**PARTE 1 – PROGRAMAÇÃO ESTRUTURADA.**

**Lista 15 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: Sobrescrita de Métodos

Exercício 1: Forma Geométrica

* Tarefa: Crie uma classe base Forma com um método virtual Desenhar(). Derive duas classes: Circulo e Quadrado, sobrescrevendo o método Desenhar() para exibir mensagens diferentes.

Exercício 2: Animal Falante

* Tarefa: Implemente uma classe base Animal com um método virtual EmitirSom(). Crie classes derivadas Cachorro e Gato, sobrescrevendo o método EmitirSom() para refletir seus sons característicos.

Exercício 3: Funcionário

* Tarefa: Crie uma classe Funcionario com um método virtual CalcularBonus(). Derive uma classe Gerente e uma classe Desenvolvedor, sobrescrevendo o método CalcularBonus() para calcular o bônus com base em regras diferentes.

Exercício 4: Veículo

* Tarefa: Desenvolva uma classe base Veiculo com um método virtual Acelerar(). Crie classes derivadas Carro e Moto, sobrescrevendo o método Acelerar() para incluir comportamentos específicos de cada veículo.

Exercício 5: Dispositivo Eletrônico

* Tarefa: Implemente uma classe Dispositivo com um método virtual Ligar(). Derive classes Smartphone e Tablet, sobrescrevendo o método Ligar() para mostrar mensagens customizadas.

Soluções dos Exercícios de Sobrescrita de Métodos

Solução do Exercício 1: Forma Geométrica

* public class Forma { public virtual void Desenhar() { Console.WriteLine("Desenhando uma forma."); } } public class Circulo : Forma { public override void Desenhar() { Console.WriteLine("Desenhando um círculo."); } } public class Quadrado : Forma { public override void Desenhar() { Console.WriteLine("Desenhando um quadrado."); } }

Solução do Exercício 2: Animal Falante

public class Animal { public virtual void EmitirSom() { Console.WriteLine("Som do animal."); } } public class Cachorro : Animal { public override void EmitirSom() { Console.WriteLine("Au au."); } } public class Gato : Animal { public override void EmitirSom() { Console.WriteLine("Miau."); } }

Solução do Exercício 3: Funcionário

* public class Funcionario { public virtual double CalcularBonus() { return 1000; // Valor base do bônus } } public class Gerente : Funcionario { public override double CalcularBonus() { return base.CalcularBonus() \* 2; // Bônus dobrado para gerentes } } public class Desenvolvedor : Funcionario { public override double CalcularBonus() { return base.CalcularBonus() + 500; // Bônus adicional para desenvolvedores } }

Solução do Exercício 4: Veículo

* public class Veiculo { public virtual void Acelerar() { Console.WriteLine("Veículo acelerando."); } } public class Carro : Veiculo { public override void Acelerar() { Console.WriteLine("Carro acelerando rapidamente."); } } public class Moto : Veiculo { public override void Acelerar() { Console.WriteLine("Moto acelerando com agilidade."); } }

Solução do Exercício 5: Dispositivo Eletrônico

* public class Dispositivo { public virtual void Ligar() { Console.WriteLine("Dispositivo ligando."); } } public class Smartphone : Dispositivo { public override void Ligar() { Console.WriteLine("Smartphone ligando. Bem-vindo!"); } } public class Tablet : Dispositivo { public override void Ligar() { Console.WriteLine("Tablet ligando. Pronto para usar."); } }