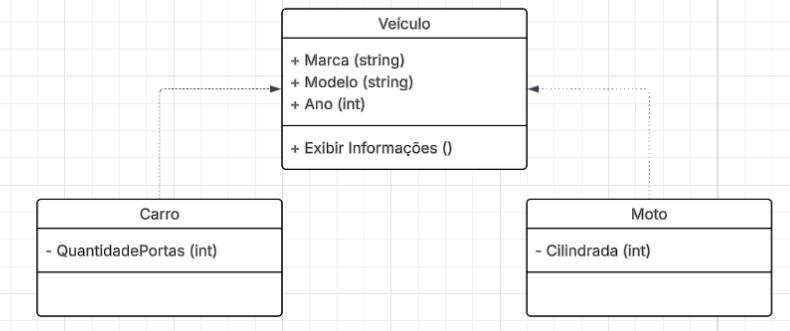
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **QUESTÃO** | **Nº1** | **VALOR** | 1 | **NOTA** |  |

* 1. Faça o diagrama UML que modele uma hierarquia simples de classes para representar veículos.
* Crie uma classe **Veiculo** com os seguintes atributos:
  + Marca (string)
  + Modelo (string)
  + Ano (int)
* Crie um método ExibirInformacoes() na classe **Veiculo** que exibe as informações do veículo na tela.
* Crie duas classes derivadas de **Veiculo**:
  + **Carro**: Adicione um atributo QuantidadePortas (int).
  + **Moto**: Adicione um atributo Cilindrada (int).



* 1. Implemente o código em C# com base no diagrama UML:

No método Main, crie um objeto da classe **Carro** e um da classe **Moto**, defina seus

atributos e exiba suas informações.

3.1) Altere os modificadores de acesso dos atributos para private e depois protect.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **QUESTÃO** | **Nº2** | **VALOR** | 1 | **NOTA** |  |

* 1. Faça o diagrama UML que modele uma hierarquia simples de classes para modelar **contas bancárias** usando herança.
* Crie uma classe **ContaBancaria** com os seguintes atributos:

Titular (string)

NumeroConta (int)

Saldo (double)

* Crie um método ExibirDados() na classe **ContaBancaria** para mostrar as informações da conta.
* Crie duas classes derivadas:

**ContaCorrente**: Adicione um atributo TaxaManutencao (double).

**ContaPoupanca**: Adicione um atributo TaxaJuros (double).

* 1. Implemente o código em C# com base no diagrama UML:

No método Main, crie um objeto de cada classe, defina os atributos e exiba suas informações..

* 1. Altere os modificadores de acesso dos atributos para private e depois protect.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **QUESTÃO** | **Nº3** | **VALOR** | 1 | **NOTA** |  |

**Herança Única**

Crie uma classe base chamada Funcionario com os atributos protected string nome e protected double salarioBase. A classe deve ter um método ExibirDados() que exibe as informações do funcionário.

Em seguida, crie uma classe derivada Gerente, que herda de Funcionario. A classe Gerente deve adicionar um atributo protected double bonus e um método ExibirDadosGerente() que exibe as informações do gerente, incluindo o bônus e o salário total.

Crie um programa que instancie um objeto da classe Gerente, atribua valores aos atributos e exiba os dados.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **QUESTÃO** | **Nº4** | **VALOR** | 1 | **NOTA** |  |

**Herança Multinível**

Crie uma classe base Veiculo com os atributos protected string marca e protected int ano. Implemente um método ExibirDetalhes() para mostrar esses dados.

Crie uma classe Carro que herda de Veiculo e adiciona um atributo protected int portas. Crie um método ExibirDetalhesCarro() que exibe as informações do veículo e a quantidade de portas.

Crie uma classe CarroEletrico que herda de Carro e adiciona protected int autonomiaBateria. A classe deve ter um método ExibirDetalhesCarroEletrico() que exibe todas as informações do veículo, incluindo a autonomia da bateria.

No programa principal, instancie um CarroEletrico, atribua valores e exiba os detalhes.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **QUESTÃO** | **Nº5** | **VALOR** | 1 | **NOTA** |  |

5.1) - Crie uma interface IVoar com um método void Voar().

5.2) - Crie uma classe base Animal com os atributos protected string nome e protected string especie. Essa classe deve ter um método ExibirInfo() para exibir os dados do animal.

5.3) - Crie duas classes Cachorro e Passaro que herdam de Animal e implementam a interface IVoar. A classe Passaro deve, além de herdar a classe Animal, incluir a interface IVoar que exibe uma mensagem informando que o animal está voando.

5.4) - No programa principal, instancie objetos de Cachorro e Passaro, atribua valores e chame os métodos ExibirInfo() e Voar() caso necessário.