Tema 1: Biblioteca de Livros

1. Crie uma classe Livro com os atributos `titulo`, `autor`, `anoPublicacao` e `numeroPaginas` como `private`. Implemente as propriedades `public` com `get` e `set` para acessar esses atributos, aplicando encapsulamento.

public class Livro

    {

        private string Titulo{ get; set; }

        private string Autor{ get; set; }

        private int AnoPublicacao{ get; set; }

        private int NumeroPaginas{ get; set; }

*// Construtor*

        public Livro(string titulo, string autor, int anoPublicacao, int numeroPaginas)

        {

*this*.Titulo = titulo;

*this*.Autor = autor;

*this*.AnoPublicacao = anoPublicacao;

*this*.NumeroPaginas = numeroPaginas;

        }

        public void ExibirDetalhes()

        {

            Console.WriteLine($"Título: {Titulo}, Autor: {Autor}, Ano de Publicação: {AnoPublicacao}, Número de Páginas: {NumeroPaginas}");

        }

    }

}

2. Implemente um construtor na classe Livro que inicialize os atributos `titulo`, `autor`, `anoPublicacao` e `numeroPaginas`. Demonstre o uso do construtor ao instanciar um objeto da classe Livro.

public class Livro

    {

        private string Titulo{ get; set; }

        private string Autor{ get; set; }

        private int AnoPublicacao{ get; set; }

        private int NumeroPaginas{ get; set; }

*// Construtor*

        public Livro(string titulo, string autor, int anoPublicacao, int numeroPaginas)

        {

*this*.Titulo = titulo;

*this*.Autor = autor;

*this*.AnoPublicacao = anoPublicacao;

*this*.NumeroPaginas = numeroPaginas;

        }

}

3. Adicione um método `ExibirDetalhes` na classe Livro que exiba os detalhes do livro. Certifique-se de acessar os atributos `private` através das propriedades `públicas`.

public class Livro

    {

        private string Titulo{ get; set; }

        private string Autor{ get; set; }

        private int AnoPublicacao{ get; set; }

        private int NumeroPaginas{ get; set; }

*// Construtor*

        public Livro(string titulo, string autor, int anoPublicacao, int numeroPaginas)

        {

*this*.Titulo = titulo;

*this*.Autor = autor;

*this*.AnoPublicacao = anoPublicacao;

*this*.NumeroPaginas = numeroPaginas;

        }

        public void ExibirDetalhes()

        {

            Console.WriteLine($"Título: {Titulo}, Autor: {Autor}, Ano de Publicação: {AnoPublicacao}, Número de Páginas: {NumeroPaginas}");

        }

    }

}

4. Crie uma classe Biblioteca com uma lista privada de Livro representando o acervo. Adicione métodos públicos para adicionar e remover livros do acervo, garantindo que a lista só seja manipulada através desses métodos.

5. Implemente um método `BuscarLivroPorTitulo` na classe Biblioteca que permita procurar um livro pelo título. Utilize encapsulamento para garantir que o acesso ao acervo seja controlado.

6. Adicione um método `ListarLivros` na classe Biblioteca que exiba todos os livros. Use o método `ExibirDetalhes` da classe Livro para acessar os atributos privados de cada livro.

7. Crie uma classe Usuario com os atributos privados `nome` e `cpf`, e uma lista privada de Livro representando os livros emprestados. Adicione métodos para emprestar e devolver livros, garantindo que a lista seja manipulada de forma encapsulada.

8. Implemente um método `ExibirLivrosEmprestados` na classe Usuario que exiba todos os livros emprestados. Teste o método, certificando-se de que o acesso aos livros seja feito de forma encapsulada.

9. Adicione um método `DevolverLivro` na classe Usuario que permita devolver um livro à biblioteca, garantindo a integridade dos dados com encapsulamento. Teste o método para garantir que o livro seja removido da lista de livros emprestados.

10. Utilize encapsulamento para proteger a integridade dos dados ao implementar o método `EmprestarLivro` na classe Biblioteca. Certifique-se de que a lista de livros seja acessível apenas por métodos controlados.

Tema 2: Sistema de Gerenciamento de Veículos

11. Crie uma classe Veiculo com os atributos privados `marca`, `modelo`, `ano` e `quilometragem`. Implemente as propriedades públicas para encapsular o acesso a esses atributos.

12. Implemente um construtor na classe Veiculo que inicialize os atributos. Mostre como o construtor pode ser usado para instanciar um objeto da classe Veiculo.

13. Adicione um método `ExibirDetalhes` na classe Veiculo que exiba os detalhes do veículo, garantindo que os atributos privados sejam acessados através de suas propriedades. Use encapsulamento para proteger os dados.

14. Crie uma classe Garagem com uma lista privada de Veiculo representando os veículos disponíveis. Adicione métodos públicos para adicionar e remover veículos, mantendo a integridade da lista através de encapsulamento.

15. Implemente um método `BuscarVeiculoPorModelo` na classe Garagem que permita procurar um veículo pelo modelo. Garanta que o acesso à lista de veículos seja feito de forma encapsulada.

16. Adicione um método `VenderVeiculo` na classe Garagem que permita vender um veículo. Utilize encapsulamento para garantir que a lista seja manipulada apenas por métodos controlados.

17. Crie uma classe Cliente com os atributos privados `nome` e `cpf`, e uma lista privada de Veiculo representando os veículos comprados. Adicione métodos públicos para comprar veículos, garantindo que a lista seja manipulada de forma encapsulada.

18. Implemente um método `ExibirVeiculosComprados` na classe Cliente que exiba todos os veículos comprados. Utilize encapsulamento para garantir que os dados sejam acessados de forma segura.

19. Adicione um método `RealizarManutencao` na classe Veiculo que permita registrar uma manutenção, atualizando a quilometragem. Proteja a integridade dos dados através do encapsulamento.

20. Crie uma classe Concessionaria que combine os conceitos anteriores, garantindo que o acesso a clientes, veículos e vendas seja controlado através de encapsulamento. Implemente métodos que utilizem encapsulamento para realizar operações comuns.

\*\*Tema 2: Sistema de Gerenciamento de Veículos\*\*

11. Crie uma classe Veiculo com os atributos privados `marca`, `modelo`, `ano` e `quilometragem`. Implemente as propriedades públicas para encapsular o acesso a esses atributos.

12. Implemente um construtor na classe Veiculo que inicialize os atributos. Mostre como o construtor pode ser usado para instanciar um objeto da classe Veiculo.

13. Adicione um método `ExibirDetalhes` na classe Veiculo que exiba os detalhes do veículo, garantindo que os atributos privados sejam acessados através de suas propriedades. Use encapsulamento para proteger os dados.

14. Crie uma classe Garagem com uma lista privada de Veiculo representando os veículos disponíveis. Adicione métodos públicos para adicionar e remover veículos, mantendo a integridade da lista através de encapsulamento.

15. Implemente um método `BuscarVeiculoPorModelo` na classe Garagem que permita procurar um veículo pelo modelo. Garanta que o acesso à lista de veículos seja feito de forma encapsulada.

16. Adicione um método `VenderVeiculo` na classe Garagem que permita vender um veículo. Utilize encapsulamento para garantir que a lista seja manipulada apenas por métodos controlados.

17. Crie uma classe Cliente com os atributos privados `nome` e `cpf`, e uma lista privada de Veiculo representando os veículos comprados. Adicione métodos públicos para comprar veículos, garantindo que a lista seja manipulada de forma encapsulada.

18. Implemente um método `ExibirVeiculosComprados` na classe Cliente que exiba todos os veículos comprados. Utilize encapsulamento para garantir que os dados sejam acessados de forma segura.

19. Adicione um método `RealizarManutencao` na classe Veiculo que permita registrar uma manutenção, atualizando a quilometragem. Proteja a integridade dos dados através do encapsulamento.

20. Crie uma classe Concessionaria que combine os conceitos anteriores, garantindo que o acesso a clientes, veículos e vendas seja controlado através de encapsulamento. Implemente métodos que utilizem encapsulamento para realizar operações comuns.