Lista de Estrutura

Prazos

- Primeiro prazo: 10/05/2024
- Segundo prazo: 17/05/2024

Enunciado 1: Dados Pessoais

Descrição:

Crie um programa que defina uma estrutura para armazenar nome, idade e altura de uma pessoa. Leia os dados de uma pessoa, verifique se a pessoa é maior de idade (18 anos ou mais) e imprima os dados junto com a informação se ela é maior de idade.

Input de Exemplo:

```
Nome: João Silva
Idade: 25
Altura: 1.75
```

Output Esperado:

```
Nome: João Silva
Idade: 25
Altura: 1.75
Maior de idade: Sim
```

Enunciado 2: Livro

Descrição:

Crie uma estrutura para armazenar o título de um livro, o autor e o ano de publicação. Leia os dados de um livro, calcule quantos anos se passaram desde a publicação e imprima os dados junto com a idade do livro.

Input de Exemplo:

```
Título: O Alquimista
Autor: Paulo Coelho
Ano de Publicação: 1988
```

Output Esperado:

```
Título: O Alquimista
Autor: Paulo Coelho
Ano de Publicação: 1988
Idade do Livro: 36 anos
```

Enunciado 3: Produto

Descrição:

Crie uma estrutura para armazenar o nome e o preço de um produto. Leia os dados de dois produtos, calcule e imprima o preço total.

Input de Exemplo:

```
Produto 1: Nome: Laptop, Preço: 3500.50
Produto 2: Nome: Mouse, Preço: 50.00
```

Output Esperado:

```
Produto 1: Nome: Laptop, Preço: 3500.50
Produto 2: Nome: Mouse, Preço: 50.00
Preço Total: 3550.50
```

Enunciado 4: Carro

Descrição:

Crie uma estrutura para armazenar o modelo, o ano de fabricação e a quilometragem de um carro. Leia os dados de um carro, calcule a idade do carro e imprima os dados junto com a idade do carro.

Input de Exemplo:

```
Modelo: Honda Civic
Ano de Fabricação: 2020
Quilometragem: 15000
```

Output Esperado:

```
Modelo: Honda Civic
Ano de Fabricação: 2020
Quilometragem: 15000
Idade do Carro: 4 anos
```

Enunciado 5: Data de Nascimento

Descrição:

Crie uma estrutura para armazenar uma data (dia, mês, ano). Crie outra estrutura para armazenar os dados de uma pessoa (nome, data de nascimento). Leia os dados, calcule a idade da pessoa com base na data de nascimento e imprima os dados.

Input de Exemplo:

```
Nome: Maria Santos
Data de Nascimento: 15/06/1993
```

Output Esperado:

```
Nome: Maria Santos
Data de Nascimento: 15/06/1993
Idade: 30
```

Enunciado 6: Lista de Produtos

Descrição:

Crie uma estrutura para armazenar o nome e o preço de um produto. Crie um array de 5 produtos, leia os dados, calcule o preço total e imprima os produtos e o total.

Input de Exemplo:

```
Produto 1: Nome: Teclado, Preço: 150.00
Produto 2: Nome: Mouse, Preço: 50.00
Produto 3: Nome: Monitor, Preço: 700.00
```

```
Produto 4: Nome: Cadeira, Preço: 300.00
Produto 5: Nome: Mesa, Preço: 250.00
```

Output Esperado:

```
Produto 1: Nome: Teclado, Preço: 150.00
Produto 2: Nome: Mouse, Preço: 50.00
Produto 3: Nome: Monitor, Preço: 700.00
Produto 4: Nome: Cadeira, Preço: 300.00
Produto 5: Nome: Mesa, Preço: 250.00
Preço Total: 1450.00
```

Enunciado 7: Lista de Alunos

Descrição:

Crie uma estrutura para armazenar o nome e as 3 notas de um aluno. Crie um array de N alunos, leia os dados, calcule a média das notas e imprima os alunos e a média.

Dicas:

- Abuse de funções.
- Quando precisar passar a referência de uma estrutura para uma função, como no exemplo `calcularMedia(Aluno *aluno)`, dentro da função não devemos acessar os atributos utilizando `.`. A maneira correta é utilizando `->`, por exemplo, `aluno->notas[i];` está correto, enquanto `aluno.notas[i];` está incorreto.
- Quando precisar passar o valor de uma estrutura para uma função, como no exemplo `imprimirAluno(Aluno aluno)`, dentro da função podemos acessar os atributos utilizando `.`, por exemplo `printf("Nome: %s\n", aluno.nome);`

Input de Exemplo:

```
Aluno 1:  Nome: Pedro,  Nota 1: 8.5   Nota 2: 6.5   Nota 3: 2.5
Aluno 2:  Nome: Ana,    Nota 1: 9.0   Nota 2: 9.0   Nota 3: 9.0
Aluno 3:  Nome: Lucas,  Nota 1: 4.0   Nota 2: 8.5   Nota 3: 9.0
```

Output Esperado:

```
Aluno 1:  Nome: Pedro,  Media: 5.83
Aluno 2:  Nome: Ana,    Media: 9.00
Aluno 3:  Nome: Lucas,  Media: 7.17
```

Enunciado 8: Sistema de Reservas

Descrição:

Desenvolva um programa para gerenciar reservas de hotel, criando uma estrutura para armazenar os dados de cada reserva. A estrutura deve incluir o nome do hóspede, o número do quarto, a data de check-in e a data de check-out. Em seguida, crie um array contendo N reservas, leia os dados de entrada, calcule a duração da estadia em dias para cada reserva e exiba os resultados.

Observação: Para simplificar, não consideraremos anos bissextos e assumiremos que todos os meses têm 31 dias.

Dicas: Quando precisar passar a referência de uma estrutura para uma função, como no exemplo `lerReserva(&reservas[i]);`, dentro da função não devemos acessar os atributos utilizando `.`. A maneira correta é utilizando `->`, por exemplo, `scanf("%d", &reserva->numeroQuarto);` está correto, enquanto `scanf("%d", &reserva.numeroQuarto);` está incorreto.

Input de Exemplo:

```
Reserva 1: Nome: Alice, Quarto: 101, Check-in: 01/05/2024, Check-out:
11/05/2024
Reserva 2: Nome: Bruno, Quarto: 102, Check-in: 10/05/2024, Check-out:
20/06/2024
Reserva 3: Nome: Carla, Quarto: 103, Check-in: 20/05/2024, Check-out:
27/07/2024
```

Output Esperado:

```
Reserva 1: Nome: Alice, Quarto: 101, Check-in: 01/05/2024, Check-out:
05/05/2024, Duração: 10 dias
Reserva 2: Nome: Bruno, Quarto: 102, Check-in: 10/05/2024, Check-out:
15/05/2024, Duração: 41 dias
Reserva 3: Nome: Carla, Quarto: 103, Check-in: 20/05/2024, Check-out:
25/05/2024, Duração: 69 dias
```

Enunciado 9: Exercício: Simulador de Agenda Telefônica

Descrição:

Crie um programa em C para simular uma agenda de telefones. Para cada pessoa, devem ser armazenados os seguintes dados:

- Nome
- E-mail
- Endereço (contendo campos para Rua, número, complemento, bairro, CEP, cidade, estado, país)
- Telefone (contendo campos para DDD e número)
- Data de aniversário (contendo campos para dia, mês, ano)
- Observações (uma linha de texto para observações especiais)

Dica: Tente separar seu programa em procedimentos e funções, utilize variável global.

Objetivos do Sistema:

1. Definir a estrutura de dados para armazenar as informações da agenda.
2. Declarar a variável agenda (vetor) com capacidade para até 100 nomes.
3. Implementar as seguintes funcionalidades:

Funcionalidades:

- a) Definir a estrutura:
    - Crie uma estrutura Endereco para armazenar os detalhes do endereço.
    - Crie uma estrutura Telefone para armazenar o DDD e o número do telefone.
    - Crie uma estrutura Data para armazenar a data de aniversário.
    - Crie uma estrutura Pessoa que inclua todas as informações mencionadas.
- b) Declarar a variável agenda:
    - Declare um vetor de 100 elementos do tipo Pessoa.
- c) Buscar por primeiro nome:
    - Implemente uma função que busca uma pessoa pelo primeiro nome e imprime os dados dessa pessoa (se houver mais de uma pessoa com o mesmo nome, imprima os dados de todas).
- d) Buscar por mês de aniversário:
    - Implemente uma função que busca todas as pessoas que fazem aniversário em um determinado mês e imprime os dados dessas pessoas.
- e) Buscar por dia e mês de aniversário:
    - Implemente uma função que busca todas as pessoas que fazem aniversário em um determinado dia e mês e imprime os dados dessas pessoas.
- f) Inserir pessoa:
    - Implemente uma função para inserir uma nova pessoa na agenda, mantendo a ordem alfabética dos nomes.
- g) Retirar pessoa:
    - Implemente uma função para retirar uma pessoa da agenda. Após a remoção, todos os elementos seguintes no vetor devem ser deslocados para a posição anterior.

- h) Imprimir agenda:
    - Implemente uma função que imprime a agenda com duas opções:
    - Imprimir apenas nome, telefone e e-mail.
    - Imprimir todos os dados.
- i) Menu principal:
    - Crie um menu principal que ofereça as opções acima para o usuário.

Enunciado 10: Biblioteca

Descrição:

Crie uma estrutura para armazenar os dados de um livro (título, autor, ano de publicação). Crie um array de 3 livros, leia os dados, e imprima-os em ordem alfabética pelo título.

Dica: A função `strcmp` da lib `string.h` funciona da seguinte maneira:

- Retorna um valor negativo se a primeira string (str1) vem antes da segunda string (str2) na ordem lexicográfica.
- Retorna zero se as duas strings são iguais.
- Retorna um valor positivo se a primeira string (str1) vem depois da segunda string (str2) na ordem lexicográfica.

Input de Exemplo:

```
Livro 1: Título: O Alquimista, Autor: Paulo Coelho, Ano de Publicação: 1988
Livro 2: Título: Dom Casmurro, Autor: Machado de Assis, Ano de Publicação:
1899
Livro 3: Título: A Revolução dos Bichos, Autor: George Orwell, Ano de
Publicação: 1945
```

Output Esperado:

```
Livro 3: Título: A Revolução dos Bichos, Autor: George Orwell, Ano de
Publicação: 1945
Livro 2: Título: Dom Casmurro, Autor: Machado de Assis, Ano de Publicação:
1899
Livro 1: Título: O Alquimista, Autor: Paulo Coelho, Ano de Publicação: 1988
```

Extra

Os próximos exercícios são opcionais para aqueles que querem ir além; não haverá acréscimo na nota com os próximos enunciados, é apenas para fixar e se desafiar.

1. Faça um programa em C que simule o jogo campo minado.

Sugestões:

1. O jogo tem uma informação que precisa ser validada pra todos os campos, sendo elas:
    1. é bomba
    2. esta aberto
    3. quantidade vizinhos bombas
2. O jogo pode ser dividido em alguns funções
    1. iniciar tabuleiro
    2. sortear bombas
    3. contar vizinhos bombas
    4. iniciar jogo
    5. dar palpite