Programação Avançada

2010/2011

Revisão de Conceitos – Manipulação de Objectos

Pretende-se que o aluno reveja conhecimentos na manipulação de objectos, incluindo referências para objectos e cópias de objectos.

Requisitos:

- *JDK 5.0 ou superior (http://java.sun.com/javase/downloads/index.jsp)*
- *NetBeans 5.5 ou superior (http://www.netbeans.info/downloads/index.php)*
- *Capacidade de interpretação e racciocínio.*

Objectivos:

- *Compreender que em Java os objectos são tipos referenciados!*
- *Utilização e manipulação de arrays de objectos.*
- *Utilização de excepções.*
- *Tomar noção dos dois tipos de cópias de objectos:* deep *e* shallow.

Descrição do Problema

Imagine que queremos emular em software o funcionamento de uma *Disk Jukebox*. Esta consiste num conjunto de discos que são acessíveis através de um braço robótico (com uma cabeça de leitura/escrita) que é responsável pelo acesso individual a cada um dos discos. Isto significa que em cada instante o braço pode aceder a apenas um disco.

As especificações da Jukebox são as seguintes:

- Um disco tem, por omissão, uma capacidade de 1024 bytes (sim, é pequenino ☺, mas é o menos relevante para este exercício); Nota: Um carácter ocupa um byte!
- A Jukebox tem uma capacidade fixa de 20 discos (N = 20);
- Todos os discos têm a capacidade por omissão de 1024 bytes, excepto os últimos 2 discos que têm a capacidade de 2048 bytes;
- Inicialmente o braço robótico está sobre o primeiro disco (disco 0).

O conjunto inicial de operações pretendidas é a seguinte (especificando desde já como se relacionam com a implementação):

- Movimentar o braço robótico até determinado disco (o braço/cabeça será movido para uma determinada posição do conjunto de discos);
- Mostrar informação sobre o disco actualmente em sua posse (capacidade em bytes, espaço livre e que posição ocupa na Jukebox);
- Ler os dados contidos no disco (a informação será guardada numa String);
- Escrever um pedaço de informação no disco (a informação será adicionada à já existente);
- Formatar o disco (apagar a informação existente);
- Copiar discos (neste caso o braço robótico possui um *buffer* interno que é capaz de guardar a informação total de um disco, para posteriormente copiar para o disco-destino). Notar que só poderá fazer cópias de discos idênticos, i.e. com a mesma capacidade;
- Espelhar discos, género solução RAID 0. As alterações que forem feitas num disco, serão reflectidas noutro (aqui será muito importante a compreensão das referências para objectos).

Detalhes de Implementação

Que objectos serão necessários para resolver este problema?

- ✓ **Como guardar um conjunto de discos, acessíveis por uma posição/índice?**
- ✓ **Em que consistirá o braço robótico?**

Crie esses objectos, utilizando Composição onde for útil.

Crie os métodos e construtores que achar necessários. Redefina o método *toString()* onde achar adequado, de forma a implementar alguma funcionalidade.

Sempre que for referida uma posição inválida de disco, deverá ser lançada uma excepção do tipo *PosicaoInvalidaException*.
Da mesma forma, sempre que se tentar copiar ou espelhar discos de capacidades diferentes, deverá ser lançada a excepção *DiscosIncompativeisException*.

Classe Emulador

Juntamente com o enunciado é-lhe disponibilizada uma classe que implementa a interacção com o utilizador pela consola (como se fosse uma linha de comandos). Estude essa classe e adapte-a, inserindo o seu código onde achar necessário. Aconselha-se que seja adicionado um atributo do tipo da classe que representa a Jukebox, cujos métodos serão depois invocados. Apenas esta classe poderá ter *System.out*'s, pelo que as outras classes deverão retornar *Strings* nos métodos cujo resultado seja texto.

Os comandos que podem ser utilizados na linha de comandos (e que deverão ser implementados) são os seguintes:

- ***mover*** **<*p*>** : movimenta o braço robótico para a posição *p*.
- ***info*** : mostra informação sobre o disco actual.
- ***ler*** : ler os dados contidos no disco (deverão ser apresentados na consola).
- ***escrever*** **<*dados*>** : adiciona os dados pretendidos (apenas uma palavra), aos dados já existentes, com uma quebra de linha.
- ***formatar*** : apaga o conteúdo do disco em utilização.
- ***copiar*** **<*a*>** **<*b*>** : copia o conteúdo do disco *a* para o disco *b*. Deverão ter a mesma capacidade.
- ***espelhar*** **<*a*>** **<*b*>** : passa a espelhar o disco *a* em *b*.
- ***backup*** : faz o backup da jukebox primária na secundária (só na Secção seguinte)
- ***usar*** **<*x*>** : usar jukebox primária (0) ou secundária/backup (1) (só na Secção seguinte)
- ***sair*** : sai do programa e destrói a Jukebox :)

Nesta classe **não deverá ser implementada nenhuma lógica**, ie. apenas deverão ser invocados métodos de um objecto que represente a Jukebox e mostrando mensagens mediante o resultado desses métodos.

Backup da Jukebox

Adicione uma outra Jukebox na qual possa guardar o *backup* da informação existente na actual. Após ter sido feito o *backup*, o braço robótico da Jukebox secundária deverá estar sobre o primeiro disco.

Para tal, deverá adicionar um novo construtor na classe que representa a Jukebox, que receba a referência de outra em argumento e copie o seu conteúdo.

Implemente os comandos *backup* e *usar*, descritos anteriormente.

- ✓ Como alternar entre as Jukeboxes? Existe uma solução simples na qual não tem que alterar nenhum código já efectuado até aqui.

Testar o Programa

A regra é simples: teste à medida que implementa!

Será importante testar que as cópias dos discos estão a ser efectuadas correctamente.

Se espelharmos o disco 0 no disco 5 e consequentemente o disco 5 no 10, o que deverá acontecer se forem adicionados dados ao disco 0?

Após ser efectuado um *backup*, modifique um disco na Jukebox primária. Esta alteração reflectiu-se na Jukebox secundária? Tal, obviamente, não deverá acontecer. Rectifique esta situação se ela ocorrer. *Dica: deep-copy vs. shallow-copy*