ADS – IFPB – Campus Monteiro

LISTA SEMANAL – PROGRAMAÇÃO II - PROGRAMAÇÃO ORIENTADA A OBJETOS  
Prof. Cleyton Caetano de Souza

**Observação:** Essa lista pode ser feita em dupla.

Semana 10

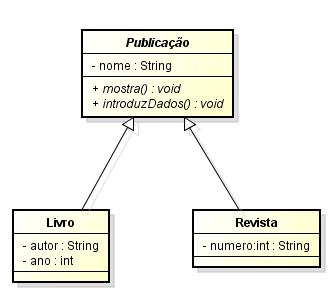
**Projeto - POOkemon**

1. Você já jogou Pokémon? Nesse jogo, cada personagem pode ter até quatro ataques (ou movimentos), que mudam. Quando o jogo começa, inclusive, o personagem tem apenas dois ataques e vai adquirindo mais, na medida em que evolui. Quando o Pokémon atingir o limite de quatro golpes, ao aprender o quinto, o jogador é perguntado se quer substituir algum dos movimentos atuais pelo novo. Assim, percebe-se que os ataques que um Pokémon tem variam bastante, durante o jogo, mas não em quantidade. Modele a classe Pokémon, com o atributo nome (String), tipo (String) e nível (inteiro). Um Pokémon tem quatro métodos para o ataque: ataqueA, ataqueB, ataqueC e ataqueD. Modele esses métodos para que eles possam mudar a forma como funcionam, dinamicamente, na medida em que o jogo ocorrer (inclusive, alguns deles, no começo, podem não fazer nada). Para ajudar, sugere-se pesquisar sobre o padrão de projeto Strategy. A resposta desta questão pode ser apresentada tanto no formato de código, quanto no formato de diagrama de classes.

2. **[Questão do Monitor – Paulo]** Uma companhia aérea lida com dois tipos de clientes: cliente comum e cliente VIP. De cada cliente, a companhia guarda as seguintes informações: nome e cpf. A diferença principal entre o cliente comum e o cliente VIP é que o segundo tem um desconto de 35% no preço das passagens aéreas. Crie um programa que faça o cadastro de 50 clientes. Antes de ler as informações, pergunte qual o tipo de cliente está sendo cadastrado. Após ler os dados de todos os clientes, o programa deve realizar a venda de uma passagem aérea, no valor de 1000 reais. Pergunte o CPF do cliente que vai comprar a passagem e informe o preço que esse cliente deverá pagar.

3. **[Autoria: Janeide (Monitora)]** Crie uma classe abstrata chamada FormatacaoTexto, que possuirá um método abstrato chamado formatarTexto com retorno do tipo String e que receberá um parâmetro de entrada do tipo String referente ao texto que se deseja formatar. Crie três classes chamadas: Maiúscula, Minúscula e Invertida. Todas devem estender da classe FormatacaoTexto e implementar o método formatarTexto de acordo com as suas particularidades. Por exemplo, na classe Maiúscula eu passo a String “Programação Orientada a Objetos” e o retorno o método formatarTexto deve ser “PROGRAMAÇÃO ORIENTADA A OBJETOS”. Escreva um programa no qual o usuário deverá digitar alguma mensagem e, em seguida, escolher qual formatador ele deseja usar para imprimir a mensagem no console; ao final, use o formatador escolhido para imprimir a mensagem no console.

4. Implemente as classes especificadas no diagrama de classes a seguir (Atente-se para as classes e métodos abstratos). O método **mostra** exibe o valor dos atributos da classe, no console. O método **introduzDados** realizará a leitura do valor dos atributos da classe, usando Scanner.



**Observação:** Colocar a impressão dentro do método “mostra” e a leitura dentro do método “introduzDados” é um **péssimo design**, entretanto, para fins didáticos, esse péssimo design vai resultar num ótimo exemplo do benefício de usar o polimorfismo.

5. Crie um programa para testar as classes criadas, anteriormente. Deve-se armazenar uma lista de publicações cadastradas pelo usuário em um ArrayList. No programa, deve ser exibido um menu com as seguintes opções: 1 – cadastrar novo livro, 2 – cadastrar nova revista, 3 – lista todas as publicações, 4 – listar apenas os livros, 5 – listar apenas as revistas e S - Sair. Ao escolher uma opção de cadastro, o usuário deve introduzir os dados sobre a publicação (utilizando o método introduzDados). Ao escolher uma das opções de listagem, o programa deve mostrar os dados das publicações cadastradas até o momento (utilizando o método mostra e considerando o tipo). O programa deve continuar a funcionar enquanto o usuário não escolher a opção de sair. Ao fim do programa, informe quantas publicações de cada tipo foram cadastradas (mas, só faça essa contagem, após todas as contas estarem no ArrayList).