



| | | | |
|---|--|--------------------|---|
|  | MINISTÉRIO DA EDUCAÇÃO UNIVERSIDADE FEDERAL DO PIAUÍ – UFPI CAMPUS SENADOR HELVÍDIO NUNES DE BARROS – PICOS | |  |
| Curso: Sistemas de Informação | | Período: 2º | Ano/Semestre: 2025.1 |
| Disciplina: Algoritmos e Programação II | | | Professor: José Denes Lima Araújo |

7º ATIVIDADE – STRUCTS

1. Crie uma struct **Aluno** com os campos:

- Nome
- Curso
- Média final

Especificações:

- Permitir o cadastro de **n** (valor fornecido pelo usuário) alunos.
- Exibir os dados de todos os alunos cadastrados.
- Exibir somente os alunos aprovados, ou seja, com média final maior ou igual a 7.0.
- Não é permitido fazer as funcionalidades dentro do main.

O programa deve ter as seguintes funções obrigatórias:

- void cadastrarAlunos(Aluno alunos[], int n)
- void exibirAlunos(Aluno alunos[], int n)
- void exibirAprovados(Aluno alunos[], int n)

2. Crie uma struct **Produto** com os campos:

- Nome: nome do produto.
- Id: identificador **único** do produto.
- Preço: preço do produto.
- Quantidade_estoque: quantidade disponível em estoque.

Especificações:

- Permitir o cadastro de **n** (valor fornecido pelo usuário) produtos.

- O programa não deve permitir o cadastro de dois produtos com mesmo id.
- Permitir que o usuário realize a busca de um produto por **id** e caso ele exista exiba seus dados, caso contrário, imprima que o produto não está cadastrado.
- Não é permitido fazer as funcionalidades dentro do main.

O programa deve ter as seguintes funções obrigatórias:

- void CadastrarProduto(Produto produtos[], int n)
- void BuscarProduto(Produto produtos[], int n)

3. Crie uma struct chamada **Livro** com os seguintes campos:

- **titulo** (string de até 100 caracteres)
- **codigo** (inteiro, identificador único)
- **disponivel** (inteiro, 1 para disponível, 0 para emprestado)

Especificações:

- O programa deve permitir:
 - Cadastrar **n** (valor fornecido pelo usuário) livros.
 - Buscar um livro pelo código e exibir suas informações.
 - **Emprestar um livro:** mudar o status de "disponível" para "emprestado" se o livro existir e estiver disponível.
 - Não deve permitir cadastrar dois livros com o mesmo código.
 - Não é permitido fazer as funcionalidades dentro do main.

O programa deve ter as seguintes funções obrigatórias:

- void CadastrarLivros(Livro livros[], int n)
- void ConsultarLivro(Livro livros[], int n)
- void EmprestarLivro(Livro livros[], int n)

4. Crie uma struct chamada **Veiculo** com os seguintes campos:

- **modelo** (string de até 50 caracteres)
- **placa** (string de até 10 caracteres – identificador único)
- **ano** (inteiro)

Especificações:

- Permitir o cadastro de **n** (valor fornecido pelo usuário) veículos.
- Não permitir o cadastro de dois veículos com a mesma placa.
- Permitir a **exibição de todos os veículos cadastrados**.
- Permitir a remoção de um veículo pela placa.
- É permitido o uso de **variável global** para armazenar o total de veículos.
- Nenhuma lógica deve estar no main.

O programa deve ter as seguintes funções obrigatórias:

- void CadastrarVeiculo(Veiculo veiculos[])
- void MostrarVeiculos(Veiculo veiculos[])
- void RemoverVeiculo(Veiculo veiculos[])

5. Faça um programa que implemente uma agenda telefônica com um menu interativo.

Structs:

- **Struct Contato:** (string)nome, (string)e-mail, (Telefone)telefone, (Endereço)endereço, (int)id.
- **Struct Endereço:** (string)rua, (int)número, (string)bairro, (string)cidade, (string)estado, (string)país.
- **Struct Telefone:** (int)DDD, (int)número.

Funções obrigatórias:

- int inserirContato(Contato ctts[], int quantidade);
- int deletarContato(Contato ctts[], int quantidade, int id);
- void listarContatos(Contato ctts[], int quantidade);

Especificações:

- **Deve ser implementado um menu que permita ao usuário:**
 - Inserir um novo contato
 - Deletar um contato existente (por ID)
 - Listar todos os contatos
 - Sair do programa

- O menu deve ser executado em loop até o usuário escolher a opção de sair.
- Usar vetores fixos de Contato (por exemplo, Contato contatos[100]).
- É importante validar as entradas do usuário (por exemplo, evitar DDDs negativos, verificar existência do ID antes de deletar).

6. Crie um programa em C que armazene os dados de até 50 funcionários de uma empresa.

Struct Funcionário:

- (string)nome
- (int)id
- Horário de início do expediente (número inteiro de 1 a 12)
- Horário de fim do expediente (número inteiro de 1 a 12)

Funções obrigatórias:

- void cadastrarFuncionarios(struct Funcionario funcionarios[], int n);
- int verificarSobreposicao(struct Funcionario funcionarios[], int n, int id1, int id2);

Especificações:

- Cadastrar os funcionários.
- Verificar se os turnos de trabalho de **dois funcionários, informados pelos seus IDs**, se sobrepõem (ou seja, se existe algum intervalo de tempo em que ambos estejam trabalhando ao mesmo tempo).

7. Você foi convidado a criar um sistema para registrar votos dos alunos em uma eleição para representante de turma. Cada aluno pode votar apenas uma vez, e o sistema deve garantir isso.

Struct Aluno:

- (int)matricula
- (string)nome
- (int)votou (1 para sim, 0 para não)

- (int)voto

Funções obrigatórias:

- void cadastrarAlunos(struct Aluno alunos[], int n);
- void registrarVoto(struct Aluno alunos[], int n);
- void exibirResultado(struct Aluno alunos[], int n);

Especificações:

- O programa deve permitir o cadastro de n alunos.
- Cada aluno pode votar apenas uma vez. Se tentar votar de novo, o sistema deve informar que ele já votou.
- Os candidatos são identificados pelos números: 1, 2 e 3.
- A função exibirResultado deve mostrar quantos votos cada candidato recebeu.
- **O main deve ter um menu com as opções:**
 - Cadastrar alunos
 - Registrar voto
 - Exibir resultado
 - Sair