Roteiro 05 – Exercícios de Fixação

1. O método de uma operação de objeto pode manipular atributos de classe (estáticos)? Justifique sua resposta.
2. O método de uma operação de classe (estática) pode manipular atributos de objeto? Justifique sua resposta.
3. O método de uma operação de classe (estática) pode acessar a palavra reservada `this`? Justifique sua resposta.
4. Qual a função do bloco `static` em uma classe Java?
5. No nosso diagrama UML (antes da criação da classe de configuração), por que os atributos `DATE_FORMAT_DDMMYYYY` e `nextId` estão sublinhados?
6. No nosso diagrama UML (antes da criação da classe de configuração), por que o atributo `DATE_FORMAT_DDMMYYYY` está com a propriedade `{read only}` e `nextId` não?
7. O que é uma API?
8. O que é polimorfismo paramétrico?
9. Como polimorfismo paramétrico é utilizado em Java?
10. O que é uma classe genérica?
11. Qual a diferença entre **upcasting** e **downcasting**?
12. Explique por que, no nosso exemplo, se trocarmos o **downcast** `(Employee)` por `(Student)`, será disparada a exceção **ClassCastException** somente em tempo de execução?
13. O que acontece se, na linha do **downcast**, modificarmos apenas a posição de 0 para 2? Explique o porquê de ter dado certo ou errado.
14. Quando devemos utilizar `instanceof`?
15. Por que o uso abusivo de `instanceof` pode ser um indicador de que sua solução pode ser melhorada?
16. O que é refatoração (*refactoring*)?
17. Quando devemos refatorar o código?
18. O que é uma operação abstrata?
19. Uma classe abstrata precisa ter operações abstratas? Justifique.
20. Uma classe que possui alguma operação abstrata precisa ser abstrata? Justifique.
21. Explique a metáfora de um contrato em relação a operações abstratas.
22. Considere que B é uma subclasse de A e C é uma subclasse de B:
    a. Se A possui uma operação abstrata, B é obrigada a implementá-la? Justifique.
    b. Se A possui uma operação abstrata, C é obrigada a implementá-la? Justifique.
23. Por que a expressão `a.equals(null)` deve retornar sempre `false`?
24. Por que não podemos comparar dois objetos somente com "=="?

Roteiro 05 – Exercícios de Fixação

25. Certifique-se que você modelou, implementou, e testou as operações `toString`, `equals` e `hashCode` para todas as classes desenvolvidas até o momento.

26. Modifique a classe `AppConfig` para que ela utilize um `ArrayList` ao invés de um vetor de tamanho fixo.

27. Modele e implemente uma solução para relatórios simples (cabeçalho, linhas e rodapé), sem formatação de página. A saída esperada de um relatório simples é:

    <Texto do cabeçalho>
    <Texto da primeira linha>
    <Texto da segunda linha>
    ...
    <Texto da última linha>
    <Texto do rodapé>

    Você deve implementar dois relatórios: a) um para imprimir a sequência (cada valor é uma linha) 10, 20, 30, etc.; b) outro para imprimir a sequência de Fibonacci: 0, 1, 1, 2, 3, 5, 8, 13, 21, etc. O número de linhas deve ser informado no construtor do relatório. A solução deve, obrigatoriamente, utilizar o padrão de projeto **Método Template**. Lembre-se que a ideia aqui é reutilizar o algoritmo básico do relatório simples para os dois relatórios específicos.