Programação Orientada a Objetos
SANTOS, Rafael

Capítulo 1 – Introdução a Orientação a Objetos

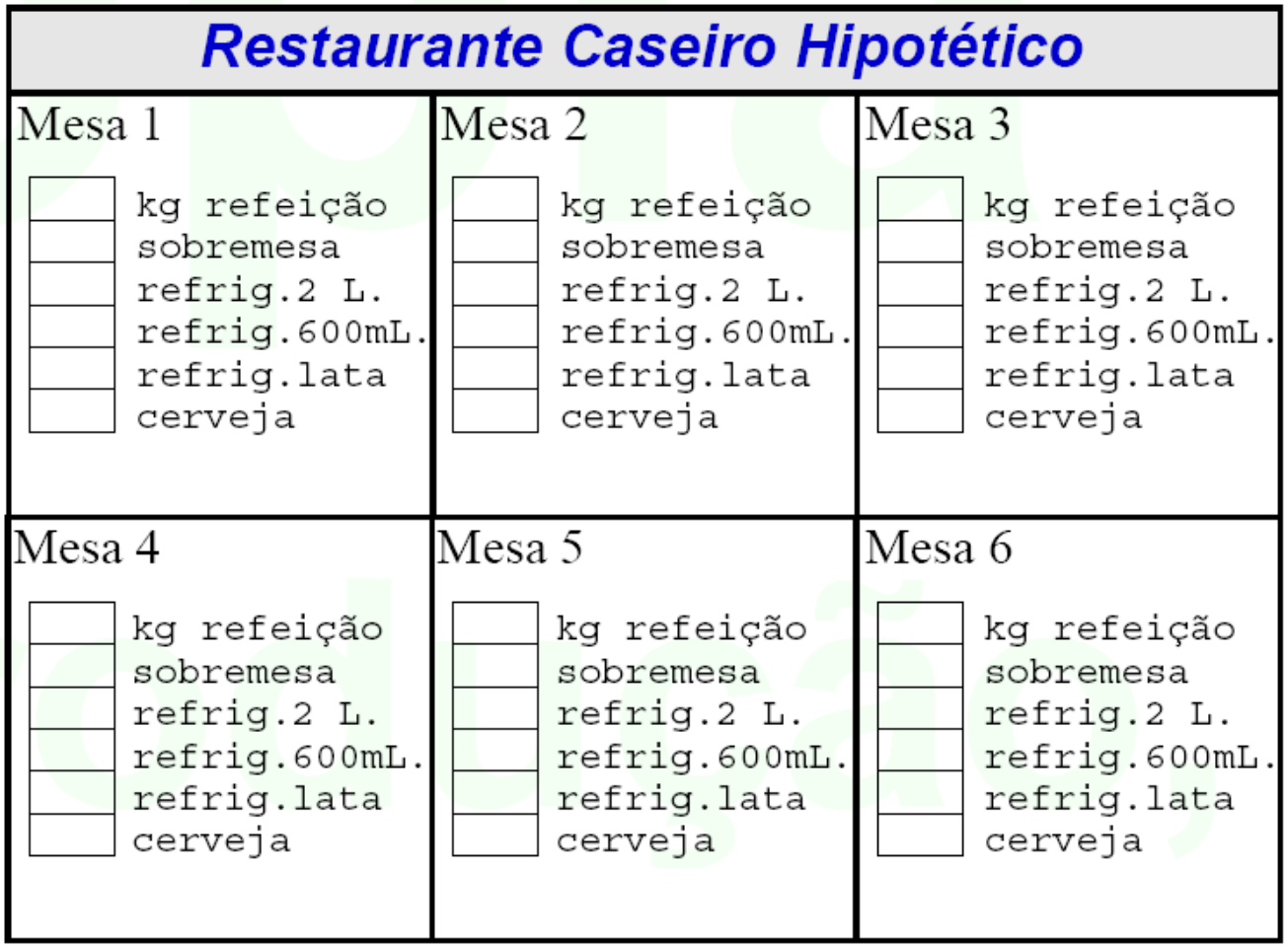
Programa de Computador

- É parte do **software**, e deve atender os requisitos do usuário
- **Controla o hardware**, incluindo periféricos de entrada e saída
- Usa um conjunto de comandos e regras: uma **linguagem de programação**
- Código ou código-fonte é **compilado***
- **Processam** (operam sobre) **dados**
- Na OO, **dados** e **operações** são considerados em conjunto, em um **modelo**

Modelos

- **Representações** simplificadas de objetos que fazem parte do **negócio** alvo do projeto
- Os **dados** e **operações** de um modelo são aqueles **relevantes** ao estudo e a semântica (uma pessoa pode ser empregado, paciente, contato comercial, etc.)
- Podem **conter** ou ser **derivados** de outros modelos

Modelos


Restaurante Caseiro Hipotético
Mesa 1
kg refeição
sobremesa
refrig.2 L.
refrig.600mL.
refrig.lata
cerveja
Mesa 2
kg refeição
sobremesa
refrig.2 L.
refrig.600mL.
refrig.lata
cerveja
Mesa 3
kg refeição
sobremesa
refrig.2 L.
refrig.600mL.
refrig.lata
cerveja
Mesa 4
kg refeição
sobremesa
refrig.2 L.
refrig.600mL.
refrig.lata
cerveja
Mesa 5
kg refeição
sobremesa
refrig.2 L.
refrig.600mL.
refrig.lata
cerveja
Mesa 6
kg refeição
sobremesa
refrig.2 L.
refrig.600mL.
refrig.lata
cerveja

Orientação a Objetos

- **Paradigma** de desenvolvimento de software (análise, projeto e programação)
- Usa **classes** (**modelos**) e **objetos** criados a partir dessas **classes**
- A classe é um **tipo** com **dados** e **operações**
- A **modelagem** (criação de modelos, classes) deve buscar principalmente: **coesão**, facilidade de manutenção (**baixo acoplamento**) e **eficiência**

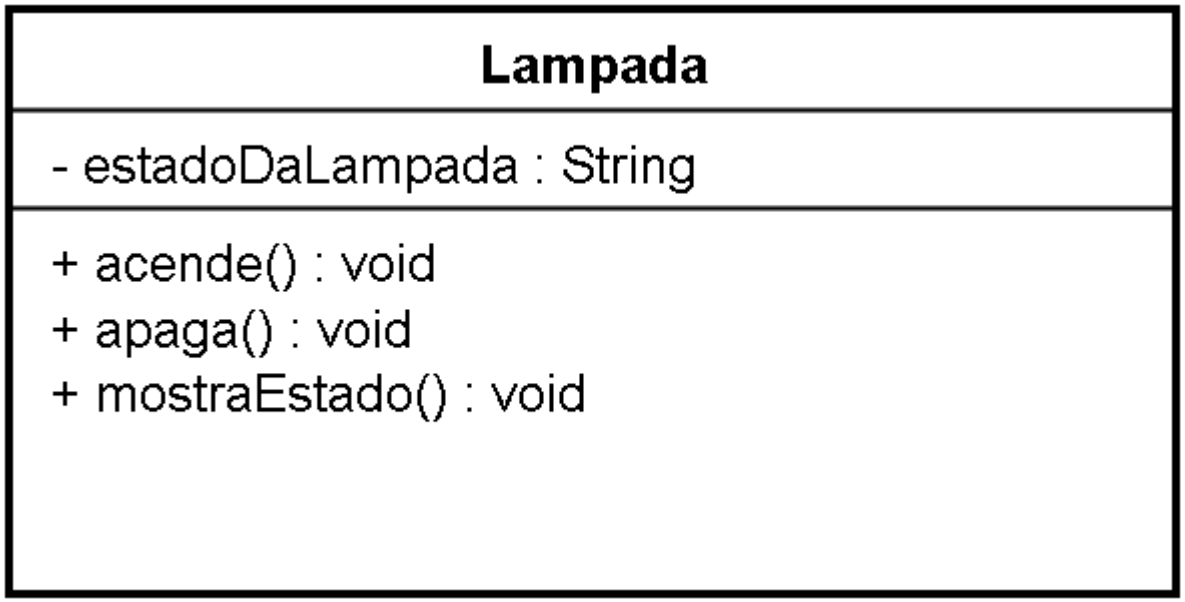
Encapsulamento

- Os dados de um modelo **não** devem ser **acessados diretamente** (visibilidade restrita)
- Para manipulação de seus dados o modelo deve oferecer **operações** específicas
- É um **benefício** dos mais palpáveis e **um dos principais objetivos** da POO
- Melhora a **clareza** e a **organização** e **reduz** a quantidade de **erros**

Exemplos de Modelos

- Uma lâmpada
- Uma conta bancária simplificada
- Uma data
- Um registro acadêmico de aluno

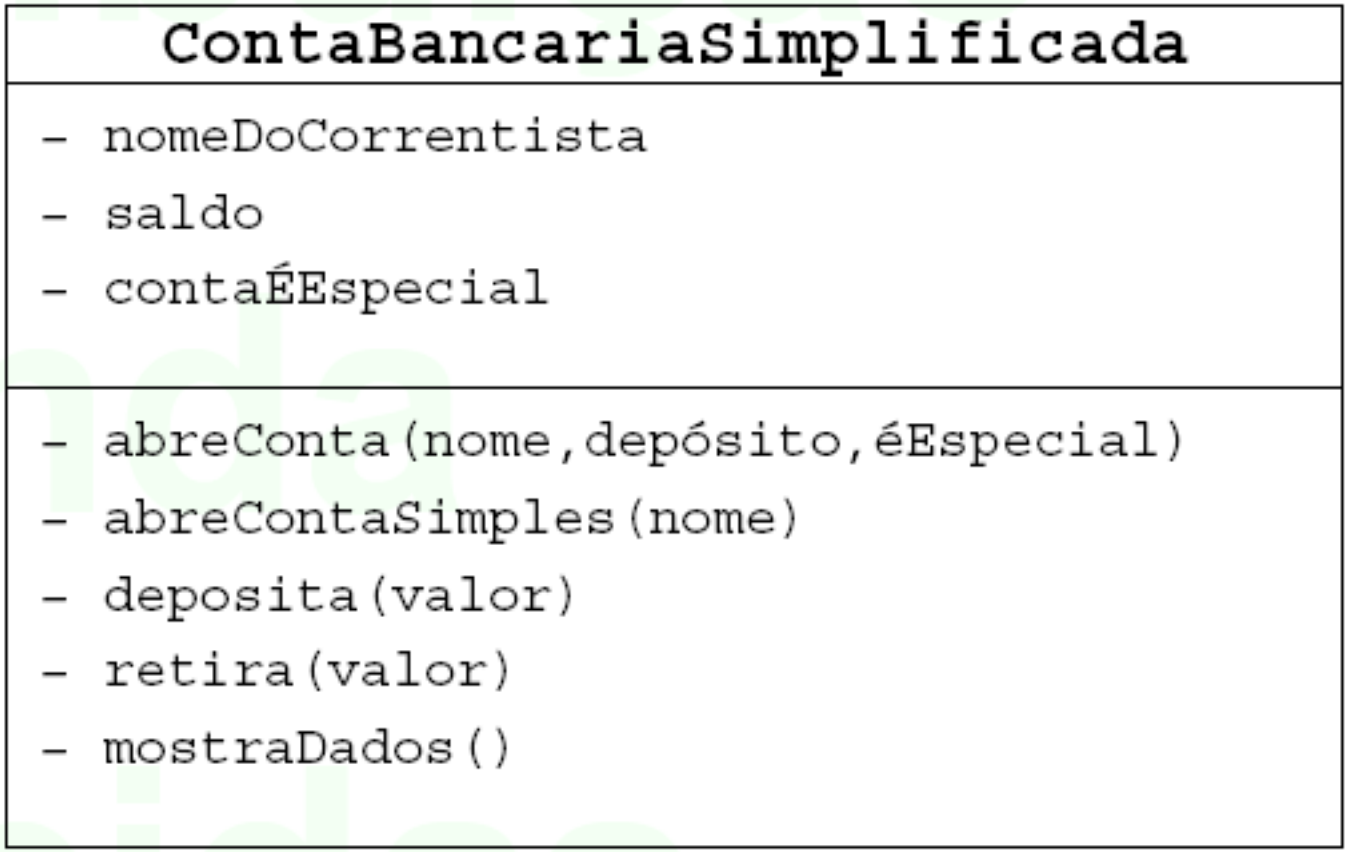
Modelo da Lâmpada
Representação da UML



Modelo da Lâmpada
Pseudocódigo

```
modelo Lampada // representa uma lâmpada em uso
início do modelo

      dado estadoDaLâmpada; // indica se está ligada ou não

      operação acende() // acende a lâmpada
      início
            estadoDaLâmpada = aceso;
      fim

      operação apaga() // apaga a lâmpada
      início
            estadoDaLâmpada = apagado;
      fim

      operação mostraEstado() // mostra o estado da lâmpada
      início
            se (estadoDaLâmpada == aceso)
                  imprime "A lâmpada está acesa";
            senão
                  imprime "A lâmpada está apagada";
      fim

fim do modelo
```

Modelo da Conta Bancária Representação da UML



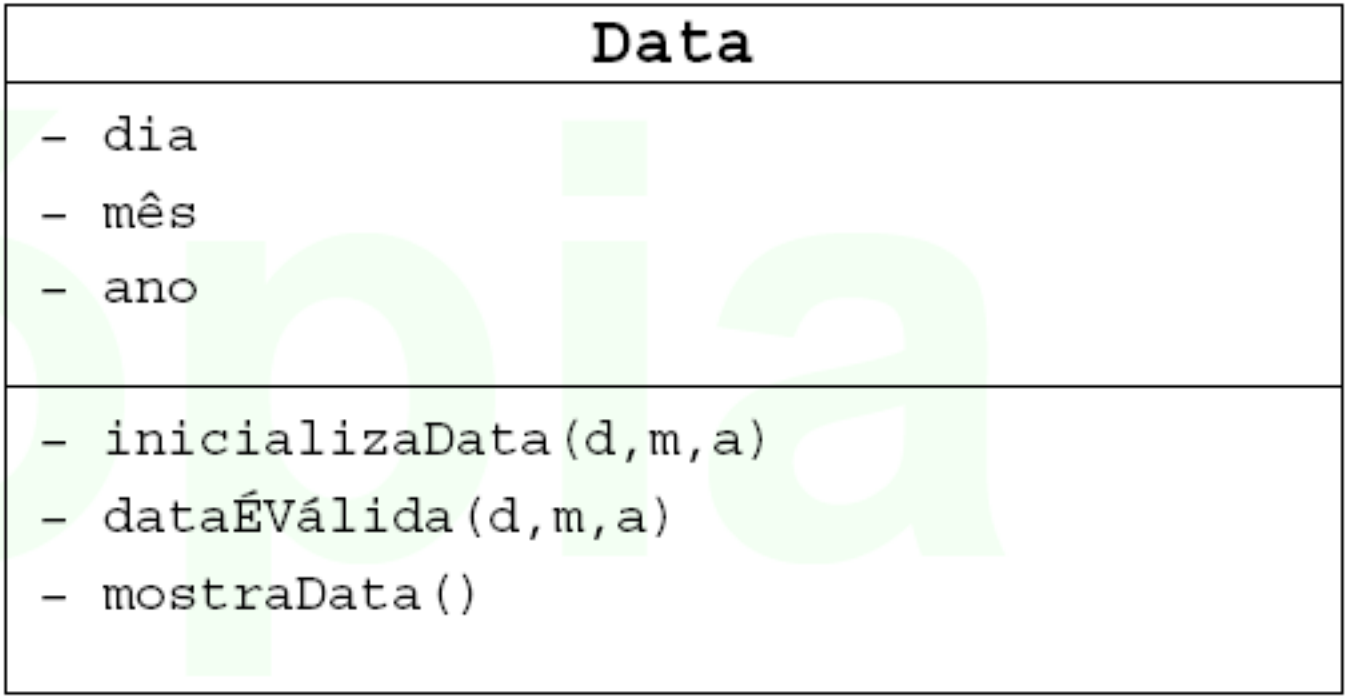
Modelo da Conta Bancária Pseudocódigo

```
modelo ContaBancariaSimplificada
início do modelo

        dado nomeDoCorrentista,saldo,contaÉEspecial; // dados da conta

        // Inicializa simultaneamente todos os dados do modelo
        operação abreConta(nome,depósito,especial) // argumentos para esta operação
        início
                // Usa os argumentos passados para inicializar os dados do modelo
                nomeDoCorrentista = nome;
                saldo = depósito;
                contaÉEspecial = especial;
        fim

        // Inicializa simultaneamente todos os dados do modelo, usando o nome
        // passado como argumento e os outros valores com valores default
        operação abreContaSimples(nome) // argumento para esta operação
        início
                nomeDoCorrentista = nome;
                saldo = 0.00;
                contaÉEspecial = falso;
        fim
```

Modelo da Conta Bancária Pseudocódigo

```
        operação deposita(valor) // Deposita um valor na conta
        início
                saldo = saldo + valor;
        fim

        operação retira(valor) // Retira um valor da conta
        início
                se (contaÉEspecial == falso) // A conta não é especial !
                        se (valor <= saldo) // se existe saldo suficiente...
                                saldo = saldo - valor; // faz a retirada.
                senão // A conta é especial, pode retirar à vontade !
                        saldo = saldo - valor;
        fim

        operação mostraDados() // mostra os dados da conta, imprimindo os seus valores
        início
                imprime "O nome do correntista é ";
                imprime nomeDoCorrentista;
                imprime "O saldo é ";
                imprime saldo;
                se (contaÉEspecial) imprime "A conta é especial.";
                senão imprime "A conta é comum.";
        fim

fim do modelo
```

Modelo da Data
Representação da UML



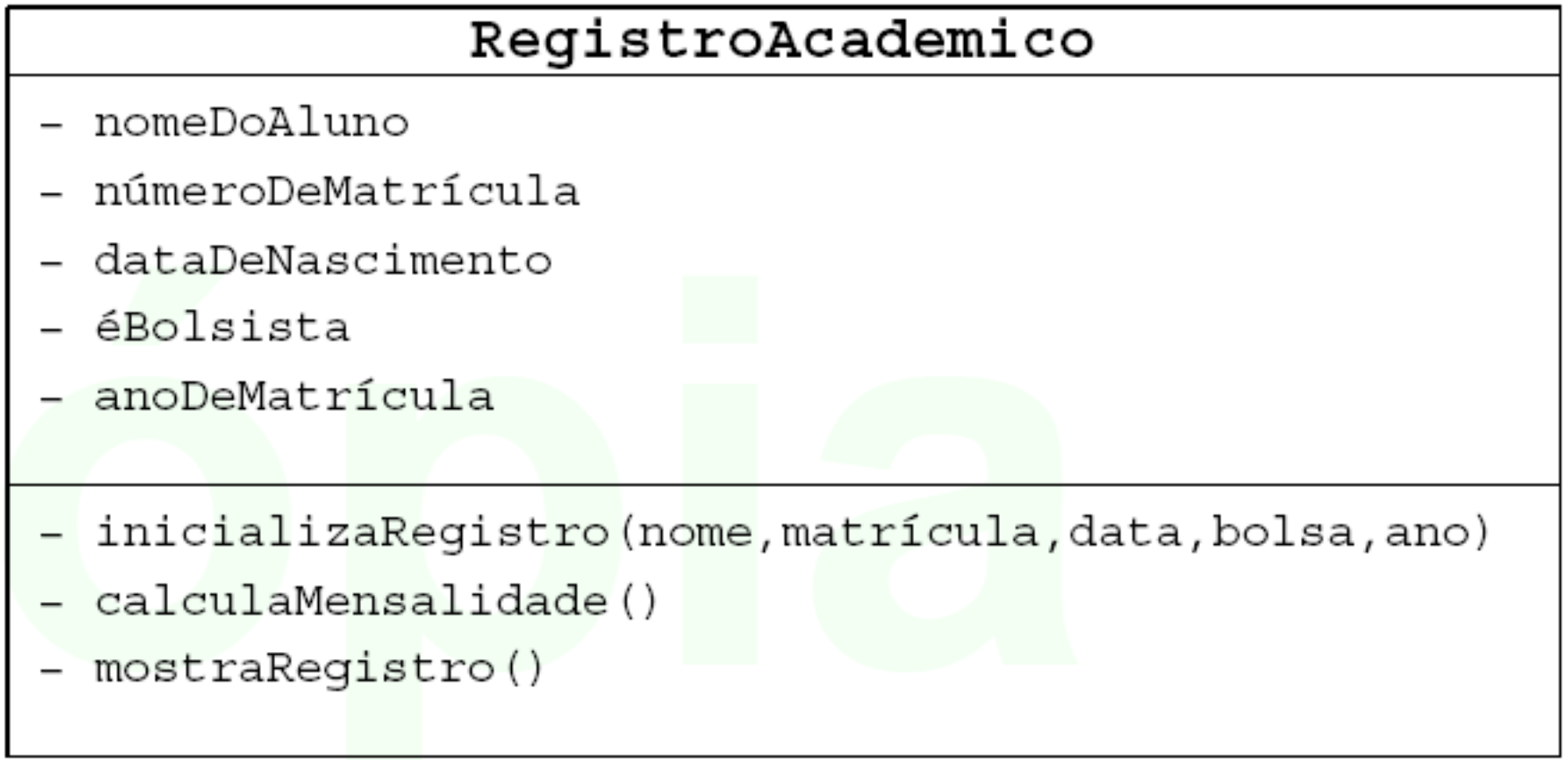
Modelo da Data
Pseudocódigo

```
modelo Data
início do modelo

    dado dia,mês,ano; // componentes da data

    // Inicializa simultaneamente todos os dados do modelo
    operação inicializaData(umDia,umMês,umAno) // argumentos para esta operação
    início
        // Somente muda os valores do dia, mês e ano se a data passada for válida
        se dataÉVálida(umDia,umMês,umAno) // Repassa os argumentos para a operação
        início
            dia = umDia;
            mês = umMês;
            ano = umAno;
        fim
        // Se a data passada não for válida, considera os valores sendo zero
        senão
        início
            dia = 0;
            mês = 0;
            ano = 0;
        fim
fim
```

Modelo da Data
Pseudocódigo

```
operação dataÉVálida(umDia,umMês,umAno) // argumentos para esta operação
início
        // Se a data passada for válida, retorna verdadeiro
        se ((dia >= 1) e (dia <= 31) e (mês >= 1) e (mês <= 12))
                retorna verdadeiro;
        // Senão, retorna falso
        senão
                retorna falso;
fim

operação mostraData() // mostra a data imprimindo valores de seus dados
início
        imprime dia;
        imprime "/";
        imprime mês;
        imprime "/";
        imprime ano;
fim

fim do modelo
```

Modelo do Registro Acadêmico
Representação da UML



Modelo do Registro Acadêmico Pseudocódigo

```
modelo RegistroAcademico
início do modelo
        // Dados do registro acadêmico
        dado nomeDoAluno,númeroDeMatrícula;
        dado dataDeNascimento,éBolsista,anoDeMatrícula;

        // Inicializa simultaneamente todos os dados do modelo, passando argumentos
        operação inicializaRegistro(oNome,aMatrícula,aData,temBolsa,qualAno)
        início
                // Usa os argumentos para inicializar os valores no modelo
                nomeDoAluno = oNome;
                númeroDeMatrícula = aMatrícula;
                dataDeNascimento = aData;
                éBolsista = temBolsa;
                anoDeMatrícula = qualAno;
        fim
```

Modelo do Registro Acadêmico Pseudocódigo

```
operação calculaMensalidade() // calcula e retorna a mensalidade
início
        mensalidade = 400;
        se (éBolsista) mensalidade = mensalidade / 2;
        retorna mensalidade;
fim

operação mostraRegistro() // mostra os dados do registro acadêmico
início
        imprime "Nome do aluno:";
        imprime nomeDoAluno;
        imprime "Número de Matrícula:";
        imprime númeroDeMatrícula;
        imprime "Data de Nascimento:";
        dataDeNascimento.mostraData(); // pede à data que se imprima !

        se (éBolsista == verdadeiro) imprime "O aluno é bolsista.";
        senão imprime "O aluno não é bolsista.";

        imprime "Ano de Matrícula:";
        imprime anoDeMatrícula;
fim

fim do modelo
```

Síntese

- **Classe** é uma estrutura da OO para implementar um determinado **modelo**
- **Objeto** é a materialização (concretização) de uma **classe**
- **Instância** é o mesmo que **objeto**
- **Campos** (ou atributos ou propriedades) são os dados definidos na classe, implementados por meio de **variáveis**
- **Métodos** são as **operações** de uma **classe**

Exercício 1

- Identifique objetos significativos dentro do universo de suas atividades acadêmicas e/ou profissionais e crie modelos, usando pseudocódigo, contendo dados e operações importantes para esses objetos

Exercício 2

- Crie um modelo em pseudo-código, com os dados e as operações que julgar necessários, para dar suporte ao Restaurante Caseiro Hipotético (ver figura). Dica: operações para *abrir conta*, *fechar conta*, *verificar consumo*, *registrar item pedido* devem ser definidas no modelo.

Exercício 3

- Crie um modelo EquacaoDoSegundoGrau que contenha somente uma operação, que calcule as raízes da equação. Considere que os valores de a, b e c serão passados como parâmetros
- Aprimore o modelo, criando dados e delegando responsabilidades a outras operações
- Qual a complexidade adicional de se criar esse modelo, quando comparado com um algoritmo estruturado sem o uso de modelos? Quais as vantagens esperadas?

Exercício extra

- Modifique a operação dataÉVálida do modelo Data para que esta considere o valor máximo para o dia, dependendo do mês. Para fevereiro o valor máximo deve ser calculado em função do ano ser bissexto ou não. *Dica:* Anos bissextos (tendo 29 dias em fevereiro) são divisíveis por quatro, a não ser que sejam divisíveis por 100. Anos que podem ser divididos por 400 também são bissextos. A operação de divisibilidade pode ser implementada pela função módulo, representada pelo sinal %.

Mais exercícios (ec1)

No estudo de caso a seguir, identifique as classes e seus atributos.
Venda de produtos de limpeza.

- Cada produto é caracterizado por um código único, nome do produto, categoria (ex. detergente, sabão em pó, sabonete, etc), data de fabricação, data de vencimento e seu preço.
- A firma possui informações sobre todos seus clientes. Cada cliente é identificado por um código único (também interno à firma), o nome do cliente, endereço (rua, número, sala, cidade, cep, UF), telefone, filiação, status do cliente ("bom", "médio", "ruim"), e o seu limite de crédito.
- Guarda-se igualmente a informação dos pedidos de produtos feitos pelos clientes. Cada pedido possui um número (único), a data de elaboração do pedido, quantidade de produtos pedidos, valor total do pedido. Cada pedido pode envolver de um a vários produtos.
- Cada produto vendido pela firma possui o seu fornecedor. Cada fornecedor possui CNPJ, Nome Fantasia, telefone, endereço, etc. Um fornecedor pode fornecer vários produtos.

Mais exercícios (ec2)

No estudo de caso a seguir, identifique as classes e seus atributos.
Vídeo locadora, módulo de locação de DVDs de filmes.

- Um filme tem obrigatoriamente ao menos uma cópia, mas pode possuir diversas delas, porém uma cópia refere-se exclusivamente a um determinado filme.
- Um sócio pode realizar muitas locações enquanto permanecer sócio da locadora, mas uma locação
- refere-se unicamente a um determinado sócio.
- Cada locação deve obrigatoriamente referenciar-se ao menos a uma cópia de um filme, podendo
- referenciar-se a muitas cópias, no entanto uma mesma cópia pode ter sido locada diversas vezes, em épocas diferentes obviamente.

Outros exercícios sugeridos

- Resolver os exercícios do capítulo 1 do livro (SANTOS, Rafael) e trazer suas dúvidas para debate.