Exercícios Java - Semana 3

Exercício 1: Classes Abstratas e Herança

Descrição: Crie uma hierarquia de classes para veículos.

1. **Passo 1**: Crie uma classe abstrata `Veiculo` com os seguintes atributos e métodos:
    - Atributos: `String marca`, `String modelo`, `int ano`
    - Métodos:
        - `abstract void ligar()`: método abstrato para ligar o veículo.
        - `abstract void desligar()`: método abstrato para parar o veículo.
        - `void mostraInfos()`: método concreto que exibe as informações do veículo.
2. **Passo 2**: Crie duas subclasses `Carro` e `Moto` que estendem `Veiculo` e implementam os métodos abstratos `ligar()` e desligar().
3. **Passo 3**: No método `main`, crie instâncias de `Carro` e `Moto`, chame os métodos `ligar()`, desligar() e mostrarInfos().

Exercício 2: Polimorfismo

Descrição: Trabalhe com polimorfismo usando a hierarquia de classes do Exercício 1.

1. **Passo 1**: No método `main`, crie um array de `Veiculo` e adicione instâncias de `Carro` e `Moto`.
2. **Passo 2**: Percorra o array e chame os métodos `ligar()`, `desligar()` e `mostraInfos()` para cada elemento do array.

Exercício 3: Interfaces

Descrição: Crie uma interface `Playable` para jogos.

4. **Passo 1**: Crie uma interface `Playable` com os seguintes métodos:
    - `void play()`
    - `void pause()`
    - `void stop()`
5. **Passo 2**: Crie duas classes `Music` e `Video` que implementam a interface `Playable`. Cada classe deve fornecer sua própria implementação para os métodos `play()`, `pause()` e `stop()`.
6. **Passo 3**: No método `main`, crie instâncias de `Music` e `Video`, e chame os métodos `play()`, `pause()` e `stop()`.

Exercício 4: Aplicação Prática

Descrição: Crie um sistema simples para gerenciar uma biblioteca.

Passo 1: Crie uma interface `Aluguel` com os seguintes métodos:

- `void alugarItem(String NomeCliente)`
- `void retornaItem()`

Passo 2: Crie uma classe abstrata `ItemBiblioteca` com os seguintes atributos e métodos:

- Atributos: `String titulo`, `String autor`, `boolean estaAlugado`
- Métodos:
    - `abstract void mostrarDetalhes()`
    - `void checarStatus()`: método concreto que imprime se o item está emprestado ou disponível.

Passo 3: Crie duas subclasses `Livro` e `Revista` que estendem `ItemBiblioteca` e implementam `Aluguel`. Ambas as classes devem fornecer implementações para `mostrarDetalhes()`, `AlugarItem()` e `retornarItem()`.

Passo 4: No método `main`, crie instâncias de `Livro` e `Revista`, então simule o empréstimo e devolução de itens, exibindo os detalhes e status dos itens.