UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE TECNOLOGIA
DEPARTAMENTO DE PROCESSAMENTO DE ENERGIA ELÉTRICA

Instâncias de classes - Objetos

Curso de Engenharia de Controle e Automação

DPEE1090 - Programação orientada a objetos para automação

Prof. Renan Duarte

1º semestre de 2024

Sumário

Instâncias de classes - Objetos

- Compreensão dos conceitos de objetos e instância de classes
- Métodos padrão de classes
- Membros estáticos
- Exemplos práticos

Relembrando

Conceito de classe

- Uma classe é como uma planta baixa que especifica o que o tipo pode fazer
- Um objeto é basicamente um bloco de memória que foi alocado e configurado de acordo com a planta baixa
- Um programa pode criar muitos objetos da mesma classe
- Os objetos também são chamados de instâncias e podem ser armazenados em uma variável nomeada ou em um array ou coleção

Objetos

Definição

Objetos são instâncias concretas de uma classe

O sistema não aloca nenhum espaço de memória quando uma classe é especificada, mas sim quando ela é instanciada, ou seja, quando um objeto é criado.

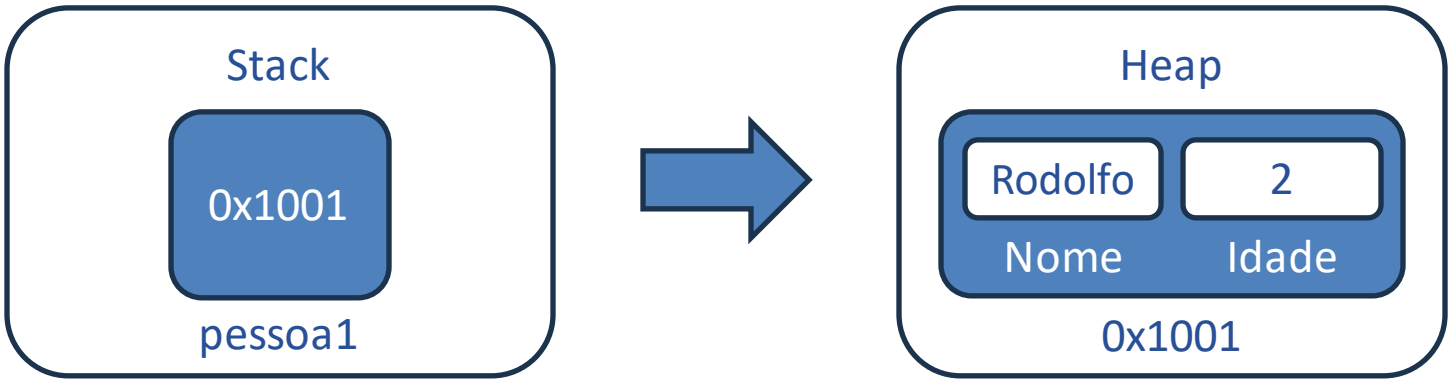
Um objeto possui estado (dados) e comportamento (código).

Objetos podem corresponder a coisas encontradas no mundo real. Portanto, por exemplo, um programa de gráficos terá objetos como círculo, quadrado, menu. Um sistema de compras online terá objetos como carrinho de compras, cliente, produto.

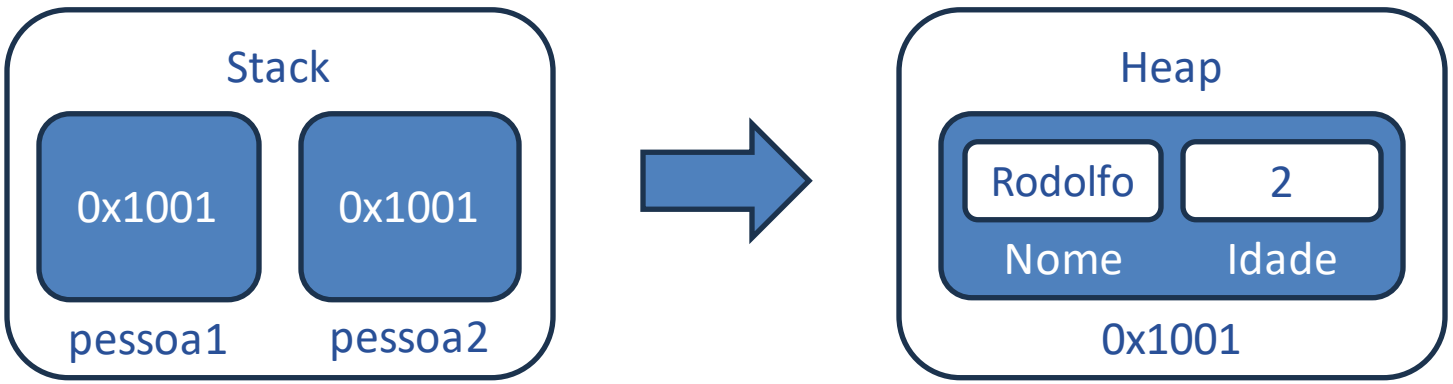
Objetos

Objetos são dados de tipo REFERÊNCIA

```
Pessoa pessoa1 = new Pessoa("Rodolfo", 2);
```




```
Pessoa pessoa2 = pessoa1;
```



Objetos

Objetos são dados de tipo REFERÊNCIA

```
// Declaração de um objeto Pessoa a partir da classe Pessoa
Pessoa pessoa1 = new Pessoa("Rodolfo", 2);

// Conteúdo de pessoa1
>> pessoa1: Nome = Rodolfo, Idade = 2

// Declaração de nova pessoa com dados de pessoa1
Pessoa pessoa2 = pessoa1;

// Mudança de dados de pessoa2
pessoa2.Nome = "Berenice";
pessoa2.Idade = 4;

// Conteúdo de pessoa2 e pessoa1
>> pessoa2: Nome = Berenice, Idade = 4
>> pessoa1: Nome = Berenice, Idade = 4
```

Objetos

Métodos padrão de uma classe

Independentemente do tipo de objeto que estamos modelando, é comum implementar alguns métodos padrão na classe para facilitar operações com os objetos criados a partir dela.

Exemplos:

- **ToString()**
- **Equals()**
- CompareTo()
- GetHashCode()

Esses métodos são herdados da classe pai Object e devem ser reescritos para uma implementação customizada

Objetos

ToString

Toda classe em C# herda implicitamente a classe Object. Portanto, todo objeto em C# recebe o método ToString:

```
Console.WriteLine(pessoa1.ToString());
Console.WriteLine(pessoa2.ToString());
```

```
>> Course.Pessoa
>> Course.Pessoa
```

Contudo, muitas vezes queremos definir como essa string é gerada para nosso objeto. Para isso, devemos sobreescrever o método ToString dentro de nossa classe.

Para sobreescrever um método herdado de uma classe pai, usamos a palavra *override*

ToString - Exemplo

Na classe pessoa:

```
public override string? ToString()
{
    return Nome + ", " + Idade;
}
```

No programa principal:

```
Console.WriteLine(pessoa1.ToString());
Console.WriteLine(pessoa2.ToString());
```

>> Rodolfo, 2

>> Berenice, 4

Objetos

Equals

Para verificar se dois objetos são iguais, primeiro devemos distinguir se desejamos saber se as duas variáveis representam o mesmo objeto na memória, ou se os valores de um ou mais de seus campos são equivalentes

```
Pessoa pessoa1 = new Pessoa("Rodolfo", 2);
Pessoa pessoa2 = pessoa1;
```

pessoa1 == pessoa2 → Verdadeiro

Essa comparação resulta em verdadeiro pois pessoa1 e pessoa2 na verdade apontam para o mesmo bloco de memória

Objetos

Equals

```
Pessoa pessoa1 = new Pessoa("Rodolfo", 2);
Pessoa pessoa2 = new Pessoa("Rodolfo", 2);
```

pessoa1 == pessoa2 → Falso

Essa comparação resulta em falso pois pessoa1 e pessoa2 representam blocos distintos de memória, apesar de possuírem os vamos valores nos atributos.

Para comparar se dois objetos do mesmo tipo possuem os mesmos valores nos atributos, devemos comparar os atributos um a um:

```
pessoa1.Nome == pessoa2.Nome && pessoa1.Idade == pessoa2.Idade;
```

Objetos

Equals

Essa comparação se torna complexa a medida que o número de campos em um objeto cresce. Para superar essa limitação, podemos sobreescrever na nossa classe Pessoa o método **Equals** que compara este objeto com outro e determina se são iguais.

Objetos

Equals - Exemplo

Na classe Pessoa:

```
public override bool Equals(object? obj)
{
    return obj is Pessoa pessoa &&
    Nome == pessoa.Nome &&
    Idade == pessoa.Idade;
}
```

No programa principal:

```
Pessoa pessoa1 = new Pessoa("Rodolfo", 2);
Pessoa pessoa2 = new Pessoa("Rodolfo ", 2);
```

pessoa1.Equals(pessoa2) → Verdadeiro

Objetos

Membros estáticos

Também chamados membros de classe

- Em oposição a membros de instância

São membros que fazem sentido independentemente de objetos

- Não precisam de objeto para serem chamados
- São chamados a partir do próprio nome da classe

Aplicações comuns:

- Classes utilitárias
- Declaração de constantes

Uma classe que possui somente membros estáticos, pode ser uma classe estática também. Esta classe não poderá ser instanciada.

Objetos

Membros estáticos – Exemplo ainda sem membros estáticos

Fazer um programa para ler um valor numérico qualquer e mostrar quanto seria o valor de uma circunferência e do volume de uma esfera para um raio do valor informado.

Informar também o valor de PI com duas casas decimais.

Exemplo:

Digite o valor do raio: **3.0**

Circunferência: 18.85

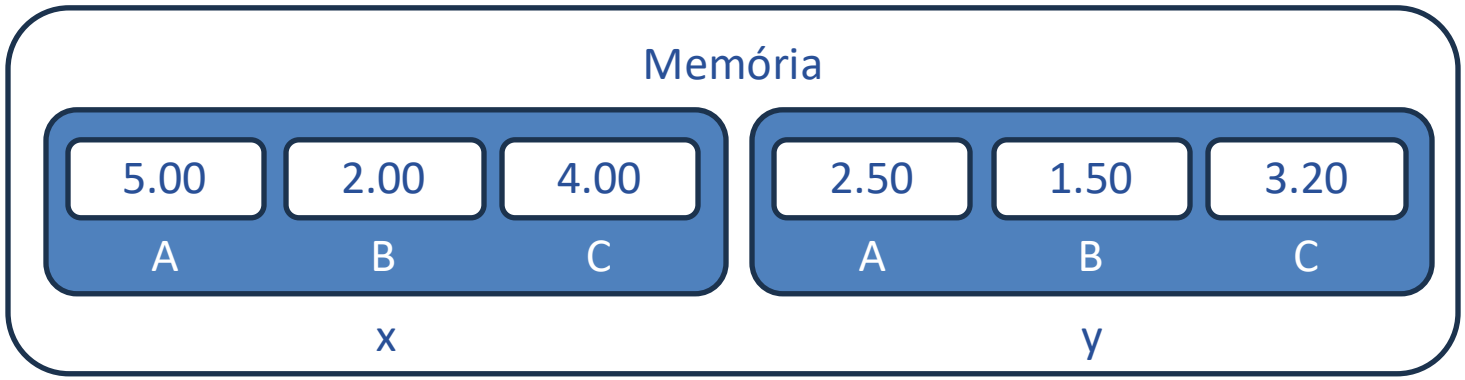
Volume: 113.09

Valor de PI: 3.14

Objetos

Membros estáticos

No exemplo dos triângulos da aula passada, a área de cada triângulo (calculada pelo método Triangulo.Area()) era uma operação concernente ao objeto Triangulo. Cada objeto possui a sua área:



x.Area() → 3.8

y.Area() → 1.824

Objetos

Membros estáticos

Já no caso da calculadora, os cálculos não mudam para objetos "calculadora" diferentesm ou seja são cálculos estáticos. O valor de Pi também é estático

```
Calculadora calc1 = new Calculadora();
Calculadora calc2 = new Calculadora();
```

calc1.Circunferencia(3.0) → 18.85

calc1.Pi → 3.1415

calc2.Circunferencia(3.0) → 18.85

calc2.Pi → 3.1415

Membros estáticos – Exemplo com membros estáticos

Nesse caso, podemos modificar o exemplo utilizando membros estáticos para otimizar o uso de recursos do nosso código:

```
// Valor de Pi
public static double Pi = 3.1415;

// Método para calcular a circunferência dado um raio
public static double Circunferencia(double raio)
{
    return 2.0*Pi*raio;
}

// Método para calcular o volume de uma esfera dado um raio
public static double Volume(double raio)
{
    return (4.0/3.0) * Pi * Math.Pow(raio, 3);
}
```

Objetos

https://bit.ly/49mw7AI

Exercício 1

Implemente uma classe que modela um produto. Os atributos são: Nome, Código, Preço e Quantidade em estoque. Implemente métodos para imprimir o objeto na forma de string, para calcular o valor total em estoque e também para comparar se dois produtos são iguais (a regra para igualdade é possuírem o mesmo nome e código)

Exemplos nos próximos slides

Exercício 1 - Continuação

Saída esperada para produtos diferentes:

Digite os dados do produto 1:
Nome: TV
Código: 123
Valor: 1000.00
Quantidade: 5

Digite os dados do produto 2:
Nome: Xbox
Código: 554
Valor: 1200
Quantidade: 10

Nome: TV, Cod: 123, Valor: 1000.00, Qtd: 5, Total em estoque: 5000.00
Nome: Xbox, Cod: 554, Valor: 1200.00, Qtd: 10, Total em estoque: 12000.00
Produtos nao sao iguais

Exercício 1 - Continuação

Saída esperada para produtos iguais:

Digite os dados do produto 1:
Nome: TV
Código: 123
Valor: 1000.00
Quantidade: 5

Digite os dados do produto 2:
Nome: TV
Código: 123
Valor: 1200
Quantidade: 10

Nome: TV, Cod: 123, Valor: 1000.00, Qtd: 5, Total em estoque: 5000.00
Nome: TV, Cod: 123, Valor: 1200.00, Qtd: 10, Total em estoque: 12000.00
Produtos sao iguais

Exercício 2 – Membros estáticos

Faça um programa para ler a cotação do dólar, e depois um valor em dólares a ser comprado por uma pessoa em reais. Informar quantos reais a pessoa vai pagar pelos dólares, considerando ainda que a pessoa terá que pagar 6% de IOF sobre o valor em dólar. Criar uma classe **ConversorDeMoeda** para ser responsável pelos cálculos.

Exemplo:

Digite a cotação do dólar: **4.90**

Digite a quantidade de dólares a ser comprada: **100.00**

Valor a ser pago em reais: 519.40

Revisão

Próxima aula

Abstração e encapsulamento

- Conceito de abstração e sua aplicação em classes
- Conceito de Interface e implementação
- Exploração do conceito de encapsulamento
- Utilização de modificadores de acesso em atributos e métodos
- Implementação de *getters, setters* e propriedades

Dúvidas?

renan.duarte@gedre.ufsm.br

GEDRE – Prédio 10 – CTLAB