Exceções

MAC0321 – Lista 06

1 de maio de 2024

Instruções de Entrega

Nome do projeto java: ListaX-NUSP1-NUSP2. Sendo X o número da lista para ser realizado em duplas; NUSP1 e NUSP2 são os números usp dos participates.

Entregar o projeto zipado, **apenas um projeto por dupla.**

Exercícios

1. (2.5 ponto) Crie uma classe `SobrevivePau`, com um método `geraPau()` que sempre gera uma excecão `ArrayIndexOutOfBoundsException`. Fazer um outro programa pegaPau() que captura de forma adequada (usando o comando `catch`) essa exceção. Fazer uma classe de testes que testa esses dois métodos.

2. (2.5 pontos) Modifique o Exercício 01 da Lista 03, de forma que um depósito só possa ser feito se o valor depositado é estritamente positivo, e os saques só podem ser feitos se o valor sacado for positivo e o saldo existente permitir; além isso, um depósito só pode ser feito se o valor a ser depositado é positivo. Crie uma classe de exceções chamada de `NegativeValueException` derivada da classe `RuntimeException` e faça com que seu programa gere uma exceção para cada um dos casos de erros mencionados acima. Use o `finally` para garantir que o saldo nunca é negativo; caso contrário, uma mensagem de erro deve ser impressa.

   Gere testes automáticos que verifiquem que exceções são geradas em casos de depósito ou saldo de valores negativos.

3. (1.5 pontos) Modifique o exemplo anterior para que uma nova classe `NegativeValueException2` seja derivada da classe `Exception`; note se houve erros de sintaxe ao substituir a classe original por essa e, se houver, trate de consertá-los. Gere testes automáticos que verifique que exceções são geradas em casos de depósito ou saldo de valores negativos e que mostre a diferença entre estes dois tipos de Exceções.

4. (3.5 pontos sofridos) Suponha que você esteja construindo uma classe `Codificadora`, escrevendo um método que analisa o formato do código:

```
void analisaCodigo(String codigo){ ... }
```

onde `codigo` deve ter o seguinte formato

`matriz` ⟨`tipo`⟩ ⟨`num linhas`⟩ ⟨`num colunas`⟩

onde `matriz` é uma palavra obrigatória no início do código, o tipo pode ser um de `binary, int,` ou `double`, e `num linhas` e `num colunas` são inteiros positivos não nulos.

Crie ao menos 4 tipos de exceções para os diversos erros que podem ocorrer na decodificação do código no formato acima. Estas exceções devem ser jgadas pelo método `analisaCodigo(código)` de acordo com a análise do código recebido como parâmetro.

Crie um programa JUnit que verifique que as exceções são jogadas nos casos em que o método recebe um string que não obedece ao formato acima. Cada tipo de exceção deve ser testada por um teste distinto.