**PARTE 1 – PROGRAMAÇÃO ESTRUTURADA.**

**Lista 13 de exercícios.**

**Instruções.**

As listas de exercícios desenvolvidas com ferramentas de inteligência artificial, e visa ajudá-lo na prática da programação. Você deve tentar fazer os exercícios sem olhar na solução, porém no início é muito natural olhar na solução para entender as propostas.

Sugestão para cada lista crie uma solução com o nome da Lista. Por exemplo, SolucaoLista1 e depois crie um projeto com o nome do exercício e da lista, por exemplo, L01EX01.

Seria interessante que após fazer o exercício, porta-se o mesmo para uma DLL, e chama-se a classe da dll no projeto principal.

Sumário

[**Lista de Exercícios: Usando a Instrução while em C#** 3](#_Toc160389632)

[1. Exercício 1: Contagem até 10 3](#_Toc160389633)

[2. Exercício 2: Contagem Regressiva 3](#_Toc160389634)

[3. Exercício 3: Soma dos Primeiros 100 Números 3](#_Toc160389635)

[4. Exercício 4: Exibir Números Pares até 20 3](#_Toc160389636)

[5. Exercício 5: Tabela de Multiplicação 3](#_Toc160389637)

[6. Exercício 6: Fatorial de um Número 3](#_Toc160389638)

[7. Exercício 7: Soma de Entradas do Usuário 3](#_Toc160389639)

[8. Exercício 8: Lista de Quadrados 3](#_Toc160389640)

[9. Exercício 9: Contagem de Letras 3](#_Toc160389641)

[10. Exercício 10: Inversão de String 3](#_Toc160389642)

[**Soluções dos Exercícios sobre Instrução while** 4](#_Toc160389643)

[1. Solução do Exercício 1 4](#_Toc160389644)

[2. Solução do Exercício 2 4](#_Toc160389645)

[3. Solução do Exercício 3 4](#_Toc160389646)

[4. Solução do Exercício 4 4](#_Toc160389647)

[5. Solução do Exercício 5 4](#_Toc160389648)

[6. Solução do Exercício 6 4](#_Toc160389649)

[7. Solução do Exercício 7 4](#_Toc160389650)

[8. Solução do Exercício 8 5](#_Toc160389651)

[9. Solução do Exercício 9 5](#_Toc160389652)

[10. Solução do Exercício 10 5](#_Toc160389653)

**Lista de Exercícios: Herança em C#**

Exercício 1: Veículo e Carro

* **Tarefa**: Crie uma classe base **Veiculo** com um método **ExibirInfo** e uma propriedade **Marca**. Depois, crie uma classe **Carro** que herda de **Veiculo** e adiciona uma propriedade **Modelo**.

Exercício 2: Pessoa e Aluno

* **Tarefa**: Defina uma classe base **Pessoa** com propriedades **Nome** e **Idade** e um método **Exibir**. Em seguida, crie uma classe **Aluno**, derivada de **Pessoa**, que inclui uma propriedade adicional **Matricula** e sobrescreve o método **Exibir**.

Exercício 3: Animal e Cachorro

* **Tarefa**: Implemente uma classe **Animal** com uma propriedade **Especie** e um método **EmitirSom**. Derive uma classe **Cachorro** de **Animal** que sobrescreve **EmitirSom** para exibir "Au au".

Exercício 4: Funcionário e Gerente

* **Tarefa**: Crie uma classe **Funcionario** com propriedades **Nome** e **Salario** e um método **ExibirInfo**. Crie uma classe **Gerente** que herda de **Funcionario** e adiciona uma propriedade **Departamento**.

Exercício 5: Forma e Círculo

* **Tarefa**: Defina uma classe abstrata **Forma** com um método abstrato **CalcularArea**. Crie uma classe **Circulo** que herda de **Forma**, implementa **CalcularArea**, e possui uma propriedade **Raio**.

**Soluções dos Exercícios de Herança**

Solução do Exercício 1: Veículo e Carro

* csharpCopy code
* public class Veiculo { public string Marca { get; set; } public void ExibirInfo() { Console.WriteLine($"Marca: {Marca}"); } } public class Carro : Veiculo { public string Modelo { get; set; } // Utilize ExibirInfo() herdado sem modificações }

Solução do Exercício 2: Pessoa e Aluno

* csharpCopy code
* public class Pessoa { public string Nome { get; set; } public int Idade { get; set; } public virtual void Exibir() { Console.WriteLine($"Nome: {Nome}, Idade: {Idade}"); } } public class Aluno : Pessoa { public string Matricula { get; set; } public override void Exibir() { base.Exibir(); Console.WriteLine($"Matrícula: {Matricula}"); } }

Solução do Exercício 3: Animal e Cachorro

* csharpCopy code
* public class Animal { public string Especie { get; set; } public virtual void EmitirSom() { Console.WriteLine("Som do animal"); } } public class Cachorro : Animal { public override void EmitirSom() { Console.WriteLine("Au au"); } }

Solução do Exercício 4: Funcionário e Gerente

* csharpCopy code
* public class Funcionario { public string Nome { get; set; } public double Salario { get; set; } public void ExibirInfo() { Console.WriteLine($"Nome: {Nome}, Salário: {Salario}"); } } public class Gerente : Funcionario { public string Departamento { get; set; } // Utilize ExibirInfo() herdado, adicione mais funcionalidades se necessário }

Solução do Exercício 5: Forma e Círculo

* csharpCopy code
* public abstract class Forma { public abstract double CalcularArea(); } public class Circulo : Forma { public double Raio { get; set; } public override double CalcularArea() { return Math.PI \* Raio \* Raio; } }