**PARTE 1 – PROGRAMAÇÃO ESTRUTURADA.**

**Lista 16 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: Polimorfismo entre Classes**

Exercício 1: Animais Fazenda

* **Tarefa**: Crie uma classe base **Animal** com um método **EmitirSom()**. Derive classes **Vaca**, **Galinha** e **Porco** da classe **Animal**, cada uma sobrescrevendo o método **EmitirSom()** para refletir seu som característico.

Exercício 2: Formas de Pagamento

* **Tarefa**: Implemente uma classe abstrata **Pagamento** com um método **EfetuarPagamento()**. Crie classes derivadas **PagamentoBoleto**, **PagamentoCartao** e **PagamentoPix** que implementem o método **EfetuarPagamento** de maneiras específicas para cada forma de pagamento.

Exercício 3: Veículos de Transporte

* **Tarefa**: Desenvolva uma classe base **Veiculo** com um método **Acelerar()**. Crie classes derivadas **Carro**, **Bicicleta** e **Barco**, implementando o método **Acelerar()** de forma única em cada classe.

Exercício 4: Dispositivos Eletrônicos

* **Tarefa**: Crie uma classe base **DispositivoEletronico** com um método **Ligar()**. Derive classes **Smartphone**, **Televisao** e **Notebook**, cada uma com uma implementação específica do método **Ligar()**.

Exercício 5: Trabalhadores

* **Tarefa**: Implemente uma classe abstrata **Trabalhador** com um método abstrato **Trabalhar()**. Crie classes derivadas **Programador**, **Designer** e **Gerente**, cada uma com sua própria versão do método **Trabalhar()**.

**Soluções dos Exercícios de Polimorfismo entre Classes**

Solução do Exercício 1: Animais Fazenda

public abstract class Animal { public abstract void EmitirSom(); } public class Vaca : Animal { public override void EmitirSom() { Console.WriteLine("Muuu"); } } public class Galinha : Animal { public override void EmitirSom() { Console.WriteLine("Cocoricó"); } } public class Porco : Animal { public override void EmitirSom() { Console.WriteLine("Oinc oinc"); } }

Solução do Exercício 2: Formas de Pagamento

public abstract class Pagamento { public abstract void EfetuarPagamento(); } public class PagamentoBoleto : Pagamento { public override void EfetuarPagamento() { Console.WriteLine("Pagando com boleto"); } } public class PagamentoCartao : Pagamento { public override void EfetuarPagamento() { Console.WriteLine("Pagando com cartão"); } } public class PagamentoPix : Pagamento { public override void EfetuarPagamento() { Console.WriteLine("Pagando com Pix"); } }

Solução do Exercício 3: Veículos de Transporte

public abstract class Veiculo { public abstract void Acelerar(); } public class Carro : Veiculo { public override void Acelerar() { Console.WriteLine("Carro acelerando"); } } public class Bicicleta : Veiculo { public override void Acelerar() { Console.WriteLine("Bicicleta ganhando velocidade"); } } public class Barco : Veiculo { public override void Acelerar() { Console.WriteLine("Barco aumentando a velocidade"); } }

Solução do Exercício 4: Dispositivos Eletrônicos

* public abstract class DispositivoEletronico { public abstract void Ligar(); } public class Smartphone : DispositivoEletronico { public override void Ligar() { Console.WriteLine("Smartphone ligando"); } } public class Televisao : DispositivoEletronico { public override void Ligar() { Console.WriteLine("Televisão ligando"); } } public class Notebook : DispositivoEletronico { public override void Ligar() { Console.WriteLine("Notebook iniciando"); } }

Solução do Exercício 5: Trabalhadores

* public abstract class Trabalhador { public abstract void Trabalhar(); } public class Programador : Trabalhador { public override void Trabalhar() { Console.WriteLine("Programador escrevendo código"); } } public class Designer : Trabalhador { public override void Trabalhar() { Console.WriteLine("Designer criando interfaces"); } } public class Gerente : Trabalhador { public override void Trabalhar() { Console.WriteLine("Gerente coordenando projetos"); } }