

Lista de Exercícios

MC302 - Programação Orientada a Objetos
Instituto de Computação
Universidade Estadual de Campinas

Design Patterns
2015
André Santanchè

Questão 1 (questão de prova)

Dada a seguinte classe cujos objetos representam uma saldo bancário (armazenado no atributo `valor`) e sua operação de movimentação da conta:

```
public class Saldo implements Transacao {
    private float valor;
    public float getValor() {
        return valor;
    }
    public void movimento(float valorMovimento) {
        valor += valorMovimento;
    }
}
```

A classe implementa a seguinte interface:

```
public interface Transacao {
    public void movimento(float valorMovimento);
}
```

Escreva uma classe herdeira de `Saldo`, denominada `SaldoRepl`, que possibilite replicar suas operações de movimento em outros objetos.

Qualquer objeto `X` da classe `SaldoRepl` deve oferecer um método em que outros *objetos assinantes* se cadastrem. Este método será acionado pelos *objetos assinantes* que desejam ser avisados quando ocorrer uma operação de movimento em `X`. A única exigência é que os *objetos assinantes* devem implementar a interface `Transacao`. A operação de movimento da classe `SaldoRepl` deverá ser estendida de tal modo que, cada vez que ela é acionada em `X`, este mesmo objeto `X` irá acionar a mesma operação (com o mesmo parâmetro) de todos os *objetos assinantes*.

Questão 2

Um laboratório de pesquisa está automatizando seu processo de testes biológicos. Cada experimento envolve uma sequência de ações automáticas a ser aplicadas em uma cultura de microrganismos, a fim de verificar a reação dos mesmos. Exemplos de ações são: elevação da temperatura ambiente, aplicação de um produto químico etc.

Cada ação é aplicada automaticamente por um robô controlado por um objeto de software que implementa a interface:

```
public interface RoboAction {
    public void execute();
}
```

Cada robô executa uma única ação e não é necessário informar parâmetros da ação.

Escreva uma classe denominada `Experimento` cujos objetos são capazes de aplicar um teste envolvendo uma sequência de ações, que podem ser realizadas por diferentes robôs. Cada objeto `Experimento` deve guardar uma lista com a sequência de ações a ser executada (cada ação é executada por um único objeto robô o qual ele guarda a referência) e deve implementar operações que permitam adicionar ações (sempre no final da sequência) e executar o teste (na sequência em que as ações foram cadastradas).

Depois de ter resolvido o exercício, leia sobre o design pattern `Command`

(http://en.wikipedia.org/wiki/Command_pattern) e analise qual a relação com o código que você implementou.

Expansão:

Escreva um método extra denominado `clone()`, que fará parte da classe `Experimento` que é capaz de clonar um experimento completo. Clonar significa realizar uma duplicata da sequência do experimento, incluindo a duplicação de todos os objetos do experimento.

Questão 3

O laboratório resolveu criar um processo automático para descobrir experimentos, que são mais eficientes na destruição de um dado tipo de microrganismo. Como as possibilidades de combinação são infinitas, decidiu-se usar um método baseado na programação genética.

Considere que existe uma classe `ExperimentoMonitorado` representando um experimento com funcionalidades extra de monitoramento. Esta classe é herdeira de `Experimento` e acrescenta o método `contaMicro()`, que não recebe parâmetros e retorna a contagem de microrganismos. Este método pode ser aplicado antes e depois da execução do experimento, de modo a verificar sua efetividade.

Implemente o processo em duas etapas:

Crie uma variante do método `clone` chamado `cloneMutant` que produz mutantes. Este método ao realizar a clonagem aplica uma única mutação na cópia que pode ser:

- um par de operações trocadas
- uma operação ausente
- uma operação duplicada

A posição da operação que sofre mutação também deve ser aleatória.

Crie um método estático (apenas o método sem uma classe) que receba dois parâmetros: um objeto da classe `Experimento` (base) e o número de ciclos que será usado para encontrar o melhor experimento. Este método deve atuar da seguinte maneira:

- (1) cria uma lista de melhores experimentos com 5 posições (inicialmente vazia)
- (2) adiciona o experimento base (recebido como parâmetro) na lista dos melhores
- (3) percorre a lista dos melhores clonando (com mutação) cada um de seus componentes
- (4) aplica todos os experimentos disponíveis (os da lista dos melhores e os clones)
- (5) seleciona os 5 melhores resultados (se houver) e os coloca na lista dos melhores
- (6) conta um ciclo e retorna ao passo (3) se não tiver completado o número de ciclos

Questão 4

Dada uma classe abstrata (já implementada) denominada `Noticiario` cuja função é enviar notícias para assinantes. Cada objeto desta classe fica monitorando uma fonte de notícias e quando surge uma nova notícia chama um método abstrato da própria classe com a seguinte assinatura:

```
public abstract void notificaNoticia(String textoNoticia, int dia, int mes, String topico);
```

Os parâmetros indicam o texto da notícia, seu dia e mês e o tópico a que se refere respectivamente. Tópico é uma string usada para classificar as notícias por tema. Exemplos de tópico: lançamento de livro, greve, inovação tecnológica etc.

Considerando que qualquer objeto consumidor de notícias implemente a interface:

```
public interface ConsomeNoticia {
    public void notificaNoticia(String textoNoticia, int dia, int mes, String topico);
}
```

Escreva uma classe herdeira de `Noticiario` chamada `NoticiarioAssina` que implemente o *pattern Observer* para divulgar suas notícias.

Questão 5

Implemente as classes de dois possíveis tipos de objeto que consomem esta notícia (implementam `ConsomeNoticia`): agregadores e publicadores.

Um objeto publicador publica cada notícia recebida (neste exercício, imprime no console).

Um objeto agregador concatena várias notícias recebidas e as envia para assinantes, implementando o mesmo *pattern* do `NoticiarioAssina`. Ele pode ser de dois tipos: agregador por tópico ou agregador por mês.

O agregador por tópico escolhe um tópico para agregar (informado no construtor) e concatena sempre dez notícias do mesmo tópico. Cada vez que consegue agregar dez notícias, ele envia a string concatenada para seus assinantes com o dia e o mês da última notícia.

O agregador por mês concatena todas as notícias do mesmo mês e, apenas quando chega uma notícia de um mês diferente, envia a string concatenada para seus assinantes com o dia zero e o tópico “mensal”.

Expansão:

Otimize o processo do *pattern Observer* para que objetos consumidores (implementam `ConsomeNoticia`) possam assinar apenas o tópico de notícia que estão interessados. Deste modo, não receberão a notificação dos outros tópicos.

Questão 6

Faça uma aplicação que concatene agregadores em sequência, ligados a um objeto `NoticiaAssina`, com um publicador no final. Uma concatenação interessante envolve o filtrar um tópico e o agregá-lo mensalmente.

Questão 7

Considere que uma biblioteca gráfica disponibiliza uma classe `Visual` os seguintes métodos estáticos:

<code>drawLine(x1, y1, x2, y2)</code>	Desenha uma linha. x1 e y1 - coordenadas do ponto de origem; x2 e y2 - coordenadas do ponto de destino.
<code>drawRectangle(x, y, altura, largura)</code>	Desenha um retângulo. x e y - coordenadas do canto esquerdo superior; altura e largura do retângulo.
<code>drawCircle(x, y, raio)</code>	Desenha um círculo. x e y - coordenadas do centro; raio - raio do círculo.

A partir desta biblioteca:

- Escreva uma classe que seja capaz de armazenar o estado e plotar um objeto geométrico.
- Usando o *pattern Composite*: defina uma interface única para desenhar os objetos e crie uma classe que permita agregar objetos geométricos simples (linha, retângulo e círculo) e compostos (objetos que já agregam outros objetos).
- Considere a seguinte estratégia padronizada para definir as medidas de uma reta, um retângulo e

um círculo: para os três casos é definido um *bounding box*, ou seja, um retângulo delimitador com as coordenadas x_1 , y_1 , x_2 , y_2 .

- a linha é desenhada como uma diagonal partindo de x_1 , y_1 até x_2 , y_2 ;
- o retângulo tem seu canto esquerdo superior em x_1 , y_1 e o esquerdo inferior em x_2 , y_2 ;
- o círculo corresponde ao maior círculo que pode ser desenhado dentro do retângulo.

Escreva uma fábrica (pattern Abstract Factory) para construir objetos simples (um único método) e compostos (outro método).

d) Usando o pattern DAO, visto em sala, acrescente métodos na interface da questão (b) de modo que o objeto seja capaz de se gravar em um arquivo e se ler do mesmo.