Java: Classe, Objeto e Método

1. Programação Orientada a Objetos

O paradigma orientado a objetos é utilizado extensivamente no processo de desenvolvimento de software. Ele surgiu como uma evolução do paradigma estruturado e possui como foco a representação de entidades do mundo real como objetos, que interagem entre si para alcançar um determinado objetivo ou solucionar um problema.

Um programa orientado a objetos é constituído por objetos, que são instâncias de classes. Cada objeto possui uma funcionalidade específica, visível ao usuário e cuja implementação fica escondida. Objetos interagem entre si para solucionar problemas complexos. Eles facilitam a busca por *bugs*, a modularidade, reutilização, manutenção e extensão do código.

Paradigma: é um conjunto de conceitos, práticas e metodologias que definem uma forma de se pensar ou trabalhar numa determinada área do conhecimento. Em programação, um paradigma estabelece regras e padrões sobre como o código deve ser organizado para que um determinado problema seja solucionado.

1.1. Definição de Classe e Objeto

Classe: É um *blueprint* dos objetos que define como eles devem funcionar. A classe representa um conjunto de objetos com características iguais/similares. Ela define o comportamento de um objeto, por meio de métodos, e quais estados ele é capaz de manter, através de atributos. Exemplos de Classes: Pessoa, Carro, Casa, Aluno, Conta Corrente, Filme, Jogo, etc.

Veja o código abaixo:

```
class Pessoa {
    String nome;
    int idade;
}
```

A classe Pessoa foi definida. Note que ela não é um objeto, mas é utilizada para construir um objeto. Neste momento a classe contém dois atributos: nome e idade. Você poderia perguntar: Mas de qual pessoa estamos falando? E a resposta seria: Nenhuma. Temos somente a definição de uma classe, de forma genérica. É preciso criar objetos para que possamos especificar uma determinada pessoa.

Objeto: É uma instância de uma classe. Um objeto é capaz de armazenar estados através de seus atributos e reagir a mensagens enviadas a ele, assim como se relacionar e enviar mensagens a outros objetos. Exemplo de objetos da classe Pessoa: João, José, Maria, Pedro, etc.

Como criar um objeto do tipo Pessoa? Veja o código abaixo:

```
Pessoa p = new Pessoa();
```

Agora temos um objeto “p”, do tipo Pessoa. Dessa maneira, o objeto “p” possui os atributos nome e idade. O operador “.” possibilita o acesso aos atributos.

```
p.nome = "José";
p.idade = 25;
```

Surge então outra questão. Como definir o comportamento de uma pessoa? Isso é realizado utilizando-se os métodos.

1.2. Definição de Método

Um método é muito parecido com uma função. Ele possui um nome e um corpo.

- O nome deve ser um identificador que reflita o propósito principal do método (ex: efetuarCalculoImposto).
- O corpo do método possui as declarações que serão executadas quando o método for invocado/chamado. É possível enviar dados para o processamento e receber o resultado de tal operação.

1.2.1. Sintaxe:

```
tipoRetorno nomeMetodo (listaParametros) {
  // corpo do método
}
```

- tipoRetorno: especifica o tipo de informação que o método retorna. Pode ser int, float, String, etc. Se você estiver escrevendo um método que não retorna um valor, utilize **void** como tipo de retorno.
- nomeMetodo: o nome utilizado para invocar o método. O nome de um método segue as mesmas regras utilizadas para a criação de variáveis.
- listaParametros: é opcional e descreve os tipos e nomes das informações que você quer enviar para processamento. Você escreve os parâmetros entre os parênteses da mesma forma como declara as variáveis. Caso seu método tenha dois ou mais parâmetros, separe-os utilizando vírgula.

Exemplo 1: Criando métodos para a classe Pessoa

```
class Pessoa {
  String nome;
  int idade;

  void conversar() {
    System.out.println("Blá Blá Blá");
  }

  void comprar() {
    System.out.println("Hora de Gastar $$$$");
  }

  int informarIdade() {
    return idade;
  }
}
```

Como Testar?

```
public class TestePessoa {
  public static void main(String[] args) {
    Pessoa p = new Pessoa();
    p.nome = "José";
    p.idade = 25;
    p.conversar();
    p.comprar();
    System.out.println("Idade da Pessoa:" + p.informarIdade());
  }
}
```

Exemplo 2:

```
int somaValores(int valor1, int valor2) {
  // corpo do método
}

somaValores;              // erro em tempo de compilação.
somaValores ();           // erro em tempo de compilação.
```

```
somaValores (37);          // erro em tempo de compilação.
somaValores ("37", "5");   // erro em tempo de compilação.
```

1.2.2. Escrevendo declarações de retorno do método

Para retornar um valor do método, escreva a declaração return dentro do método, seguida pela expressão a qual você quer retornar, seguida de “;”.

```
int somaValores(int valor1, int valor2) {
  // corpo do método
  return valor1 + valor2;
}
```

1.2.3. Definindo Escopo Local

```
public class Exemplo {
  void primeiroMetodo() {
    int campo; // ok
  }

  void segundoMetodo() {
    campo = 42; // errado
  }
}
```

1.2.4. Definindo Escopo de Classe

```
class Exemplo {
  public int campo = 0;

  public void primeiroMetodo() {
    campo = 42; // ok
  }

  public void segundoMetodo() {
    campo++; // ok
  }
}
```

1.2.5. Efetuando a Chamada de um método

```
nomeObjeto.metodo();

class TesteFuncionario {
  public static void main(String[] args) {
    Funcionario f = new Funcionario();
    f.nome = "Edilson";
    f.salarioBruto = 1000;
    System.out.println(f.calcularSalarioLiquido());
  }
}

class Funcionario {
  String nome;
  double salarioBruto;

  public double calcularSalarioLiquido() {
    return salarioBruto * 0.9;
  }
}
```

2. Exercícios Propostos

2.1. Escreva um programa que auxilie no controle da produção de cana de açúcar de uma usina. O programa deve armazenar a quantidade de cana de açúcar estimada para processamento (em toneladas), a quantidade que foi realmente processada na usina, o custo de produção e o preço de venda da tonelada. Após isso, um relatório deve ser elaborado, contendo as seguintes informações:

a) Quantidade de toneladas processadas.
b) Valor da tonelada.
c) Lucro da usina.
d) Capacidade de produção utilizada (processamento real dividido pelo processamento estimado).

2.2. Crie uma classe que permita a simulação de uma conta bancária. Ela deve conter o nome do cliente, o tipo da conta (1-Especial; 2-Comum) e o saldo. As seguintes funcionalidades devem ser fornecidas: a) saque; b) depósito; c) bônus; d) cálculo da tarifa. O bônus e a tarifa devem ser calculados de acordo com as tabelas abaixo:
Detalhe: o bônus pode ser utilizado somente uma vez.

Saldo	Bônus
Até 1000	5%
Acima de 1000	10%

Tipo Conta	Tarifa
1	R$ 25
2	R$ 90

2.3. Crie um programa que gerencie um estacionamento rotativo de carros. O programa deve armazenar a descrição do carro, a placa, o horário de entrada e o horário de saída (despreze os minutos). O estacionamento cobra X reais pela primeira hora de permanência com o automóvel e X/3 pelas demais horas. Além disso, é fornecido um desconto para o pagamento de acordo com a tabela abaixo:

Valor	Desconto (%)
Até R$ 20 (inclusive)	5
Entre R$ 20 e R$ 50 (inclusive)	10
Acima de R$ 50	20

O programa deve exibir um relatório contendo as seguintes informações:

a) Tipo do carro.
b) Placa.
c) Hora da Entrada.
d) Hora da Saída.
e) Valor Pago.

3. Exercícios Complementares

3.1. Crie uma classe chamada Calçado. A classe deve conter os atributos descricao, tipo, anoFabricacao e valor. Além disso, forneça métodos para informar/recuperar a descrição do calçado, o tipo, o ano no qual foi lançado e também seu preço.

3.2. Especifique uma classe que represente uma Casa. A classe deve permitir a verificação da quantidade de pessoas em seu interior, se as luzes externas estão acesas ou não, a quantidade de água e energia elétrica consumidos, assim como a temperatura ambiente.

3.3. Crie uma classe Aluno especificando o nome, o sobrenome, prontuário, média geral das disciplinas. Inclua métodos para identificação do nome do aluno e também alteração/exibição das demais informações.

3.4. Crie uma classe Livro especificando o título, o autor, a editora e o número de páginas. Inclua métodos para retornar o título, a editora e para determinar se o livro está emprestado ou não.
Dica: Como vou saber se o livro está emprestado? A utilização de um atributo resolveria esse problema?

3.5. Crie uma classe que represente uma Calculadora simples. Essa classe terá os métodos de adição, subtração, divisão e multiplicação. Os métodos devem receber dois parâmetros (Ex: valor1 e valor2) e retornar o resultado.

3.6. Escreva um programa (utilizando classes e métodos) para calcular o aumento salarial de um funcionário baseando-se na tabela abaixo. Além do salário, a classe deve conter o nome do funcionário e o ano de contratação.

Salário	**Aumento (%)**
Acima de R$ 4.000,00	5%
R$ 1.900,00 ○-----● R$ 4.000,00	10%
Até R$ 1.900,00	20%

3.7. Escreva um programa para gerenciar duas rotas de uma companhia aérea (São Paulo – Rio e São Paulo – Salvador). Para cada uma das rotas, devem ser armazenados a quantidade de assentos disponíveis e ocupados, a tarifa e o tipo de aeronave utilizada. Após a leitura das informações, o programa deve produzir um relatório contendo os seguintes itens:

a) Tipo da Aeronave.
b) Origem.
c) Destino.
d) Percentual de ocupação da aeronave.
e) Valor da tarifa.
f) Faturamento do voo.