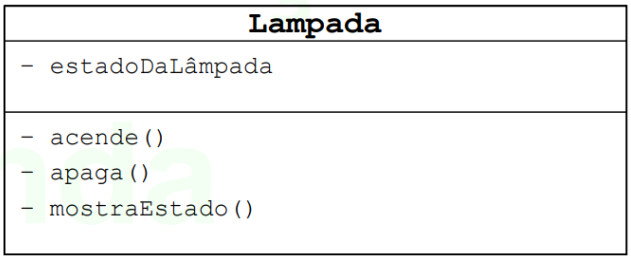
Lista de Exercícios

Tema 03. Introdução à Programação Orientada a Objetos. Modelos.

Nota: Se recomenda o uso de pseudocódigo para à implementação dos modelos em todos os exercícios desta lista.

1. Considerando o modelo de lâmpada mostrado em sala de aulas e sua implementação em 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
```

Projete e implemente um modelo para representar uma lâmpada a venda em um supermercado. Que dados devem ser representados por este modelo? Que operações este modelo deve realizar?

2. Imagine uma lâmpada que possa ter três estados: apagada, acesa e meia-luz. Usando o modelo Lampada (Exercício 1) como base, escreva o modelo LampadaTresEstados.

3. Generalize o modelo LampadaTresEstados do exercício 2 para que ele possa representar uma lâmpada onde a luminosidade pode ser ajustada com qualquer valor entre 0% (apagada) e 100% (acessa).

4. Inclua no modelo Lampada (Exercício 1) uma operação estaLigada que retorne verdadeiro se a lâmpada estiver ligada e falso caso contrário.

5. Considere o modelo ContaBancariaSimplificada e sua implementação

ContaBancariaSimplificada
- nomeDoCorrentista - saldo - contaÉEspecial
- abreConta(nome,depósito,éEspecial) - abreContaSimples(nome) - deposita(valor) - retira(valor) - mostraDados()

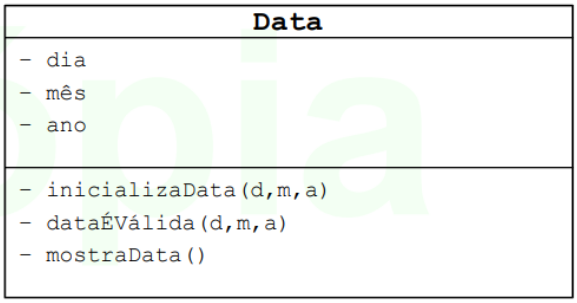
```
1 modelo ContaBancariaSimplificada
2 início do modelo
3   dado nomeDoCorrentista,saldo,contaÉEspecial; // dados da conta
4
5   // Inicializa simultaneamente todos os dados do modelo
6   operação abreConta(nome,depósito,especial) // argumentos para esta operação
7     início
8       // Usa os argumentos passados para inicializar os dados do modelo
9       nomeDoCorrentista = nome;
```

```
10        saldo = depósito;
11        contaÉEspecial = especial;
12      fim
13
14    // Inicializa simultaneamente todos os dados do modelo, usando o nome
15    // passado como argumento e os outros valores com valores default
16    operação abreContaSimples(nome) // argumento para esta operação
17      início
18        nomeDoCorrentista = nome;
19        saldo = 0.00;
20        contaÉEspecial = falso;
21      fim
22
23    // Deposita um valor na conta
24    operação deposita(valor)
25      início
26        saldo = saldo + valor;
27      fim
28
29    // Retira um valor da conta
30    operação retira(valor)
31      início
32        se (contaÉEspecial == falso) // A conta não é especial !
33          início
34            se (valor <= saldo) // se existe saldo suficiente...
35              saldo = saldo - valor; // faz a retirada.
36          fim
37        senão // A conta é especial, pode retirar à vontade !
38          saldo = saldo - valor;
39      fim
40
41    operação mostraDados() // mostra os dados da conta, imprimindo os seus valores
42      início
43        imprime "O nome do correntista é ";
44        imprime nomeDoCorrentista;
45        imprime "O saldo é ";
46        imprime saldo;
47        se (contaÉEspecial) imprime "A conta é especial.";
48        senão imprime "A conta é comum.";
49      fim
50
51  fim do modelo
```

A operação abreConta permite que alguém crie uma conta bancária passando como argumento um valor negativo, criando uma conta já em débito. Modifique a operação abreConta para que se alguém passar um saldo inicial negativo, que este seja considerado como zero.

6. A operação abreConta do modelo ContaBancariaSimplificada (Exercício 5) para que, caso o saldo negativo, uma mensagem de alerta seja impressa. Considere que o saldo só poderá ser negativo se a conta for especial.

7. Baseado no modelo Data e sua implementação em 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

  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;
```

```
29        // Senão, retorna falso
30        senão
31          retorna falso;
32      fim
33
34    operação mostraData() // mostra a data imprimindo valores de seus dados
35      início
36        imprime dia;
37        imprime "/";
38        imprime mês;
39        imprime "/";
40        imprime ano;
41      fim
42
43  fim do modelo
```


Crie o modelo HoraAproximada, que representa uma hora qualquer (usando valores para representar horas e minutos). Que dados e operações este modelo deve ter?

8. Baseado no modelo Data e HoraAproximada do exercício 7, crie o modelo HoraPrecisa, que represente uma hora qualquer com precisão de centésimos de segundo (horas, minutos, segundos, centésimos de segundo). Que dados e operações este modelo deve ter? Que dados e operações podem ser usados do modelo HoraAproximada?

9. Crie um modelo Livro que represente os dados básicos de um livro sem se preocupar com sua finalidade.

10. Partindo do resultado do Exercício 9, crie um modelo LivroDeLivraria que represente os dados e operações básicas de um livro que está à venda em uma livraria.

11. Partindo do resultado do Exercício 9, crie um modelo LivroDeBiblioteca que represente os dados e operações básicas de um livro de uma biblioteca que pode ser emprestado a leitores. Compare e comente as diferenças com o modelo anterior.

12. Crie um modelo Ponto2D para representar um ponto no espaço cartesiano de duas dimensões. Que dados e operações este modelo deve ter? Suponha um gráfico no qual você tenha que desenhar pontos usando este modelo.

13. Crie um modelo para representar uma linha, criado pela união de dois pontos no espaço cartesiano de duas dimensões, usando o modelo do Exercício 12. Que dados e operações este modelo deve ter?

14. Crie um modelo para representar um retângulo, cujos pontos opostos sejam instâncias do modelo Ponto2D (Exercício 12). Que dados e operações este modelo deve ter?

15. **A operação inicializaData do modelo Data (Exercício 7) tem uma abordagem simplista demais para verificar se o dia sendo usado é válido ou não: nesta operação ainda seria possível passar a data 31/02/2000 e a operação iria considerar os valores passados como sendo válidos. Modifique a operação dataÉVálida para que esta considere o valor máximo que pode ser aceito como válido dependendo do mês, de forma que para meses com 30 dias, o valor 31 para o dia seja considerado incorreto, e que para fevereiro o valor máximo seja 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. Desta forma, 1964 e 2000 são bissextos, mas 1900 não é bissexto.

16. **Aprimore o modelo Data (Exercício 7) de forma a incluir uma operação que retorne o dia da semana (domingo, segunda-feira, terça-feira, ..., sábado) para uma data válida.

Nota: Estes exercícios foram retirados do capítulo 1 do livro "Introdução à Programação Orientada a Objetos usando Java" de Rafael Santos.