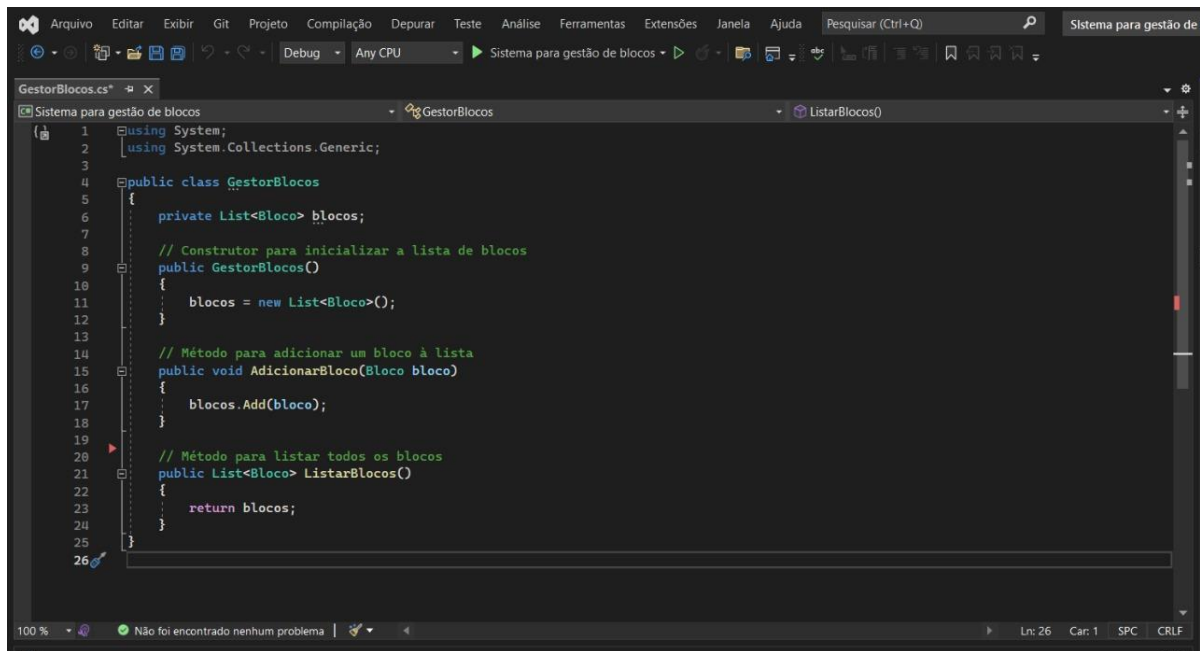


Alunos: Guilherme Sabino de Oliveira, Arthur Costalonga Colli

Link do git: <https://github.com/Guilherme1593/Sistema-para-gest-o-de-blocos.git>

A screenshot of the Visual Studio IDE showing a C# file named GestorBlocos.cs. The code defines a class GestorBlocos with a private List<Bloco> named blocos. It includes a constructor, an Add method, and a Listar method. The interface is in Portuguese. The status bar at the bottom indicates '100 %' zoom and 'Não foi encontrado nenhum problema' (No problems found).

```
1 using System;
2 using System.Collections.Generic;
3
4 public class GestorBlocos
5 {
6     private List<Bloco> blocos;
7
8     // Construtor para inicializar a lista de blocos
9     public GestorBlocos()
10    {
11        blocos = new List<Bloco>();
12    }
13
14    // Método para adicionar um bloco à lista
15    public void AdicionarBloco(Bloco bloco)
16    {
17        blocos.Add(bloco);
18    }
19
20    // Método para listar todos os blocos
21    public List<Bloco> ListarBlocos()
22    {
23        return blocos;
24    }
25 }
26
```

## GestorBlocos

A classe GestorBlocos é responsável por gerenciar uma coleção de blocos. Ela possui uma lista privada de objetos Bloco e métodos para adicionar, remover e listar blocos.

**blocos:** Lista privada que armazena os objetos Bloco.

**AdicionarBloco (Bloco bloco):** Adiciona um bloco à

lista..**ListarBlocos():** Retorna a lista de blocos atual.

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6
7
8 namespace Sistema_para_gestão_de_blocos
9 {
10     internal class Valordecompra
11     {
12         static void CadastrarBloco(GestorBlocos gestor)
13         {
14             Console.WriteLine("Número do Bloco: ");
15             string numero = Console.ReadLine();
16
17             double valorCompra, valorVenda;
18
19             do
20             {
21                 Console.WriteLine("Valor de Compra: ");
22             } while (!double.TryParse(Console.ReadLine(), out valorCompra));
23
24             do
25             {
26                 Console.WriteLine("Valor de Venda: ");
27             } while (!double.TryParse(Console.ReadLine(), out valorVenda));
28
29             Console.WriteLine("Nome do Bloco: ");
30             string nome = Console.ReadLine();
31
32             Console.WriteLine("Comprimento (mm): ");
33             double comprimento = double.Parse(Console.ReadLine());
34
35             Console.WriteLine("Largura (mm): ");
36             double largura = double.Parse(Console.ReadLine());
37
38             Console.WriteLine("Espessura (mm): ");
39             double espessura = double.Parse(Console.ReadLine());
40
41             Console.WriteLine("Material: ");
42             string material = Console.ReadLine();
43
44             // Cria um objeto Bloco com os dados fornecidos pelo usuário
45             Bloco bloco = new Bloco(nome, comprimento, largura, espessura, material);
46
47             // Define as propriedades de número, valor de compra e valor de venda
48             bloco.Numero = numero;
49             bloco.ValorCompra = valorCompra;
50             bloco.ValorVenda = valorVenda;
51
52             // Adiciona o bloco à lista gerenciada pelo GestorBlocos
53             gestor.AdicionarBloco(bloco);
54
55             Console.WriteLine("Bloco cadastrado com sucesso!");
56         }
57     }
58 }
59
60
```

A função ValorDeCompra é uma função que registra o valor de compra de um bloco específico. No contexto do programa, ela deve atualizar a propriedade ValorCompra de um objeto Bloco com o valor de compra informado pelo usuário.

Nesta função:

bloco: É o objeto Bloco ao qual você deseja atribuir o valor de compra.

valorCompra: É o valor de compra que você deseja registrar para o bloco.

Quando você chama essa função e passa um objeto Bloco e um valor de compra, ela atualiza a propriedade ValorCompra desse bloco com o valor fornecido.

```
Sistema para gestão de blocos Bloco
4 {
5     // Propriedades para armazenar informações sobre o bloco
6     public string Nome { get; set; }
7     public double Comprimento { get; set; }
8     public double Largura { get; set; }
9     public double Espessura { get; set; }
10    public string Material { get; set; }
11    public string? Numero { get; internal set; }
12    public double ValorCompra { get; internal set; }
13    public double ValorVenda { get; internal set; }
14
15    // Construtor para criar um objeto Bloco com dados iniciais
16    public Bloco(string nome, double comprimento, double largura, double espessura, string material)
17    {
18        Nome = nome;
19        Comprimento = comprimento;
20        Largura = largura;
21        Espessura = espessura;
22        Material = material;
23    }
24
25    // Método para calcular o volume do bloco
26    public double CalcularVolume()
27    {
28        return Comprimento * Largura * Espessura;
29    }
30
31    // Sobrescrevendo o método ToString para exibir informações do bloco
32    public override string ToString()
33    {
34        double lucro = CalcularLucro();
35        return $"{Nome} - Material: {Material}, Dimensões: {Comprimento}x{Largura}x{Espessura} mm, Lucro de Venda: {lucro:C}";
36    }
37
38    // Dentro da classe Bloco
39
40    public double CalcularLucro()
41    {
42        // Calcula o lucro subtraindo o custo de compra do valor de venda
43        return ValorVenda - ValorCompra;
44    }
45
46 }
47
```

## Bloco

A classe Bloco representa um bloco de mármore ou granito. Ela contém propriedades para armazenar informações sobre o bloco, como nome, comprimento, largura, espessura, material, número, valor de compra e valor de venda. Além disso, a classe possui métodos para calcular o volume do bloco e o lucro de venda.

Nome: Armazena o nome do bloco.

Comprimento, Largura, Espessura: Armazenam as dimensões do bloco

Material: Armazena o material do bloco.

Numero: Armazena o número do bloco.

ValorCompra: Armazena o valor de compra do bloco.

ValorVenda: Armazena o valor de venda do bloco.

CalcularLucro(): Calcula o lucro de venda subtraindo o valor de compra do valor de venda.

ToString(): Sobrescrito para exibir informações formatadas do bloco, incluindo o lucro de venda.

```
Sistema para gestao de blocos
1 using System;
2
3 namespace Sistema_para_gestao_de_blocos
4 {
5     O referências
6     class Program
7     {
8         private static readonly object meuBloco;
9
10        O referências
11        static void Main(string[] args)
12        {
13            GestorBlocos gestor = new GestorBlocos();
14
15            while (true)
16            {
17                Console.WriteLine("Menu:");
18                Console.WriteLine("1. Cadastrar Bloco");
19                Console.WriteLine("2. Listar Blocos");
20                Console.WriteLine("3. Sair");
21                Console.WriteLine("Escolha uma opção: ");
22
23                string opcao = Console.ReadLine();
24
25                switch (opcao)
26                {
27                    case "1":
28                        CadastrarBloco(gestor);
29                        break;
30
31                    case "2":
32                        Console.WriteLine("Lista de Blocos:");
33                        var blocos = gestor.ListarBlocos();
34                        foreach (var b in blocos)
35                        {
36                            Console.WriteLine(b);
37                        }
38                        break;
39
40                    case "3":
41                        Console.WriteLine("Saindo do programa.");
42                        Environment.Exit(0);
43                        break;
44
45                    default:
46                        Console.WriteLine("Opção inválida. Tente novamente.");
47                        break;
48                }
49            }
50
51            O referências
52            private static void CalcularLucro(GestorBlocos gestor)
53            {
54                throw new NotImplementedException();
55            }
56
57            // Função para cadastrar um bloco com número, valor de compra e valor de venda
58            1 referências
59            static void CadastrarBloco(GestorBlocos gestor)
60            {
61                Console.WriteLine("Número do Bloco: ");
62                string numero = Console.ReadLine();
63                Console.WriteLine("Valor de Compra: ");
```

## Program.CS

A classe Program é a classe de entrada do programa. Ela contém o método Main, que é o ponto de partida da execução. No método Main, o programa exibe um menu de opções para o usuário, permitindo cadastrar blocos, listar blocos, calcular lucro e sair do programa.

Main(string[] args): O ponto de entrada do programa. Controla o menu principal e as interações com o usuário.

CadastrarBloco(GestorBlocos gestor): Função para cadastrar um bloco com informações fornecidas pelo usuário.

CalcularLucro(GestorBlocos gestor): Função para calcular o lucro de venda de um bloco com base no número do bloco fornecido pelo usuário

Programa testado.

```
C:\Users\arthur\Source\Repos' x + v
Menu:
1. Cadastrar Bloco
2. Listar Blocos
3. Sair
Escolha uma opção: 1
Número do Bloco: 1203
Valor de Compra: 35000,00
Valor de Venda: 58500,78
Nome do Bloco: Cinza Itabira
Comprimento (mm): 4500
Largura (mm): 5000
Espessura (mm): 5500
Material: Granito branco
Lucro de venda do bloco: 23500,78
Bloco cadastrado com sucesso!
Menu:
1. Cadastrar Bloco
2. Listar Blocos
3. Sair
Escolha uma opção: 1
Número do Bloco: 1534
Valor de Compra: 23450,98
Valor de Venda: 30400,15
Nome do Bloco: Preto São Gabriel
Comprimento (mm): 3500
Largura (mm): 3500
Espessura (mm): 2700
Material: Granito Preto
Lucro de venda do bloco: 6949,1700000000002
Bloco cadastrado com sucesso!
Menu:
1. Cadastrar Bloco
2. Listar Blocos
3. Sair
Escolha uma opção: 2
Lista de Blocos:
Cinza Itabira - Material: Granito branco, Dimensões: 4500x5000x5500 mm, Lucro de Venda: R$ 23.500,78
Preto São Gabriel - Material: Granito Preto, Dimensões: 3500x3500x2700 mm, Lucro de Venda: R$ 6.949,17
Menu:
1. Cadastrar Bloco
2. Listar Blocos
3. Sair
```

