Roteiro 04 – Exercícios de Fixação

1. Explique o conceito de pilha de execução e qual a sua relação com o tratamento de exceções em Java.

2. Qual a diferença entre `catch` e `finally`?

3. Quando devemos utilizar `catch`?

4. Quando devemos utilizar `finally`?

5. Quais as diferenças entre exceções checadas (*checked*) e não checadas (*unchecked*) em Java?

6. Quais as orientações para escolher entre uma exceção checada e uma não checada?

7. Em quais situações devemos declarar uma exceção no método com a cláusula `throws`?

8. Quando devemos criar a nossa própria classe de exceção? Quais orientações devem ser seguidas?

9. Por que colocar `try-catch(Exception)` ao longo do programa para evitar que ele aborte não é uma boa prática?

10. O que é um invariante da classe?

11. Como podemos respeitar invariantes em Java?

12. Considerando o contexto do nosso sistema acadêmico, implemente regras de validação para as classes implementadas até o momento. Preencha corretamente o **javadoc**, incluindo também a **annotation** `@throws`. Além de utilizar `OutOfRangeException`, você pode criar outras exceções (desde que elas agreguem maior entendimento sobre o erro – veja as orientações). Por último, atualize o seu diagrama de classe.
    - Um nome não pode ser nulo e deve ser formado de pelo menos duas palavras.
    - A data de nascimento não pode ser nula e deve ser anterior à data atual.
    - A data de admissão/matrícula de um estudante não pode ser nula e não pode ser superior à data atual.
    - A data de desligamento de um estudante não pode ser inferior à data de admissão/matrícula.
    - Um estudante que estiver na situação matriculado não pode ter data de desligamento.
    - Um estudante que tiver data de desligamento não pode estar na situação matriculado.
    - As horas contratadas por semana para um empregado devem estar entre 1 e 40, inclusive (revisar implementação apresentada nos exemplos).
    - A data de contratação de um empregado não pode ser nula, não pode ser superior à data atual e nem superior à data de demissão.
    - A data de demissão de um empregado não pode ser inferior à data de contratação.
    - Um empregado que estiver na situação trabalhando não pode ter data de demissão.
    - Um empregado que tiver data de demissão não pode estar na situação trabalhando.
    - O valor da hora de um empregado deve ser positivo, maior do que zero.

13. No método `validateHoursPerWorkWeek`, nós utilizamos diretamente os valores para a faixa esperada (1 e 40). Melhore seu código, transformando estes valores em constantes na classe `Employee`. Este tipo de prática ajuda na manutenibilidade (esforço para procurar e corrigir erros e/ou mudar um programa) e na legibilidade.

14. Na nossa solução atual, uma exceção é disparada imediatamente quando uma regra de negócio for quebrada. Como seria uma solução alternativa para avaliar o estado inteiro do objeto e disparar uma única exceção com todos os erros reportados nela. Implemente sua proposta e veja se ela realmente funciona.