CONCURSO PÚBLICO

Maio - 2008

caern
COMPANHIA DE ÁGUAS E ESGOTOS DO RIO GRANDE DO NORTE

Analista de Sistemas

Leia estas instruções:

1	Confira se os dados contidos na parte inferior desta capa estão corretos e, em seguida, assine no espaço reservado para isso. Caso se identifique em qualquer outro local deste Caderno, você será eliminado do Concurso.
2	Verifique se este Caderno contém, respectivamente, **uma** proposta de Redação, **três** questões discursivas de Conhecimentos Específicos e **trinta** questões de múltipla escolha de Conhecimentos Específicos.
3	Se o Caderno contiver alguma imperfeição gráfica que impeça a leitura, comunique isso imediatamente ao Fiscal.
4	Na Redação e nas questões discursivas, você será avaliado **exclusivamente** por aquilo que escrever dentro dos espaços destinados ao texto definitivo e às respostas.
5	Escreva de modo legível. Dúvida gerada por grafia ou rasura implicará redução de pontos.
6	Cada questão de múltipla escolha apresenta apenas uma resposta correta.
7	Os rascunhos e as marcações feitas neste Caderno não serão considerados para efeito de avaliação.
8	Interpretar as questões faz parte da avaliação; portanto, não adianta pedir esclarecimentos aos Fiscais.
9	Utilize qualquer espaço em branco deste Caderno para rascunhos e não destaque nenhuma folha.
10	Você dispõe de, no máximo, quatro horas e meia para elaborar, em caráter definitivo, a Redação, responder às questões discursivas e às de múltipla escolha e preencher a Folha de Respostas.
11	O preenchimento da Folha de Respostas é de sua inteira responsabilidade.
12	Antes de retirar-se definitivamente da sala, devolva ao Fiscal a Folha de Respostas e este Caderno.

Assinatura do Candidato: _____

funpec

COMPERVE
UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE

Prova de Redação

O saneamento básico se constitui um direito do cidadão no mundo atual.

- Elabore um texto argumentativo no qual você explicite os prejuízos que o descumprimento desse direito acarreta para a população.

OBSERVAÇÕES:

- O texto deverá ser redigido em prosa, no registro padrão da língua portuguesa escrita, de forma coesa e coerente.
- Ao texto com **menos de 15 (quinze) linhas**, será atribuído **zero**.
- **NÃO assine** a Redação.

ESPAÇO DESTINADO À REDAÇÃO DEFINITIVA

1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
11	
12	
13	
14	
15	
16	
17	
18	

19	
20	
21	
22	
23	
24	
25	
26	
27	
28	
29	
30	
31	
32	
33	
34	
35	

NÃO assine a Redação.

Discursiva ⇨ Conhecimentos Específicos ⇨ 1 a 3

1. Considere o projeto de um *Blog* cuja persistência é feita através do *framework Hibernate*. Nesse projeto, existem duas entidades - Post e Comentario - e um relacionamento *one-to-many* (um-para-muitos) entre elas, ou seja, cada Post está associado a vários Comentários. Para listar todos os Posts e seus respectivos comentários, o programador desenvolveu o código abaixo:

```
Session session = getSession();
List<Post> posts = session.createCriteria(Post.class).list();
for (Post post : posts) {
     System.out.println(post.getTitulo());
     System.out.println(post.getConteudo());
     System.out.println("---- Comentários ----");
     for (Comentario comentario : post.getComentarios()) {
           System.out.println(comentario.getData());
           System.out.println(comentario.getConteudo());
     }
     System.out.println();
}
```

Ao executar-se esse código, foram realizadas diversas consultas ao banco de dados, o que tornou a implementação ineficiente. Após uma análise, o programador percebeu que o *Hibernate* estava realizando uma consulta para cada um dos comentários em cada um dos *posts*.

Com base nessas informações, responda:

A) por que esse problema aconteceu

B) quais seriam as possíveis soluções para o problema

Espaço destinado à Resposta

Fim do espaço destinado à Resposta

Rascunho

2. Na análise de um sistema de controle de águas e esgotos através da WEB, o analista designado para essa tarefa e o gestor responsável pelo sistema consideraram que era necessário utilizarem diagramas UML e relacionaram vários problemas a serem resolvidos:

- fluxo de negócio de um determinado caso de uso;
- operações do sistema e suas interações com os usuários e outros sistemas;
- entendimento das entidades de domínio do sistema;
- fluxo de ativação de componentes da arquitetura do sistema;
- ciclo de vida de um objeto no sistema;
- ambiente de produção da aplicação.

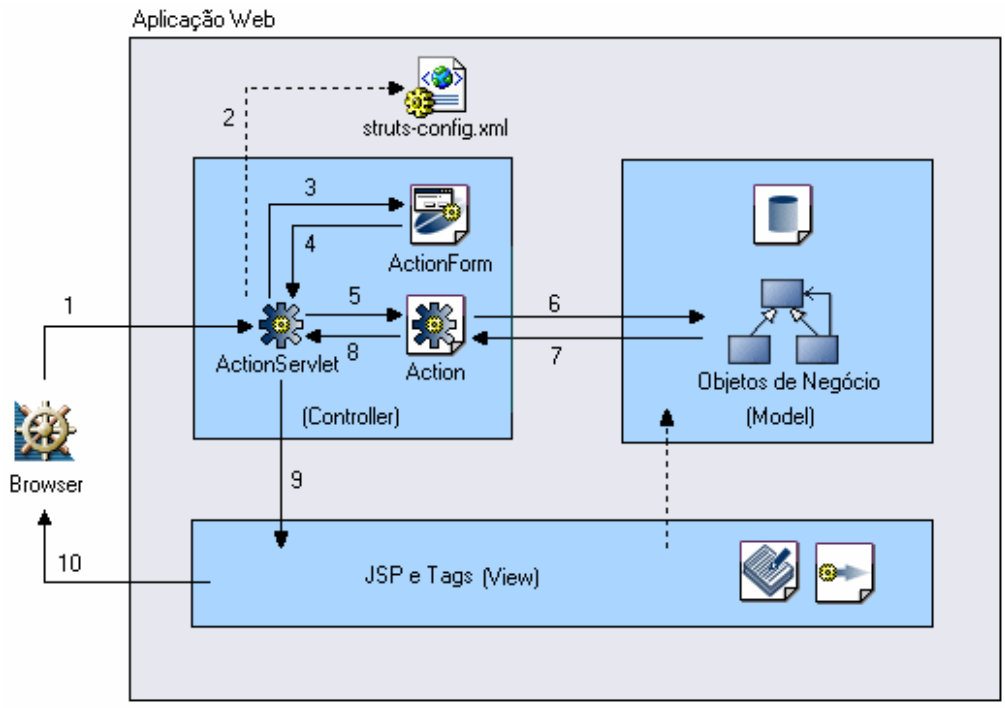
Mencione o diagrama UML mais indicado para solucionar cada problema. Justifique.

Espaço destinado à Resposta

Fim do espaço destinado à Resposta

Rascunho

3. A figura abaixo representa o fluxo de uma requisição utilizando-se o *framework* MVC Struts 1.X.



Disponível em http://www.jeebrasil.com.br/jsp/artigos/artigo_0011_jsp/struts.gif

Por intermédio dos números que constam na figura, explicite as ações representadas pelas setas do gráfico acima.

Espaço destinado à Resposta

Mais espaço para resposta na folha seguinte.

Rascunho

Continuação do espaço para Resposta à Questão 3.

Fim do espaço destinado à Resposta da Questão 3

Rascunho

Objetiva ⇨ Conhecimentos Específicos ⇨ 01 a 30

Os dados a seguir, que representam as informações de estoque do banco de dados (PostgreSQL) de uma empresa, serão usados como base para as questões 1 e 2:

Fornecedores

id	nome	Cidade
1	A	Natal
2	B	Parnamirim
3	C	Natal
4	D	Mossoró
5	E	Mossoró

Pecas

id	nome	cor	valor
1	U	Vermelha	2
2	V	Azul	1
3	W	Verde	2
4	X	Azul	5
5	Y	Vermelha	2
6	Z	Vermelha	2

Estoque

fornecedor_id	peca_id	quantidade
1	1	300
1	2	200
1	3	100
1	4	400
1	5	250
1	6	300
2	1	100
2	2	400
3	2	200
4	2	200
4	4	300
4	5	100

01. A consulta "**select f.nome, sum(p.valor * e.quantidade) as total from Fornecedores f, Pecas p, Estoque e where f.id = e.fornecedor_id and p.id = e.peca_id group by f.nome having sum(p.valor * e.quantidade) > 500 order by total asc** " teve como retorno os seguintes dados:

A)

B	200
C	200
D	200
A	600

B)

B	600
D	1900
A	4100

C)

A	1500
B	500
D	600

D)

C	200
B	600
D	1900
A	4100

02. Utilizando-se os dados das tabelas, realizou-se uma consulta cujo resultado foi:

A	U	300
A	Y	250
A	Z	300
B	U	100
D	Y	100

A consulta realizada foi:

A) select f.nome, p.nome, e.quantidade from Fornecedores f, Pecas p, Estoque e where f.id = e.fornecedor_id and p.id = e.peca_id having e.quantidade < 300 and p.cor = 'Vermelha'

B) select f.nome, p.nome, e.quantidade from Fornecedores f, Pecas p, Estoque e where f.id = e.fornecedor_id and p.id = e.peca_id and e.quantidade < 400

C) select f.nome, p.nome, e.quantidade from Fornecedores f, Pecas p, Estoque e where f.id = e.fornecedor_id and p.id = e.peca_id and p.valor < 2

D) select f.nome, p.nome, e.quantidade from Fornecedores f, Pecas p, Estoque e where f.id = e.fornecedor_id and p.id = e.peca_id and p.cor = 'Vermelha'

03. Considere o fragmento textual abaixo, extraído do livro *UML Distilled*, de Martin Fowler:

> "[...] captura o comportamento de um cenário único. O diagrama mostra um número de objetos de exemplo e as mensagens que são passadas entre esses objetos dentro do caso de uso."

No fragmento textual, Fowler se refere ao diagrama:

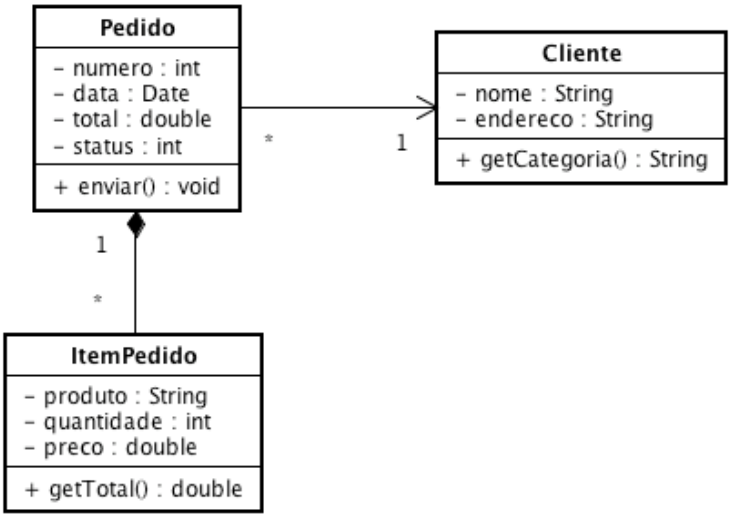
A) de casos de uso

B) de classes

C) de seqüência

D) de estado

04. O diagrama de classes abaixo foi inserido em uma ferramenta CASE, para que fosse feita a geração de código fonte Java correspondente.



De acordo com esse diagrama, seriam opções de código semanticamente corretos para as classes Pedido, ItemPedido e Cliente, **respectivamente**:

A)

```
public class Pedido {

 private int numero;
 private Date data;
 private double total;
 private int status;
 private Cliente cliente;
 private List<ItemPedido>
itens;

 public void enviar() {
   // ...
 }

}
```

```
public class ItemPedido {

 private String produto;
 private int quantidade;
 private double preco;
 private Pedido pedido;

 public double getTotal() {
   return 0;
 }

}
```

```
public class Cliente {

 private String nome;
 private String
endereco;


 public String

getCategoria(){
   return null;
 }

}
```

B)

```
public class Pedido {

 int numero;
 Date data;
 double total;
 int status;
 Cliente cliente;
 List<ItemPedido> itens;

 void enviar() {
   // ...
 }

}
```

```
public class ItemPedido {

 String produto;
 int quantidade;
 double preco;
 Pedido pedido;

 double getTotal() {
   return 0;
 }

}
```

```
public class Cliente {

 String nome;
 String endereco;
 List<Pedido> pedidos;

 String
getCategoria(){
   return null;
 }

}
```

C)

```
public class Pedido {

 private int numero;
 private Date data;
 private double total;
 private int status;
 private Cliente cliente;
 private ItemPedido
                   itens;

 public void enviar() {
   // ...
 }

}
```

```
public class ItemPedido {

 private String produto;
 private int quantidade;
 private double preco;
 private Pedido pedido;

 public double getTotal() {
   return 0;
 }

}
```

```
public class Cliente {

 private String nome;
 private String
endereco;
 private Pedido
pedido;

 public String

getCategoria(){
   return null;
 }

}
```

D)

```
public class Pedido {

 private int numero;
 private Date data;
 private double total;
 private int status;

 public void enviar() {
   // ...
 }

}
```

```
public class ItemPedido {

 private String produto;
 private int quantidade;
 private double preco;

 public double getTotal() {
   return 0;
 }

}
```

```
public class Cliente {

 private String nome;
 private String
endereco;

 public String
getCategoria(){
   return null;
 }

}
```

05. Considere o seguinte trecho de código:

```
TreeSet numeros = new TreeSet();
numeros.add(“um“);
numeros.add(“dois“);
numeros.add(“tres“);
numeros.add(“quatro“);
numeros.add(“um“);
for (Iterator it = numeros.iterator(); it.hasNext(); ) {
    System.out.print( it.next() + “ “ );
}
```

Considere também que as classes TreeSet e Iterator fazem parte do pacote java.util.

Nesse caso, após a execução do programa, a saída será:

A) um tres quatro dois um

B) um dois tres quatro um

C) quatro tres dois um

D) dois quatro tres um

06. Sobre o tratamento de exceções em Java, afirma-se:

I	Uma cláusula *try* tem que estar acompanhada das cláusulas *catch* e *finally*.
II	Uma cláusula *try* pode ser usada com uma cláusula *finally* e sem uma cláusula *catch*.
III	As *unchecked exceptions* estendem java.lang.RuntimeException e o seu tratamento não é obrigatório.
IV	As *checked exceptions* forçam o programador a tratar as condições excepcionais através do uso da cláusula *finally*.
V	NullPointerException é uma *unchecked exception*, enquanto que SQLException é uma *checked exception*.

Dentre essas afirmações estão corretas apenas:

A) I, II e IV **B)** II, III e V **C)** II, III e IV **D)** I, III e V

07. Em relação à Programação Orientada a Objetos em Java, afirma-se:

I	Todo objeto é instância de alguma classe.
II	Uma interface pode estender mais de uma interface.
III	Um atributo com visibilidade *package* (ou *default*) pode ser acessado por subclasses de qualquer pacote.
IV	A herança é uma relação do tipo “é-um” entre dois objetos e é a base para a utilização do polimorfismo.
V	Nenhuma forma de *herança múltipla* é permitida em Java

Dentre essas afirmativas, estão corretas apenas:

A) II, III e IV **C)** I, II e IV

B) I, III e V **D)** I, IV e V

08. Sobre o mapeamento de atributos e associações no Hibernate é correto afirmar:

A) É possível realizar-se o mapeamento de coleções (*one-to-many*) utilizando diferentes tipos de estruturas de dados, por exemplo: Set e List.

B) Para se mapear uma associação do tipo *many-to-many* é necessário criar-se uma classe que represente a entidade associativa no banco de dados.

C) O valor padrão para a configuração *lazy* das associações é *false*.

D) O mapeamento da chave primária de uma entidade não é obrigatório, desde que exista na entidade um atributo cujo valor seja único.

09. Considere o trecho de código HTML abaixo:

```
<div class="contents">
   <p> Parágrafo 1</p>
   <div id="contents">
      <p>Parágrafo 2</p>
      <p>Parágrafo 3</p>
   </div>
  <p>Parágrafo 4</p>
</div>
```

Se o estilo #contents { color: #FF0000; } for aplicado a esse HTML, ficarão com a cor vermelha os seguintes parágrafos:

A) 2 e 3

B) 1, 2, 3 e 4

C) 1 e 4

D) 1, 2 e 3

10. No *framework Struts*, se uma *Action* estende *DispatchAction*, então:

A) Ela possui múltiplos métodos com os mesmos parâmetros do execute(). O método a ser chamado dependerá de um mapeamento feito no *struts-config.xml*.

B) Só é permitida a existência de um único método *execute*(), mas este poderá manipular diferentes *ActionForms.*

C) É possível ter-se, dentro da classe, múltiplos métodos execute() com parâmetros diferentes. O método *execute*() a ser chamado será especificado por um parâmetro em *request*.

D) É possível ter-se, dentro da classe, múltiplos métodos com os mesmos parâmetros do método *execute*(). O método a ser chamado será especificado por um parâmetro em *request*.

11. Sobre o desenvolvimento de relatórios com o JasperReports e o iReport, é correto afirmar:

A) Os dados pertencentes à banda Page Header aparecem apenas no topo da primeira página do relatório.

B) Os parâmetros de um relatório devem ser armazenados em um Map e passados para o método fillReport() da classe JasperFillManager.

C) Para informar o número de cada página no relatório, deve-se usar a variável PAGE_COUNT.

D) O atributo *name* dos elementos Field de um relatório representa o nome do objeto do *datasource* que se deseja exibir.

12. Considere o Filtro de Servlet implementado no código abaixo:

```
public class FiltroExemplo implements Filter{

  public void init(FilterConfig config) throws ServletException{

  }

  public void doFilter(ServletRequest req, ServletResponse res,
                       FilterChain chain) throws ServletException, IOException{

    HttpServletRequest request = (HttpServletRequest) req;
    HttpServletResponse response = (HttpServletResponse) res;

    if (“true“.equals(req.getParameter(“redirect“))) {
       response.sendRedirect(“index.jsp“);
    } else {
      chain.doFilter(req,res);
    }
  }

  public void destroy(){

  }
}
```

Esse filtro foi configurado de forma a interceptar qualquer URL da aplicação. Se a URL **`/pagina1.jsp`** for chamada com o parâmetro *redirect* tendo valor “*false*”, a resposta do servidor será:

A) o conteúdo enviado pelo arquivo **`index.jsp`**.

B) vazia. Nenhum dado será enviado pelo servidor.

C) o conteúdo enviado pelo arquivo **`pagina1.jsp`**.

D) uma solicitação de usuário e senha feita pelo *browser.*

13. Considere o código abaixo:

```
<jsp:useBean id="pessoa" scope="session" class="bean.Pessoa"/>
<jsp:setProperty name="pessoa" property="*"/>
```

A utilização dessas *tags* em uma página JSP provoca:

A) a utilização de um *bean* da classe Pessoa em sessão, se ele existir, ou a criação de um novo *bean* em sessão. Em seguida, todos os seus atributos são populados com os seus valores padrão.

B) a criação de um *bean* da classe Pessoa que será acessível durante um único *request* e a população de todos os seus atributos com parâmetros de *request*.

C) a utilização de um *bean* da classe Pessoa da sessão, se ele existir, ou a criação de um novo *bean* na sessão. Em seguida, todos os seus atributos são populados com parâmetros de *request*.

D) um erro, pois não existe um atributo chamado “*” na classe Pessoa.

14. Sobre os Enterprise JavaBeans, é correto afirmar que os Stateless Session Beans

A) são compartilhados entre vários clientes.

B) podem ser passivados.

C) não suportam transações.

D) guardam estado entre transações.

15. O comando **`du -sm eclipse/`** faz retornar

A) o caminho completo do diretório eclipse.

B) as permissões de acesso (leitura) ao diretório eclipse.

C) o número total de arquivos contidos no diretório eclipse

D) o espaço total (em *megabytes*) do diretório eclipse.

16. Sobre a programação de Scripts Shell utilizando-se **BASH**, é correto afirmar:

A) O fato de não ser possível utilizar funções obriga a utilizar-se o comando **`goto`**.

B) Um exemplo de sintaxe válida para o comando **se** é **`if [ $a -lt $b ] then`**.

C) Um exemplo de sintaxe válida para o comando **se** é **`if ( #variavel = “2” ) then`**.

D) A variável especial $$ faz retornar o PID do processo pai.

17. Sobre a codificação de caracteres (Character Set) utilizados em bancos de dados PostgreSQL, é correto afirmar:

A) PT_BR é a codificação de caracteres padrão na instalação em português.

B) SQL_ASCII é recomendado para a língua portuguesa, pois suporta o armazenamento de todos os caracteres usados nesta língua.

C) UTF8 é o equivalente a US-ASCII, armazenando apenas os caracteres da tabela ASCII.

D) LATIN9 é recomendado para a língua portuguesa, pois suporta o armazenamento de todos os caracteres usados nesta língua.

18. *Tablespace*, no PostgreSQL, é:

A) um recurso que possibilita a tabela do PostgreSQL não ter limitação de tamanho.

B) um recurso usado na instalação do banco de dados que permite compactação das tabelas armazenadas.

C) uma funcionalidade que permite ao administrador de banco de dados distribuir tabelas ou índices em discos diferentes.

D) uma funcionalidade que permite *cluster* ativo das tabelas de bancos de dados.

19. Sobre a utilização de índices no PostgreSQL, é correto afirmar:

A) Ao utilizar-se o comando **`CREATE INDEX`** sem se especificar o tipo de índice o PostgreSQL cria um índice *b-tree*, por padrão.

B) Ao utilizar-se o comando **`CREATE INDEX`** sem se especificar o tipo de índice o PostgreSQL cria um índice *hash*, por padrão.

C) As versões do PostgreSQL 8 não dão suporte à utilização dos índices multi-colunas.

D) O comando **`CREATE UNIQUE INDEX indice_funcao ON pessoa (lower(sobrenome))`** reporta um erro, devido à impossibilidade de criar índices de funções únicos.

20. As ferramentas de verificação de rotas (*tracert* ou *traceroute*) são extensivamente utilizadas pelos administradores de rede para descobrirem o caminho que um determinado datagrama IP trafega até o seu destino. Essa ferramenta

A) utiliza um serviço *tracert* que executa nos roteadores de borda e de núcleo.

B) utiliza mensagens ICMP, através da manipulação do campo TTL do datagrama IP.

C) capta a informação do tráfego na resposta do datagrama IP fornecida pelo destino.

D) capta esta informação através de uma consulta a um repositório (Ex: registro.br)

21. Sobre o protocolo HTTP (Hypertext Transfer Protocol), é correto afirmar:

A) A utilização do método **POST** em formulários realiza transferência dos dados com confidencialidade.

B) O *request header* **host** viabiliza a utilização de servidores com vários *sites* hospedados.

C) O *request header* **Accept-Encoding** é utilizado para verificar se o *browser* suporta compressão.

D) O *cookie* é enviado para o servidor sempre que tiver havido um **Cookie-Request** por parte do servidor.

22. Sobre o diagrama de casos de uso da UML, é correto afirmar:

A) O relacionamento de herança pode ser aplicado entre dois atores.

B) Apenas usuários do sistema podem ser atores.

C) O relacionamento de inclusão pode ser aplicado entre dois atores.

D) Um relacionamento de <<extend>> denota o fluxo básico do caso de uso.

23. Sobre o IO com a plataforma Java, é correto afirmar:

A) O **DataInputStream** é *um low-level input stream* capaz de ler *bytes* diretamente de qualquer dispositivo.

B) Através do método **readLine()** da classe **FileInputStream**, é possível lerem-se linhas de arquivos texto.

C) A classe **java.io.InputStream** é capaz de ler pontos flutuantes, através do método **readDouble().**

D) A classe **java.io.File** pode ser usada também para representar diretórios.

24. Considere o código Java abaixo:

```
Cliente c = new Cliente();
Pessoa p = new Pessoa();
Pessoa p2 = p;
int x  = 10;
```

Ao final da execução desse código a quantidade de elementos que estão em memória, na pilha e no *heap* são, **respectivamente**:

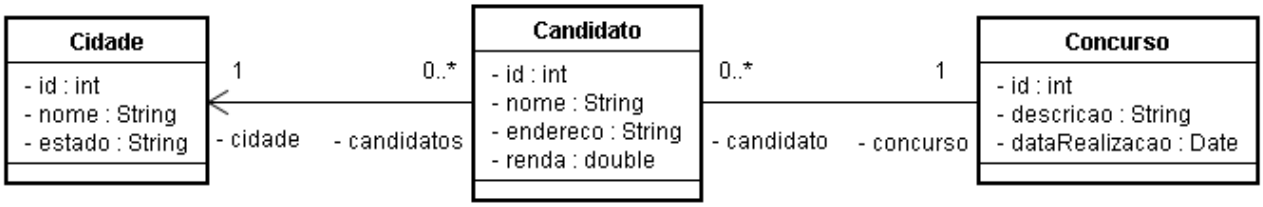
A) 2 e 4

B) 3 e 3

C) 4 e 2

D) 3 e 2

25. Considere que as classes mostradas no diagrama UML abaixo são entidades mapeadas através do Hibernate.



Caso todos os atributos e relacionamentos abaixo estejam mapeados, pode-se afirmar que a consulta HQL

A) **`from Cidade c where c.candidatos.renda > 5000`** é inválida, por operar em um relacionamento OneToMany

B) **`from Candidato c where c.cidade.estado.value = 'RN'`** faz retornar todos os candidatos do RN

C) **`select new Candidato(id, nome) from Candidato c where c.concurso = 10`** funciona independentemente de um novo construtor em candidato (além do construtor padrão)

D) **`select count(c.id), c.concurso.descricao from Candidato c group by c.concurso.descricao`** faz retornar o total de candidatos por concurso.

26. Sobre a utilização do *framework* de desenvolvimento WEB Struts, é correto afirmar:

A) A classe `org.apache.struts.actions.DispatchAction` permite a adição de várias ações em uma única classe *action*.

B) Em uma arquitetura MVC (Model-View-Controller), as classes que herdam `org.apache.struts.action.Action` são responsáveis pela implementação da regra de negócio.

C) A classe *ActionServlet* é final e não pode ser herdada, por questões de segurança do controlador.

D) O controlador *Struts* é sempre ativado pela extensão .do na URL.

27. O servidor JBoss é um dos mais utilizados pela comunidade Java para servir aplicações Java Enterprise Edition.

Esse servidor **NÃO** suporta

A) a utilização de Enterprise JavaBeans sem necessidade de se instalar um pacote adicional.

B) a replicação de dados gravados no sistema de arquivos.

C) a arquitetura Micro-Kernel utilizando-se a tecnologia JMX (*Java Management Extensions*).

D) o *clustering* de EJBs Session Beans e de sessões WEB.

28. Sobre a área de testes de *software* é correto afirmar:

A) Os testes estruturais são mais adequados para se avaliarem as interfaces gráficas com o usuário em sistemas WEB.

B) Os testes estruturais podem ajudar a melhorar a qualidade de código.

C) Os testes unitários são também denominados testes funcionais de unidade.

D) Os testes de aceitação devem ser realizados pelos analistas de testes da equipe de *software*.

29. Em relação à programação orientada a objetos com Java, afirma-se:

I	O polimorfismo permite que métodos de uma superclasse sejam redefinidos em suas subclasses.
II	Uma classe somente pode ter mais de um método construtor se eles possuírem nomes diferentes.
III	Um objeto de uma subclasse sempre é um objeto da superclasse correspondente; o inverso nunca é verdadeiro.
IV	Quando um método é sobrescrito só é possível alterar-se o seu tipo de retorno.
V	É possível descobrir os métodos e atributos de uma classe através da API Reflection.

Dentre as afirmativas acima, estão corretas apenas:

A) I, III e IV

B) II, IV e V

C) I, III e V

D) I e IV

30. Sobre o modelo **CMM (Capability Maturity Model),** é correto afirmar:

A) O nível 4 (**gerenciado**) define um modelo de métricas precisas para controlar os esforços de desenvolvimento de *software.*

B) O nível de maturidade 3 (**repetível**) define um ambiente que utiliza ferramentas de gerência de *software* para mapear custos e prazos.

C) O nível de maturidade 1 (**inicial**) pode ser aplicado a instituições que possuem uma gerência implantada, sendo esse o fator responsável pelo sucesso dos projetos.

D) O **CMMI** (Capability Maturity Model Integration) possui doze níveis de maturidade na sua representação contínua.