Lista de exercícios sobre Interfaces

Nos exercícios de 1 a 4 considere o exemplo Aluno, Funcionario Atleta apresentado em aula:

1. Imagine que por alguma razão os dados dos funcionários têm de ser armazenados todos em uma única String separados por ",". Analise o impacto desta nova implementação da classe *Funcionario*. Considerando que a implementação antiga deve continuar, qual seria a melhor solução para o problema?

2. Todas as instituições de ensino a partir de um certo tamanho devem manter brigadas de incêndio voluntárias. Tanto alunos como funcionários (atletas ou não) podem ser voluntários. Voluntários são definidos pela interface "*Voluntario"* que segue. O código é um código de duas letras que indica o tipo de ação no qual o voluntário pode atuar. A partir da interface *Voluntário,* crie uma classe capaz de manter o cadastro de voluntários de uma Instituição.

   ```
   interface Voluntario{
    String getNomeVoluntario();
    String getCodigo();
   }
   ```

3. Explore o uso da interface *List* de Java e suas implementações (*ArrayList* e *LinkedList*) na implementação do exercício 2.

4. Analise as abstrações que modelam funcionário e aluno. Existem dados/métodos em comum entre elas? Em caso positivo, tente refatorar todo o código de maneira a usar herança para evitar duplicação de código.

5. Altere a classe *Funcionario* da lista de exercícios sobre herança de maneira que a mesma implemente a interface *Comparable<Funcionario>*. Teste o uso desta interface no método *sort* da classe *Arrays* de java ou no método *sort* da interface *List*.

6. Crie uma classe que implemente a interface *Comparator* visando comparar instancias de *Funcionario*. Teste seu uso no método no método *sort* da classe *Arrays*.

7. Compare os usos das interfaces *Comparable* e *Comparator*. Explique as vantagens e desvantagens de cada um.