Lista de Exercício do 2º Estágio 2015.2

Objetivo

O objetivo desta lista de exercícios é avaliar o conhecimento adquirido pelo aluno neste estágio da disciplina com relação aos conceitos de Orientação a Objetos, Framework de Collections e JUnit .

Avisos

Alguns avisos/dicas/sugestões para serem utilizadas nessa lista:

- As respostas para a lista devem ser entregues até a data marcada pelo professor.
- Devem ser enviadas via *GitHub* conforme apresentado em sala de aula e dentro de um repositório chamado "[MLPIII_CC__2015_2_LISTA2]".
- Devem ser aplicados os conceitos de OO vistos em sala de aula.
- Para aumentar a interação do seu programa com o usuário, informe-o quais dados você quer que ele digite e quais dados você está retornando

Exercícios

1) Criar uma classe Item com os atributos código do item e descrição do item. Criar uma classe Inventario que tem uma coleção ordenada de itens. A ordem dos itens deve ser feita pelo código do item. Implementar uma classe Principal, testar a iteração sobre os itens dessa coleção, verificar se a coleção está vazia, mostrar uma mensagem caso a coleção esteja vazia, verificar se um determinado item pertence a coleção e por fim, mostrar uma mensagem caso um item não esteja presente na coleção.

2) Implementar testes unitários para as classes ContaCorrente e ContaPoupanca desenvolvidas em sala de aula utilizando-se obrigatoriamente dos métodos assertEquals(a,b), assertFalse(a), assertTrue(a), assertNotNull(a), assertNull(a), assertNotSame(a,b), assertSame(a,b) e fail() da API resumida do JUnit.

3) Desenvolva uma mini aplicação para um curso de capacitação, que contém um número ilimitado de alunos, sob a orientação de um único professor. Nosso objetivo é disponibilizar as seguintes funcionalidades:
   a) Incluir alunos
   b) Remover aluno (por nome ou matrícula)
   c) Verificar se um aluno está matriculado (por matrícula)
   d) Listar todos os alunos aprovados (média superior à 7,0)
   e) Listar todos os alunos reprovados por falta (faltas superior à 15)
   f) Listar por ordem alfabética

Sabendo que o cadastro do Aluno possui, nome, matrícula e média e o cadastro do Professor possui nome, matrícula e especialidade e o cadastro do Curso de Capacitação contém além dos Alunos e Professor, nome e número de identificação.

4) Na aula de conjuntos foi dito que a classe LinkedHashSet é mais rápida que a classe TreeSet, pois esta última tem complexidade de tempo de O(log(n)). Como consultor contratado pelo Facebook Inc. para descobrir por que o sistema de envio de mensagens está tão lento, emita um parecer embasando sua decisão de alterar todas as implementações que o facebook utiliza de TreeSet para LinkedHashSet. Dica: além de escrever um texto embasando sua consultoria escreva um trecho de código que verifique a performance entre essas duas classes.

5) Escreva uma classe chamada Propriedades (classe principal) que leia os parâmetros de conexão de banco de dados de um arquivos chamado database.properties. Dica utilizar a classe Properties que herda da classe de HashTable que por usa vez implementa um Map. O arquivo deve se chamar database.properties e conter as linhas abaixo:

```
conexao.url=localhost
conexão.database=MPLIIIDB
conexao.usuario=jeffersondb
conexao.senha=secreta
conexão.porta=5432
```

Dicas

- Revisar os *slides* vistos em sala de aula e a apostila da *Caelum* disponibilizada pelo professor no ambiente *Moodle*.