ao acervo.
2. Alterar os dados de um filme (título, gênero ou ano).
3. Excluir um filme pelo ID.
4. Consultar as informações de um filme pelo ID.
5. Listar todos os filmes cadastrados.

---

6. Cadastro de Cidades

Crie um programa que gerencie o cadastro de cidades de um país. O programa deve armazenar os dados em um arquivo de texto, e cada registro deve conter as seguintes informações:

- Campos da struct:
    - int codigo: Código único da cidade.
    - char nome[50]: Nome da cidade.
    - char estado[30]: Nome do estado onde a cidade está localizada.
    - int populacao: População da cidade.
- Funcionalidades do programa:

1. Incluir cidade: Adicionar uma nova cidade ao cadastro, fornecendo o código, nome, estado e população.
2. Alterar dados: Modificar os dados de uma cidade existente (nome, estado ou população).
3. Excluir cidade: Remover uma cidade pelo código.
4. Consultar cidade: Exibir os dados de uma cidade com base no código.
5. Listar cidades: Mostrar todas as cidades cadastradas no arquivo.

Dicas para Implementação

- Use fgets e delimitadores como vírgulas para gravar e ler os dados de forma eficiente (como no exemplo anterior).
- Utilize arquivos temporários (temp.txt) para manipular dados durante as operações de alteração ou exclusão.
- Valide entradas como identificadores (ID, matrícula, código) para evitar inconsistências no arquivo.
- Teste as funcionalidades para verificar se as operações de gravação e leitura no arquivo estão funcionando corretamente.