Fundação CECIERJ - Vice Presidência de Educação Superior a Distância

Curso de Tecnologia em Sistemas de Computação
Disciplina de Arquitetura e Projeto de Sistemas
Gabarito da AP3 – 2° semestre de 2016

Observações:
1. Prova sem consulta e sem uso de equipamentos.
2. Use caneta para preencher o seu nome e assinar nas folhas de respostas.
3. Você pode usar lápis para responder as questões.
4. Ao final da prova devolva as folhas de questões e as de respostas.
5. Todas as respostas devem ser transcritas nas folhas de respostas. As respostas nas folhas de questões não serão corrigidas.

Questão 1 (2,5 pontos): Relacione cada conceito da coluna da esquerda com uma e somente uma descrição na coluna da direita que melhor o representa.

(a) Encapsulamento	(1) Interfaces devem ser implementadas por classes não abstratas
(b) Abstração	(2) O sistema deve ser decomposto em um conjunto altamente coeso e fracamente acoplado de objetos
(c) Modularidade	(3) Objetos herdam atributos e métodos de outros objetos
(d) Hierarquia	(4) A representação computacional do objeto real deve se concentrar nas características que são relevantes para o problema
(e) Polimorfismo	(5) O objeto deve esconder seus dados e os detalhes de sua implementação
	(6) Classes são capazes de instanciar objetos
	(7) Objetos podem ter métodos redefinidos de outros objetos
	(8) Pacotes contêm classes

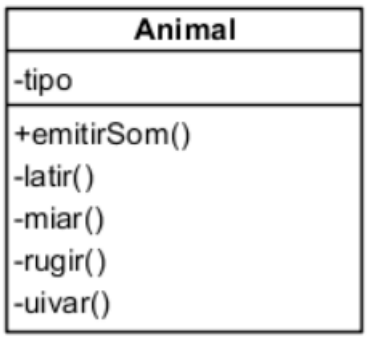
Resposta: a – 5; b – 4; c – 2; d – 3; e – 7

Questão 2 (2,5 pontos): Relacione cada elemento da coluna da esquerda com um e somente um elemento da coluna da direita.

(a) diagrama de classes	(1) Explicita os processos de negócio do cliente.
(b) diagrama de casos de uso	(2) Explicita as possibilidades de interação entre os usuários e o sistema.
(c) diagrama de transição de estados	(3) Detalha o comportamento de um objeto no decorrer da sua vida.
(d) diagrama de sequencia	(4) Detalha uma determinada possibilidade de interação entre o usuário e o sistema.
(e) diagrama de atividades	(5) Explicita a estrutura estática interna do sistema.
	(6) Detalha a interação entre diferentes objetos do sistema para atender a uma funcionalidade específica.

Resposta: a → 5; b → 2; c → 3; d → 6; e → 1;

Questão 3 (2,5 pontos): Observe a classe "Animal" e o trecho de implementação de seu método público "emitirSom". Considerando que outros tipos de animal e outros métodos públicos, representando outros comportamentos, poderão ser adicionados ao sistema na medida que ele evolui, responda.



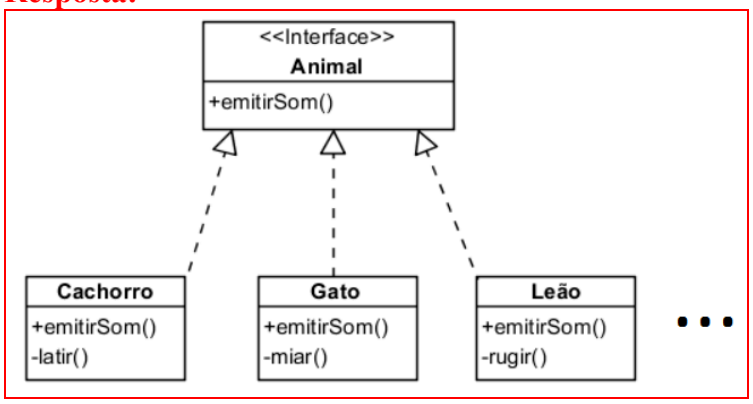

```
public class Animal{
    private String tipo;
    ...
    public void emitirSom(){
        if (tipo.equals("Cachorro")){
            latir();
        }
        if (tipo.equals("Gato")){
            latir();
        }
        if (tipo.equals("Leão")){
            rugir();
        }
        ...
    }
}
```

(a) Qual o problema desta solução e qual padrão GRASP poderia ser utilizado para resolver este problema (1 ponto)?

Resposta: O problema é a implementação do comportamento com base na testagem do tipo da classe (IF/TIPADO). O padrão GRASP que pode ser utilizado para resolver este problema é o Polimorfismo.

(b) Resolva o problema aplicando o padrão GRASP, desenhando o novo diagrama de classes (1 ponto).

Resposta:



Observação: Note que o atributo tipo não é mais necessário. A solução permite facilmente acrescentar novos tipos de animais e novos métodos públicos. Também facilita a manutenção com métodos mais curtos e mais focados.

Questão 4 (2,5 pontos): Indique o padrão GoF entre os vistos em aula (Abstract Factory, Singleton, Adapter, Composite, Façade, Proxy, Command, Observer, State, Strategy e Template Method) mais indicado para cada situação a seguir:

a) As classes cliente devem ser independentes de como seus produtos são criados, compostos ou representados.
b) As classes cliente devem ser capazes de ignorar a diferença entre composições de objetos e objetos individuais.
c) Deseja-se fornecer um substituto para um objeto quando for inconveniente ou indesejável acessá-lo diretamente.
d) Deseja-se fornecer uma interface diferente para um objeto, ajustando o para uma situação de uso específica.
e) Deseja-se assegurar que diversos objetos de diferentes classes possam ficar cientes de alterações no estado de um objeto específico.

Resposta: a → Abstract Factory, b → Composite, c → Proxy, d → Adapter, e → Observer.