# Exercícios – Tópico 2

1. Reimplemente os exercícios do tópico anterior, usando os conceitos de orientação a objeto.
2. Implemente uma classe **Livro**, imaginando que ela será usada como componente de software em uma biblioteca. A classe Livro deve ter membros de dados para ISBN, título, autor. Além disso, armazene dados que indique se o Livro está emprestado ou não. Crie os métodos accessors e mutators para a classe. Faça validações simples para os dados fornecidos para um Livro; por exemplo, aceite ISBNs somente na forma **n-n-n-x**, sendo **n** um número inteiro e **x** um dígito ou uma letra.
3. Crie um tipo enumerado para a classe **Livro**, chamado **Gênero**. Liste os tipos: ficção, não ficção, periódico, biografia, infantil. Atribua um **Gênero** a cada Livro e faça as alterações apropriadas nas funções membro de Livro e onde achar necessário.
4. Crie uma classe que representa um ponto no plano cartesiano. Em seguida, crie uma classe que representa um triângulo, reusando a classe anterior por composição. Finalmente, escreva um programa que receba do usuário as coordenadas dos vértices do triângulo e imprima seu perímetro.
5. Escreva uma classe **Time** com os atributos de hora, minuto e segundo. Implemente métodos para mostrar o horário em formato universal (HH:MM:SS) e no formato AM/PM (HH:MM:SS AM ou PM). Nas alterações dos atributos, apenas aceitar valores válidos. Implementar uma função *tick* para incrementar o tempo em 1 segundo. Apresente o funcionamento correto de todos os métodos da classe. Não esqueça de:
   - Incrementar para o próximo minuto.
   - Incrementar para a próxima hora.
   - Incrementar para o próximo dia.
6. Escreva uma classe **Date** com os atributos de dia, mês e ano. Implemente método para mostrar a data (DD/MM/AAAA). Nas alterações dos atributos, apenas aceitar valores válidos. Implementar uma função *nextDay* para incrementar o tempo em 1 dia. Apresente o funcionamento correto de todos os métodos da classe. Não esqueça de:
   1. Incrementar para o próximo mês.
   2. Incrementar para a próxima ano.
7. Combine a classe *Time* e *Date* em uma classe chamada **DateAndTime**. Realize as adequações necessárias para que as mudanças no método *tick* reflita no correto comportamento de *nextDay*. Escreva um programa para testar a nova classe *DateAndTime*.
8. Crie uma classe **Retangulo** com atributos comprimento e largura, cada uma com o valor padrão 1. Forneça métodos que calculem o perímetro e a área do retângulo. Além disso, forneça funções set e get para os atributos comprimento e largura. As funções mutators devem verificar se comprimento e largura estão entre os valores em ponto flutuante de 0.0 e 20.0 (inclusivo).
9. Crie uma classe que representa um funcionário, registrando seu nome, salário e data de admissão. Em seguida, crie uma classe que represente um departamento de uma empresa, registrando o nome e os funcionários que nele trabalham (para uso de vetores, considere um máximo de 100 funcionários). Por fim, crie uma classe que representa uma empresa, registrando seu nome, CNPJ e departamentos (considere um máximo de 10 departamentos). Faça um programa que:
   - Crie uma empresa;
   - Adicione a esta empresa alguns departamentos;
   - Adicione aos departamentos alguns funcionários;
   - Dê aumento de 10% a todos os funcionários de um determinado departamento;
   - Transfira um funcionário de um departamento para outro.

É esperado que seu código seja bem encapsulado. Por exemplo, para adicionar um departamento em uma empresa (ou um funcionário a um departamento), não se deve acessar o vetor (ou lista) de departamentos diretamente, mas sim ter um método na classe que representa a empresa para adicionar um departamento.