

Unidade 0 - Nivelamento - Introdução a Orientação por Objetos



PUC Minas

Instituto de Ciências Exatas e Informática
Departamento de Ciência da Computação

- Reaproveitamento de código (produtividade)
- Fazer resumo das seções 3.1 à 3.7 e 8.1 à 8.7

Definições

- Classe é um tipo, um conjunto de regras
- Objeto é uma variável do tipo classe

Método Construtor

- Método para inicializar os atributos de uma classe
- Apresenta o mesmo nome da classe
- Não apresenta tipo de retorno
- Sempre será do tipo público

Primeira Classe

- Ver código em:

[./primeiraClasse/](#)

Exercício

- Seja a classe Lixão abaixo e a Funcionário do exemplo anterior, faça o quadro de memória para o código abaixo

```
class Lixao{  
    public static void main(String[] args){  
        Funcionario f1;  
        f1 = new Funcionario();  
        f1 = new Funcionario(5);  
    }  
}
```

Exercício

- Faça uma classe retângulo contendo os atributos base e altura, pelo menos dois métodos construtores, o método `double getArea()` que retorna a área do retângulo, o método `double getPerimetro()` que retorna o perímetro do retângulo e o método `double getDiagonal()` que retorna a diagonal do retângulo
- Faça também uma classe Lixao contendo um método `main` sendo que esse terá dois objetos do tipo retângulo e chamará os métodos desenvolvidos na classe retângulo

Visibilidade

- *public* : acessíveis “dentro” e “fora” da classe
- *private* : acessíveis somente “dentro” da classe
- **Qual é a vantagem disso?**

Visibilidade

- Ver código em:

[./visibilidade/](#)

Métodos *get* e *set*

- Normalmente, temos um *get* e um *set* para cada atributo privado
- O método ***get****NomeAtributo* retorna o conteúdo do atributo privado
- O método ***set****NomeAtributo* altera o conteúdo do atributo privado

Métodos `get` e `set`

- O tipo de retorno do método `getNomeAtributo` é o mesmo do seu atributo e esse método não recebe parâmetros
- O método `setNomeAtributo` retorna `void` e ele possui um parâmetro de entrada cujo tipo é o mesmo do atributo

- Atributos e métodos estáticos podem ser chamados sem que um objeto seja instanciado (por exemplo, *IO.println*)
- **Qual é a vantagem disso?**

- O valor dos atributos estáticos é compartilhado por todos os objetos da classe
- Os métodos estáticos acessam somente os métodos ou atributos estáticos da classe

- Ver código em:

[./estatico/](#)

Operador *this*

- Explicita que um método ou atributo pertence ao objeto corrente
- Ele também pode ser utilizado para explicitar o objeto corrente

Exemplo: Classe Retângulo

Ver código em: `LixaoRetangulo.java` e
`Retangulo.java`

Exemplo: Classe Matriz

Ver código em: `Matriz.java`

- Um aluno desenvolveu a classe abaixo e pediu sua ajuda para compilá-la. Para ajudar, você deve criar uma classe Ponto com as seguintes regras:
 - Quatro atributos privados: double x, double y, int id e int nextID
 - Os atributos id e nextID serão alterados somente por um método construtor.
 - Implemente os métodos get e set tanto para o atributo x como para o y
 - Na declaração do atributo nextID, o mesmo deve receber zero. Além disso, a alteração do valor desse atributo por um objeto sempre será compartilhada com qualquer objeto da classe ponto
 - Implemente dois construtores sendo que o primeiro não recebe parâmetros e inicializa os valores de x e y com zero. O segundo recebe dois parâmetros (cujos nomes são obrigatoriamente x e y) e devem ser utilizados para inicializar os valores dos atributos x e y, respectivamente
 - Os dois construtores devem atribuir o valor corrente do atributo nextID ao atributo id e incrementar o valor de nextID. Observe que cada objeto terá um ID distinto
 - Implemente qualquer método que seja necessário para compilar a classe LixaoPonto

```
class LixaoPonto {  
    public static void main (String[] args){  
        Ponto p1 = new Ponto(4,3);  
        Ponto p2 = new Ponto(8,5);  
        Ponto p3 = new Ponto(9.2,10);  
  
        System.out.println("Distancia p1 entre e p2: " + p1.dist(p2));  
        System.out.println("Distancia p1 entre e (5,2): " + p1.dist(5,2));  
        System.out.println("Distancia (4,3) entre e (5,2): " + Ponto.dist(4,3,5,2));  
        System.out.println("P1, P2, P3 sao triangulo:" + Ponto.isTriangulo(p1,p2,p3));  
        System.out.println("Area retangulo:" + p1.getAreaRetangulo(p2));  
        System.out.println("ID de P1: " + p1.getID());  
        System.out.println("ID de P2: " + p2.getID());  
        System.out.println("ID de P3: " + p3.getID());  
        System.out.println("Next ID: " + Ponto.getNextID());  
    }  
}
```

