DS2
Orientação a Objetos

SERVIÇO NACIONAL DE APRENDIZAGEM INDUSTRIAL

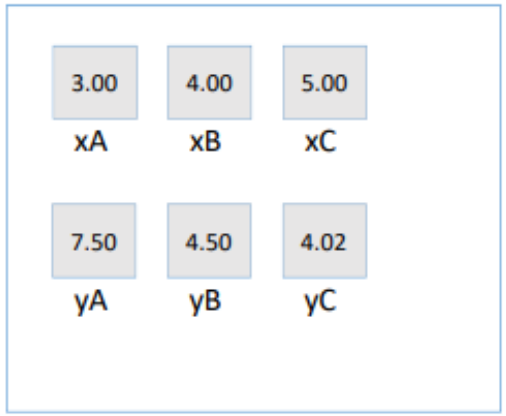
Triângulo é uma entidade com três atributos: a, b, c.

Estamos usando três variáveis distintas para representar cada triângulo:

```
double xA, xB, xC, yA, yB, yC;
```

Para melhorar isso, vamos usar uma CLASSE para representar um triângulo.

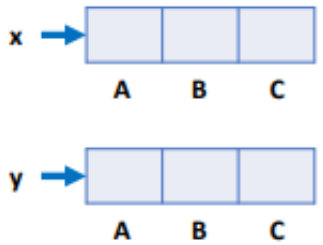
Memória:




```
namespace Course {
    class Triangulo {

        public double A;
        public double B;
        public double C;
    }
}
```

```
double xA, xB, xC, yA, yB, yC;
```

```
Triangulo x, y;
x = new Triangulo();
y = new Triangulo();
```

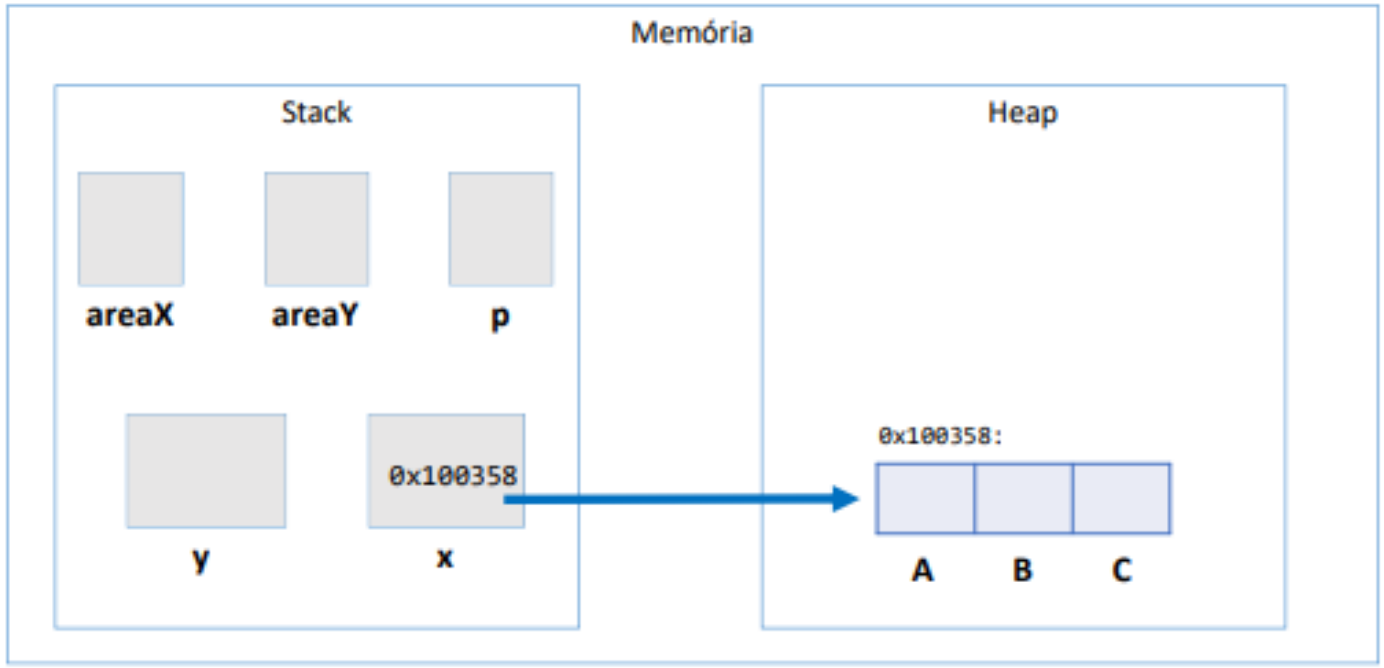


DS2
Orientação a Objetos

Instanciação

(alocação dinâmica de memória)

```
double areaX, areaY, p;
Triangulo x, y;

x = new Triangulo();
```



DS2
Orientação a Objetos

SERVIÇO NACIONAL DE
APRENDIZAGEM INDUSTRIAL

Classe

- É um tipo estruturado que pode conter (membros):
  - Atributos (dados / campos)
  - Métodos (funções / operações)

- A classe também pode prover muitos outros recursos, tais como:
  - Construtores
  - Sobrecarga
  - Encapsulamento
  - Herança
  - Polimorfismo

DS2
Orientação a Objetos

Classes, objetos, atributos

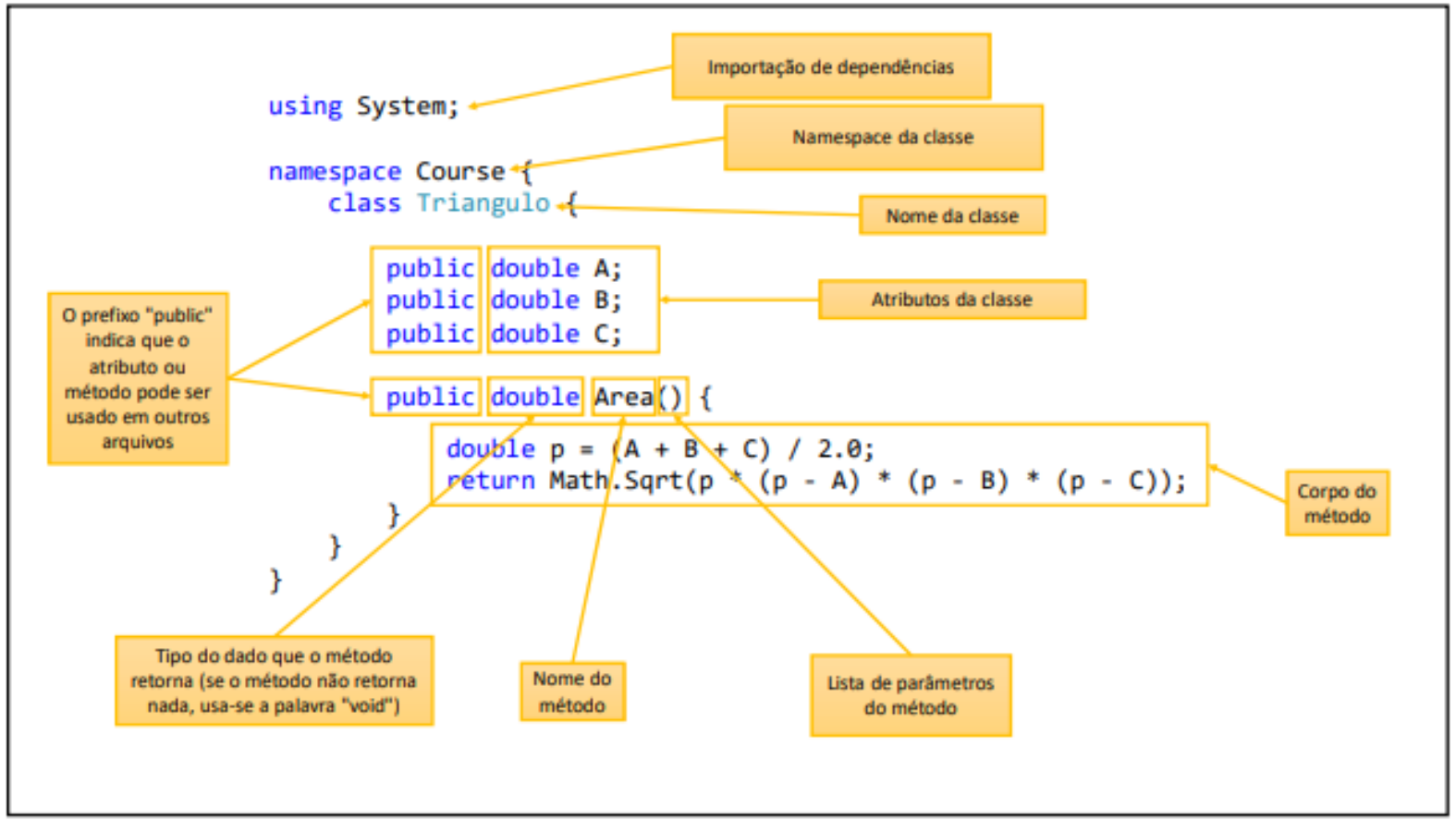
- Classe: é a definição do tipo

```
namespace Course {
    class Triangulo {

        public double A;
        public double B;
        public double C;
    }
}
```

- Objetos: são instâncias da classe

DS2
Orientação a Objetos

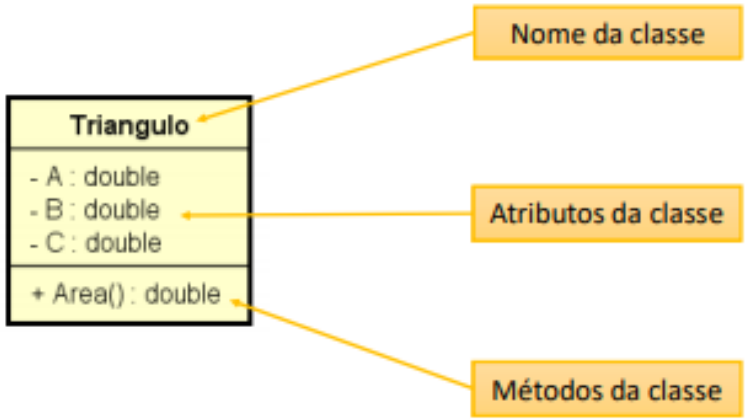
SERVIÇO NACIONAL DE APRENDIZAGEM INDUSTRIAL



DS2
Orientação a Objetos

SERVIÇO NACIONAL DE APRENDIZAGEM INDUSTRIAL

Projeto da classe (UML)



DS2
Orientação a Objetos

Quais são os benefícios de se calcular a área de um triângulo por meio de um MÉTODO dentro da CLASSE Triangulo?

1) Reaproveitamento de código: nós eliminamos o código repetido (cálculo das áreas dos triângulos x e y) no programa principal.

2) Delegação de responsabilidades: quem deve ser responsável por saber como calcular a área de um triângulo é o próprio triângulo. A lógica do cálculo da área não deve estar em outro lugar.

DS2
Orientação a Objetos

Construtor

- É uma operação especial da classe, que executa no momento da instanciação do objeto

- Usos comuns:
  - Iniciar valores dos atributos
  - Permitir ou obrigar que o objeto receba dados / dependências no momento de sua instanciação (injeção de dependência)

- Se um construtor customizado não for especificado, a classe disponibiliza o construtor padrão:
  - Produto p = new Produto();

- É possível especificar mais de um construtor na mesma classe (sobrecarga)

DS2
Orientação a Objetos

Sobrecarga

- É um recurso que uma classe possui de oferecer mais de uma operação com o mesmo nome, porém com diferentes listas de parâmetros.

```
public Produto() {
}

public Produto(string nome, double preco, int quantidade) {
    Nome = nome;
    Preco = preco;
    Quantidade = quantidade;
}

public Produto(string nome, double preco) {
    Nome = nome;
    Preco = preco;
    Quantidade = 0;
}
```

DS2
Orientação a Objetos


```
using System.Globalization;

namespace Course {
    class Produto {

        public string Nome;
        public double Preco;
        public int Quantidade;

        public Produto() {
        }

        public Produto(string nome, double preco, int quantidade) {
            Nome = nome;
            Preco = preco;
            Quantidade = quantidade;
        }

        (...)
```

```
Produto p = new Produto("TV", 900.00, 10);
```

DS2
Orientação a Objetos

Palavra this

- É uma referência para o próprio objeto

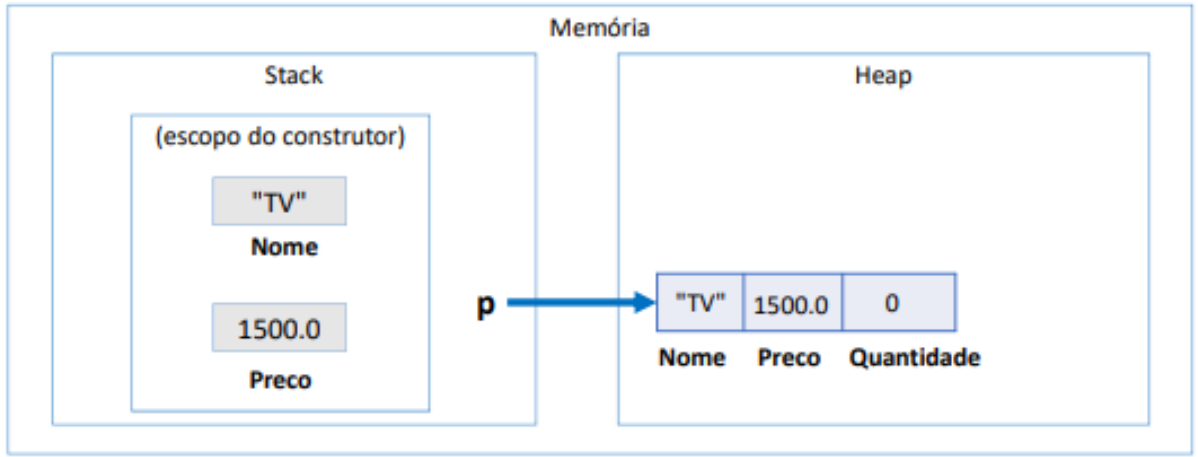
- Usos comuns:
  - Diferenciar atributos de variáveis locais (Java)
  - Referenciar outro construtor em um construtor

DS2
Orientação a Objetos

Diferenciar atributos de variáveis locais

```
Produto p = new Produto("TV", 1500.0);
```

```
public Produto(string Nome, double Preco) {
    this.Nome = Nome;
    this.Preco = Preco;
    Quantidade = 0;
}
```



DS2
Orientação a Objetos

Referenciar outro construtor em um construtor

```
using System.Globalization;

namespace Course {
    class Produto {

        public string Nome;
        public double Preco;
        public int Quantidade;

        public Produto() {
            Quantidade = 0;
        }

        public Produto(string nome, double preco) : this() {
            Nome = nome;
            Preco = preco;
        }

        public Produto(string nome, double preco, int quantidade) : this(nome, preco) {
            Quantidade = quantidade;
        }

        (...)
```

DS2
Orientação a Objetos

SERVIÇO NACIONAL DE APRENDIZAGEM INDUSTRIAL

Encapsulamento

- É um princípio que consiste em esconder detalhes de implementação de um componente, expondo apenas operações seguras e que o mantenha em um estado consistente.

- Regra de ouro: o objeto deve sempre estar em um estado consistente, e a própria classe deve garantir isso.

```
using System.Globalization;

namespace Course {
    class Produto {

        private string _nome;
        private double _preco;
        private int _quantidade;

        public Produto() {
        }

        public Produto(string nome, double preco, int quantidade) {
            _nome = nome;
            _preco = preco;
            _quantidade = quantidade;
        }

        public string GetNome() {
            return _nome;
        }

        public void SetNome(string nome) {
            if (nome != null && nome.Length > 1) {
                _nome = nome;
            }
        }

        public double GetPreco() {
            return _preco;
        }
```

DS2
Orientação a Objetos

Propriedades

- São definições de métodos encapsulados, porém expondo uma sintaxe similar à de atributos e não de métodos

- https://docs.microsoft.com/pt-br/dotnet/csharp/programming-guide/classes-and-structs/properties
  - Uma propriedade é um membro que oferece um mecanismo flexível para ler, gravar ou calcular o valor de um campo particular. As propriedades podem ser usadas como se fossem atributos públicos, mas na verdade elas são métodos especiais chamados "acessadores". Isso permite que os dados sejam acessados facilmente e ainda ajuda a promover a segurança e a flexibilidade dos métodos.

DS2
Orientação a Objetos

SERVIÇO NACIONAL DE APRENDIZAGEM INDUSTRIAL

```
using System.Globalization;

namespace Course {
    class Produto {

        private string _nome;
        private double _preco;
        private int _quantidade;

        public Produto() {
        }

        public Produto(string nome, double preco, int quantidade) {
            _nome = nome;
            _preco = preco;
            _quantidade = quantidade;
        }

        public string Nome {
            get { return _nome; }
            set {
                if (value != null && value.Length > 1) {
                    _nome = value;
                }
            }
        }

        public double Preco {
            get { return _preco; }
        }

        public int Quantidade {
            get { return _quantidade; }
        }

        public double ValorTotalEmEstoque {
            get { return _preco * _quantidade; }
        }

        public void AdicionarProdutos(int quantidade) {
            _quantidade += quantidade;
        }

        public void RemoverProdutos(int quantidade) {
            _quantidade -= quantidade;
        }

        public override string ToString() {
            return _nome
                + ", $ "
                + _preco.ToString("F2", CultureInfo.InvariantCulture)
                + ", "
                + _quantidade
                + " unidades, Total: $ "
                + ValorTotalEmEstoque.ToString("F2", CultureInfo.InvariantCulture);
        }
    }
}
```

DS2
Orientação a Objetos

Propriedades autoimplementadas

- É uma forma simplificada de se declarar propriedades que não necessitam lógicas particulares para as operações get e set.

DS2 Orientação a Objetos

```
public double Preco { get; private set; }
```

https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/classes-and-structs/auto-implemented-properties

DS2
Orientação a Objetos


```
using System.Globalization;

namespace Course {
    class Produto {

        private string _nome;
        public double Preco { get; private set; }
        public double Quantidade { get; set; }

        public Produto() {
        }

        public Produto(string nome, double preco, int quantidade) {
            _nome = nome;
            Preco = preco;
            Quantidade = quantidade;
        }

        public string Nome {
            get { return _nome; }
            set {
                if (value != null && value.Length > 1) {
                    _nome = value;
                }
            }
        }
```

DS2
Orientação a Objetos

Ordem sugerida

- Atributos privados
- Propriedades autoimplementadas
- Construtores
- Propriedades customizadas
- Outros métodos da classe

DS2
Orientação a Objetos

- Toda classe em C# é uma subclasse da classe Object

- Object possui os seguintes métodos:
  - GetType - retorna o tipo do objeto
  - Equals - compara se o objeto é igual a outro
  - GetHashCode - retorna um código hash do objeto
  - ToString - converte o objeto para string

DS2
Orientação a Objetos

SERVIÇO NACIONAL DE APRENDIZAGEM INDUSTRIAL

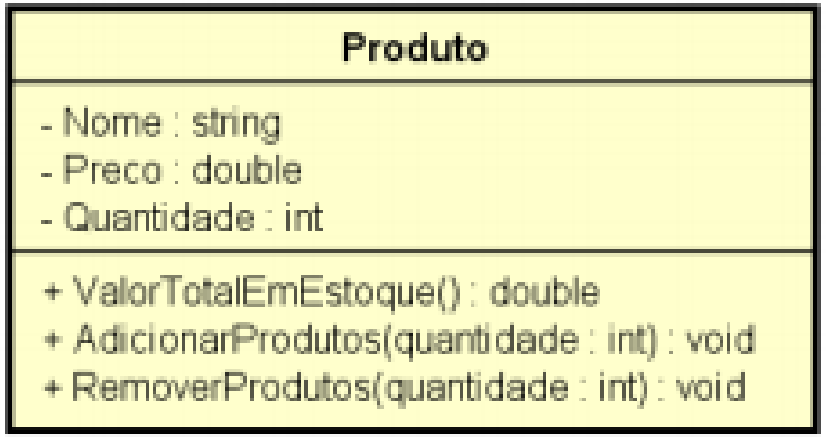
```
using System.Globalization;

namespace Course {
    class Produto {

        public string Nome;
        public double Preco;
        public int Quantidade;

        public double ValorTotalEmEstoque() {
            return Preco * Quantidade;
        }

        public override string ToString() {
            return Nome
                + ", $ "
                + Preco.ToString("F2", CultureInfo.InvariantCulture)
                + ", "
                + Quantidade
                + " unidades, Total: $ "
                + ValorTotalEmEstoque().ToString("F2", CultureInfo.InvariantCulture);
        }
    }
}
```

```
using System;
using System.Globalization;

namespace Course {
    class Program {
        static void Main(string[] args) {

            Produto p = new Produto();

            Console.WriteLine("Entre os dados do produto:");
            Console.Write("Nome: ");
            p.Nome = Console.ReadLine();
            Console.Write("Preço: ");
            p.Preco = double.Parse(Console.ReadLine(), CultureInfo.InvariantCulture);
            Console.Write("Quantidade no estoque: ");
            p.Quantidade = int.Parse(Console.ReadLine());

            Console.WriteLine();
            Console.WriteLine("Dados do produto: " + p);

            Console.WriteLine();
            Console.Write("Digite o número de produtos a ser adicionado ao estoque: ");
            int qte = int.Parse(Console.ReadLine());
            p.AdicionarProdutos(qte);

            Console.WriteLine();
            Console.WriteLine("Dados atualizados: " + p);

            Console.WriteLine();
            Console.Write("Digite o número de produtos a ser removido do estoque: ");
            qte = int.Parse(Console.ReadLine());
            p.RemoverProdutos(qte);

            Console.WriteLine();
            Console.WriteLine("Dados atualizados: " + p);
        }
    }
}
```

DS2
Orientação a Objetos

Fazer um programa para ler os dados de um produto em estoque (nome, preço e quantidade no estoque). Em seguida:

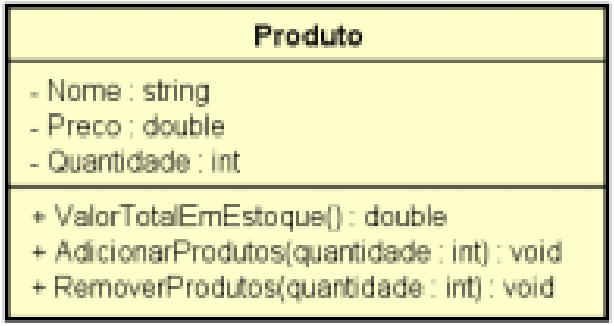
- Mostrar os dados do produto (nome, preço, quantidade no estoque, valor total no estoque)
- Realizar uma entrada no estoque e mostrar novamente os dados do produto
- Realizar uma saída no estoque e mostrar novamente os dados do produto



DS2
Orientação a Objetos

SERVIÇO NACIONAL DE APRENDIZAGEM INDUSTRIAL

```
Entre os dados do produto:
Nome: TV
Preço: 900.00
Quantidade no estoque: 10

Dados do produto: TV, $ 900.00, 10 unidades, Total: $ 9000.00

Digite o número de produtos a ser adicionado ao estoque: 5

Dados atualizados: TV, $ 900.00, 15 unidades, Total: $ 13500.00

Digite o número de produtos a ser removido do estoque: 3

Dados atualizados: TV, $ 900.00, 12 unidades, Total: $ 10800.00
```



DS2
Exercício

SERVIÇO NACIONAL DE APRENDIZAGEM INDUSTRIAL

EXERCÍCIO 01:
Fazer um programa para ler os dados de duas pessoas, depois mostrar o nome da pessoa mais velha.

Exemplo:

```
Dados da primeira pessoa:
Nome: Maria
Idade: 17
Dados da segunda pessoa:
Nome: Joao
Idade: 16
Pessoa mais velha: Maria
```

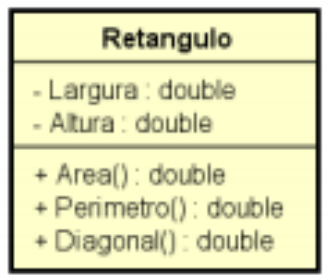
EXERCÍCIO 02:
Fazer um programa para ler nome e salário de dois funcionários. Depois, mostrar o salário médio dos funcionários.

Exemplo:

```
Dados do primeiro funcionário:
Nome: Carlos Silva
Salário: 6300.00
Dados do segundo funcionário:
Nome: Ana Marques
Salário: 6700.00
Salário médio = 6500.00
```

DS2
Exercício

Fazer um programa para ler os valores da largura e altura de um retângulo. Em seguida, mostrar na tela o valor de sua área, perímetro e diagonal. Usar uma classe como mostrado no projeto ao lado.



Exemplo:

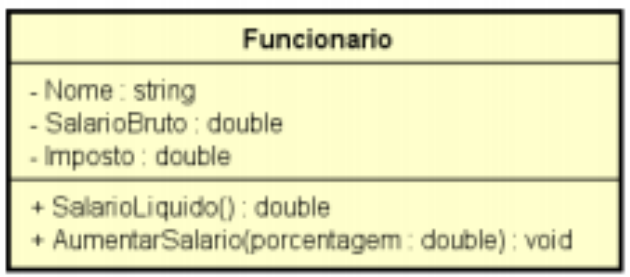
```
Entre a largura e altura do retângulo:
3.00
4.00
AREA = 12.00
PERÍMETRO = 14.00
DIAGONAL = 5.00
```

DS2
Exercício

Fazer um programa para ler os dados de um funcionário (nome, salário bruto e imposto). Em seguida, mostrar os dados do funcionário (nome e salário líquido). Em seguida, aumentar o salário do funcionário com base em uma porcentagem dada (somente o salário bruto é afetado pela porcentagem) e mostrar novamente os dados do funcionário. Use a classe projetada abaixo.

Exemplo:

```
Nome: Joao Silva
Salário bruto: 6000.00
Imposto: 1000.00

Funcionário: Joao Silva, $ 5000.00

Digite a porcentagem para aumentar o salário: 10.0

Dados atualizados: Joao Silva, $ 5600.00
```



DS2
Exercício

SERVIÇO NACIONAL DE
APRENDIZAGEM INDUSTRIAL

Fazer um programa para ler o nome de um aluno e as três notas que ele obteve nos três trimestres do ano (primeiro trimestre vale 30 e o segundo e terceiro valem 35 cada). Ao final, mostrar qual a nota final do aluno no ano. Dizer também se o aluno está APROVADO ou REPROVADO e, em caso negativo, quantos pontos faltam para o aluno obter o mínimo para ser aprovado (que é 60 pontos). Você deve criar uma classe **Aluno** para resolver este problema.

Exemplo 1:

```
Nome do aluno: Alex Green
Digite as três notas do aluno:
27.00
31.00
32.00
NOTA FINAL = 90.00
APROVADO
```

Exemplo 2:

```
Nome do aluno: Alex Green
Digite as três notas do aluno:
17.00
20.00
15.00
NOTA FINAL = 52.00
REPROVADO
FALTARAM 8.00 PONTOS
```