CONHECIMENTO ESPECÍFICO

26. Considere o trecho de código a seguir.

```
public class List<E>
{
        public E head;
        public List<E> tail;

        public List(E h, List<E> t)
        {
     if ( h == null )
        throw new Error();

                head = h;
                tail = t;
        }

   public int size()
   {
      if ( tail == null )
         return 1;

      return 1 + tail.size();
   }
}
```

Após a leitura, assinale a alternativa verdadeira.

(A) A linguagem de programação orientada a objetos *Java* permite métodos genéricos, mas não classes genéricas.
(B) As informações entre '<' e '>' informam que quando classes desse tipo são utilizadas, devemos fornecer um ou mais nomes de tipo dentro de '<' e '>' para completar a definição.
(C) O *"E"*, entre '<' e '>', é o nome de um objeto anteriormente definido.
(D) Na definição de classes genéricas, o nome entre '<' e '>' deve ser sempre *E*.
(E) O uso de um nome de tipo, tal como List<String> ou List<Integer>, define a classe.

27. Considerando *x* um número pertencente ao conjunto dos números reais, sendo ρ(*x*) o piso de *x* (ou seja, o maior inteiro não excedendo *x*), observe a seguinte função recursiva.

```
função Recursiva(y: inteiro, z: inteiro)
  se z  = 0 então
    retorne 0;
  senão
    se z é ímpar então
      retorne Recursiva(2*y, ρ(z/2)) + y;
    senão
      retorne Recursiva(2*y, ρ(z/2));
fim_função
```

Qual será o retorno dessa função, na sua primeira invocação, com y igual a 4 e z igual 3?

(A) 6.
(B) 9.
(C) 12.
(D) 15.
(E) 16.

28. Sobre conceitos envolvendo técnicas de orientação a objetos, assinale a alternativa correta.

(A) Uma classe interna é declarada dentro de outra classe, e ambas são visíveis em qualquer lugar do programa.
(B) Classes são fábricas de objetos construídos, em *Java,* com o operador *nil*.
(C) Herança é um mecanismo para estender classes já existentes acrescentando métodos e atributos.
(D) Em um programa, consideram-se atributos aquelas entidades manipuladas pela invocação de métodos.
(E) Para sinalizar uma situação excepcional em um programa *Java*, usa-se o comando *catch* para disparar um objeto de exceção.

29. Leia o código a seguir.

```
public class SuperClasse
{
   ...
}

public class SubClasse1 extends SuperClasse
{
   ...
}

public class SubClasse2 extends SuperClasse
{
   ...
}

public class Teste
{
   public static void main(String args[])
   {
      SuperClasse o1 = new SubClasse1();
      SuperClasse o2 = new SubClasse1();
   …
   }
   …
}
```

Considerando os conceitos de orientação a objetos, o trecho de código apresentado ilustra o uso exatamente de

(A) herança múltipla, que é a capacidade de duas classes herdarem atributos e métodos de outra classe.
(B) encapsulamento, que é a capacidade de ocultar dados do objeto e fornecer métodos, qualquer que seja o modificador, para acessar esses dados.
(C) abstração de dados, que é o processo de encontrar um conjunto de classes abstratas necessárias para um programa.
(D) objetos, que possuem atributos e métodos de classes diferentes daquela da definição do objeto.
(E) polimorfismo, que indica que o comportamento pode variar, dependendo do tipo real de um objeto.

30. A função a seguir efetua a multiplicação entre duas matrizes – $A_{m \times p}$ e $B_{q \times n}$ –, resultando em uma matriz $C_{m \times n}$.

```
função MultiplicaMatriz(A[1..m][1..p], B[1..q][1..n]): C[1..m][1..n]
  se p < > q então
    escreva "dimensões incompatíveis.";
  senão
    para i de 1 até m faça
      ----------------------
    fim_para
  retorne C;
fim_função
```

A parte que falta para completar a estrutura de repetição "para" é?

(A)
```
para j de 1 até q faça
        C[i][j] ← A[i][j] * B[i][j];
fim_para
```

(B)
```
para j de 1 até n faça
        para k de 1 até p faça
          C[i][j] ← A[i][k] * B[k][j];
        fim_para
fim_para
```

(C)
```
C[i][i] ← A[i][i] * B[i][i];
```

(D)
```
para j de 1 até n faça
        C[i][j] ← 0;
        para k de 1 até p faça
          C[i][j] ← C[i][j] + A[i][k] * B[k][j];
        fim_para
 fim_para
```

(E)
```
para k de 1 até n faça
        C[i][j] ← 0;
        para j de 1 até p faça
          C[i][k] ← C[i][j] + A[i][k] * B[k][j];
        fim_para
 fim_para
```

31. Assinale a alternativa que apresenta a vantagem na construção de algoritmos compostos por módulos, ou seja, a construção de algoritmos por meio da modularização.

(A) Economiza espaço, tempo e esforço. Frequentemente, é necessário executar uma mesma tarefa em vários lugares de um mesmo algoritmo. Uma vez que um módulo foi escrito, ele pode ser invocado quantas vezes for preciso.

(B) Eleva o nível de abstração. É possível entender o que um algoritmo faz por saber apenas o que seus módulos fazem. No entanto, existe necessidade de entender, detalhadamente, as particularidades internas aos módulos.

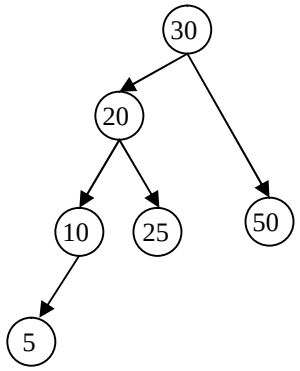
(C) Torna o algoritmo mais fácil de ler. Dividido em módulos, ele permite que alguém, que não seja o seu autor, possa entendê-lo de maneira muito mais rápida

(D) Estende a linguagem. Uma vez tendo sido criado um módulo, ele não poderá ser utilizado em outros algoritmos, a não ser que seu código seja reescrito.

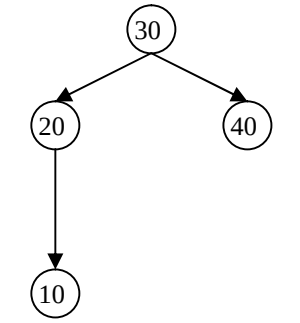
(E) Torna o algoritmo mais fácil de escrever. O desenvolvedor pode focalizar todo o problema complicado e escrever a solução, uma única vez.

32. Considere as seguintes árvores binárias de busca.

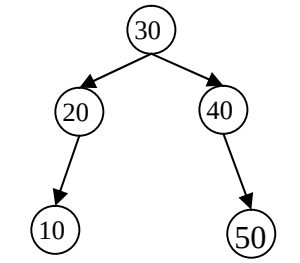
I



II



III



Assinale a alternativa que apresenta árvores balanceadas AVL?

(A) I e II, apenas.
(B) I, II e III.
(C) I e III, apenas.
(D) III, apenas.
(E) II e III, apenas.

33. O *Scrum* é um *framework* para desenvolvimento. Assinale, dentre as afirmações a seguir, a alternativa que NÃO se aplica ao *Scrum*.

(A) Esse *framework* não pode ser aplicado a pequenos projetos de *software*, uma vez que foi desenvolvido para ser utilizado por grandes equipes.

(B) Clientes se tornam parte da equipe de desenvolvimento do projeto.

(C) Discussões diárias de *status* são realizadas com a equipe de desenvolvimento do projeto.

(D) Reuniões frequentes com os *stakeholders* são realizadas para monitorar o progresso do projeto.

(E) *Scrum* é um *framework* para desenvolvimento ágil e Gerenciamento de Projetos.

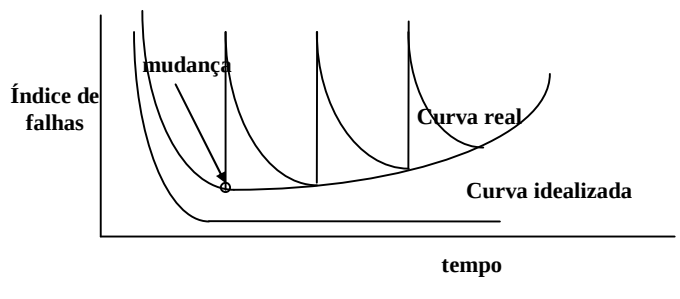
34. Observe as seguintes considerações: 1ª) as operações de inserção e remoção de elementos do tipo inteiro, em uma fila circular, são, respectivamente ENQUEUE(Q, x) e DEQUEUE(Q), onde Q é a fila, e x, um elemento a ser inserido ou removido da fila; 2ª) toda fila tem início e fim; 3ª) após a operação DEQUEUE(Q), a posição onde estava o elemento retirado da fila torna-se vazia. Sobre uma fila Q, inicialmente vazia no vetor Q[1..5], assinale alternativa que resulta da seguinte sequência de operações: ENQUEUE(Q, 3), ENQUEUE(Q, 8), ENQUEUE(Q, 7), DEQUEUE(Q), ENQUEUE(Q, 1), DEQUEUE(Q), ENQUEUE(Q, 6), ENQUEUE(Q, 5).

(A) Q[1] = 5, Q[2] = vazio, Q[3] = 7, Q[4] = 1 e Q[5] = 6.
(B) Q[1] = 7, Q[2] = 1, Q[3] = 6, Q[4] = 5 e Q[5] = vazio.
(C) Q[1] = 5, Q[2] = 7, Q[3] = 1, Q[4] = 6 e Q[5] = vazio.
(D) Q[1] = vazio, Q[2] = 5, Q[3] = 7, Q[4] = 1 e Q[5] = 6.
(E) Q[1] = vazio, Q[2] = 7, Q[3] = 1, Q[4] = 6 e Q[5] = 5.

35. A *Extreme Programming* (XP) é talvez o mais conhecido e mais amplamente usado dos métodos ágeis. O nome se deve ao fato de que a abordagem foi desenvolvida pelo avanço do desenvolvimento iterativo e o envolvimento do cliente em níveis "extremos". Dentro desse contexto, assinale a alternativa correta.

(A) Esse método também é conhecido como "cascata", uma vez que suas fases são sequenciais.
(B) Todos os requisitos são expressos em cenários (chamados histórias do usuário), que são implementados diretamente como uma série de tarefas.
(C) Aplicando esse método, por ser ágil, não se aplica a atividade de teste de *software*.
(D) Não existe nenhum método de desenvolvimento ágil chamado XP, sendo essa nomenclatura usada para referenciar um modelo/versão de sistema operacional.
(E) Esse método utiliza a prática de trabalho chamada "Programação em Pares", que foca o desenvolvimento paralelo de dois programas, ou seja, o desenvolvedor irá trabalhar sempre com duas aplicações em paralelo.

36. Observe a figura a seguir.



CURVA DE FALHAS DO SOFTWARE

Com base nessa curva, assinale a alternativa verdadeira.

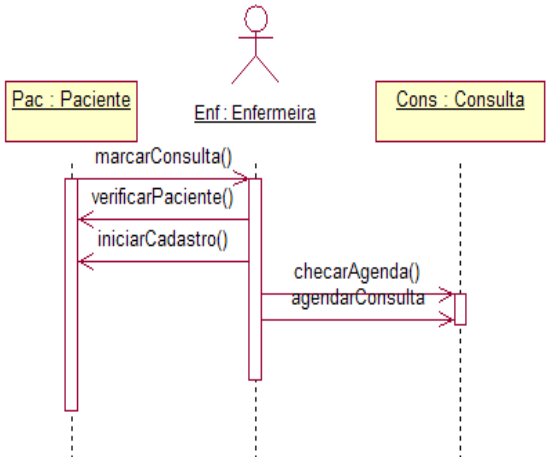
(A) A curva de falhas do *software* é idêntica à curva de falhas do *hardware*, uma vez que ambos são manufaturados no sentido clássico.
(B) A curva idealizada corresponde realmente ao que acontece na maioria dos *softwares* desenvolvidos utilizando-se técnicas de Análise Estruturada.
(C) A curva idealizada corresponde realmente ao que acontece na maioria dos *softwares* desenvolvidos utilizando-se técnicas de Análise Orientadas a Objetos.
(D) A cada alteração efetuada no software pode-se estar incluindo novos erros, com isso, o software não se desgasta como o hardware mais se deteriora em função do tempo.
(E) A cada alteração efetuada, o sistema acusa a inclusão de novos dados na Estrutura.

37. Em um Diagrama de Classe, existem relacionamentos que interligam as classes/objetos entre si, criando relações lógicas. A partir disso, analise a figura a seguir.

Assinale a alternativa correta em relação ao que foi demonstrado.

(A) A classe "Pessoa", ao se relacionar com a classe "Município", herda suas características. Em outros termos, a classe "Pessoa" possui os mesmos atributos e métodos da classe "Município".
(B) A classe "Pessoa" e a classe "Município" mantêm um relacionamento de associação, onde 1 pessoa reside em 1 município, e em 1 município reside 0 ou mais pessoas.
(C) A classe "Município" é uma classe de interface, pois possui apenas um atributo e não se relaciona com a classe "Pessoa".
(D) A classe "Município" é uma classe abstrata, uma vez que possui apenas um atributo relacionado à classe "Pessoa".
(E) A classe "Pessoa" é uma superclasse da classe "Município", com esta se relacionando.

38. Analise a figura a seguir.



Assinale a alternativa correta em relação ao diagrama apresentado.

(A) A classe “Enfermeira” possui os seguintes métodos: verificarPaciente(), iniciarCadastro(), checarAgenda() e agendarConsulta().
(B) A classe “Consulta” não irá possuir nenhum método.
(C) A classe “Paciente” possui os seguintes métodos: verificarPaciente() e iniciarCadastro().
(D) O objeto “Cons” é do tipo “Consulta”, tratando-se de uma classe do tipo *ator*.
(E) Representa um diagrama de colaboração, que mostra a troca de mensagem entre os objetos.

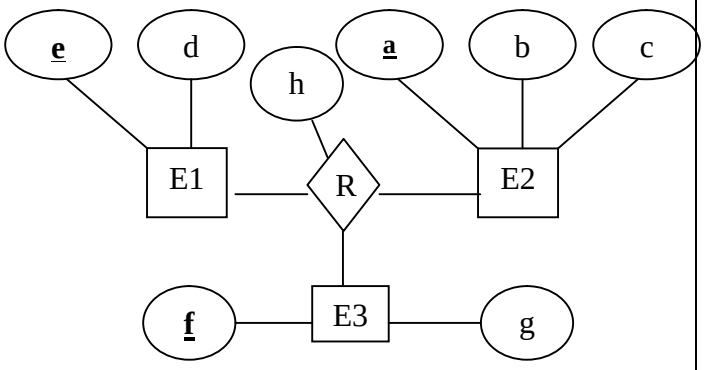
39. Em um Modelo Entidade-Relacionamento, considere um conjunto de entidades PessoaFisica com os seguintes atributos:

PessoaFisica = {nome, sobrenome, altura, peso, cpf, rg, email}

Sabe-se que os atributos são atômicos, monovalorados e não podem ser nulos, ou seja, são de preenchimento obrigatório. Considere que cada um dos atributos {cpf}, {rg} e {email} é único para cada pessoa física. Assinale a alternativa correta, em relação aos conceitos de superchave, chave, chave candidata e chave primária.

(A) Não existe superchave no conjunto de entidades PessoaFisica.
(B) Existem três chaves no conjunto de entidades PessoaFisica: {cpf}, {rg} e {email}. Cada uma delas também é uma chave candidata, sendo que, dentre elas, uma será escolhida como chave primária.
(C) O atributo {email}, do conjunto de entidades PessoaFisica, é uma chave estrangeira.
(D) {cpf} e {rg} são chaves candidatas do conjunto de entidades PessoaFisica. Apesar de único, o atributo {email}, por ser alfanumérico, não pode ser considerado chave candidata.
(E) Uma superchave desse conjunto de entidades é {nome, sobrenome, altura}. Dificilmente, existirão duas ou mais pessoas físicas com esses mesmos três atributos com valores iguais.

40. Observe o seguinte Modelo Entidade-Relacionamento.



Agora, considere as seguintes informações: 1ª) E1, E2 e E3 são conjuntos de entidades; 2ª) R é um conjunto de relacionamentos; 3ª) os atributos são representados pelas letras minúsculas "a" até "h", todos atômicos e monovalorados; 4ª) as chaves primárias estão sublinhadas. Portanto, as chaves primárias de E1, E2 e E3 são, respectivamente, {e}, {a} e {f}. Assinale a alternativa correta em relação ao mapeamento do Modelo Entidade-Relacionamento para o Modelo Relacional, sendo que nenhuma nova coluna e/ou chave poderá ser criada durante o mapeamento.

(A) O conjunto de relacionamentos R é mapeado em uma tabela própria, possuindo ao total três colunas: "a", "e", "f".
(B) Por ser ternário, o conjunto de relacionamentos R não é mapeado em uma tabela própria do modelo relacional.
(C) Os conjuntos de entidades E1, E2 e E3 são mapeados em tabelas próprias. O atributo "h", do conjunto de relacionamentos R, será mapeado juntamente com qualquer uma das tabelas E1, E2 ou E3.
(D) O conjunto de relacionamentos R é mapeado em uma tabela própria contendo todos os atributos do Modelo Entidade-Relacionamento: "a", "b", "c", "d", "e", "f", "g", "h". Sua chave primária, que é a junção das chaves primárias dos conjuntos de entidades E1, E2 e E3, será constituída pelos atributos "a", "e", "f".
(E) O conjunto de relacionamentos R é mapeado em uma tabela própria, sendo que o mesmo terá quatro colunas: "a", "e", "f", "h".

41. Em um banco de dados de uma loja de departamentos, foi encontrada a relação Empregado (descrita a seguir), que tem este objetivo principal: armazenar o número de CPF (Cadastro de Pessoas Físicas), nome, data de nascimento e endereço dos empregados, além de indicar o número e o nome do departamento em que os empregados trabalham. A loja possui apenas três departamentos, sendo o departamento número 1, de nome "Cozinha"; o departamento número 2, "Eletrônicos"; e o departamento número 3, "Cama e Banho". A chave primária é {cpf}, único atributo grifado na relação Empregado, mostrada no quadro a seguir.

Empregado = {cpf, nome, data_nascimento, endereço, numero_departamento, nome_departamento}

Assinale a alternativa correta sobre normalização da relação Empregado, considerando que todos os atributos são atômicos e monovalorados.

(A) A relação Empregado está normalizada na 1FN (Primeira Forma Normal), na 2FN (Segunda Forma Normal) e também na 3FN (Terceira Forma Normal).

(B) A relação Empregado não está normalizada na 1FN (Primeira Forma Normal), uma vez que existe dependência funcional entre seus atributos.

(C) A relação Empregado está normalizada na 1FN (Primeira Forma Normal) e também na 2FN (Segunda Forma Normal). Entretanto, não está normalizada na 3FN (Terceira Forma Normal), pois existe dependência transitiva entre seus atributos.

(D) A relação Empregado está normalizada na 1FN (Primeira Forma Normal). Porém, não está normalizada na 2FN (Segunda Forma Normal), uma vez que o atributo {*nome_departamento*} depende do atributo não-chave {*numero_departamento*}.

(E) A relação Empregado não está normalizada em nenhuma das três formas normais, uma vez que {*número_departamento*} e {*nome_departamento*} deveriam fazer parte de uma outra relação, uma vez que são dependentes transitivos.

42. Considere o seguinte comando em SQL.

```
alter table (X) add (Y) (Z);
```

Três partes desse comando foram omitidas e substituídas pelos valores hipotéticos (X), (Y) e (Z). Sobre o resultado do comando supracitado, assinale a alternativa correta.

(A) Tal comando adiciona uma nova coluna de nome (Z), de um domínio específico contido em (Y), na tabela de nome (X).

(B) Tal comando modifica o domínio de uma coluna de nome (Y) para o novo domínio contido em (Z), na tabela de nome (X).

(C) Tal comando modifica o valor de todas as tuplas da coluna de nome (Y) para o valor contido em (Z), na tabela de nome (X).

(D) Tal comando adiciona duas novas colunas de nomes (Y) e (Z), sem especificar o domínio, na tabela de nome (X).

(E) Tal comando adiciona uma nova coluna de nome (Y), de um domínio específico contido em (Z), na tabela de nome (X).

43. Considere um banco de dados PostgreSQL, cujas tabelas e colunas estão representadas a seguir.

Tabela: Proprietario			
Num	**NomeProprietario**	**RG**	**Telefone**

Tabela: Carro					
Placa	**Cor**	**Modelo**	**Valor**	**NumProp**	**CodigoMontadora**

Tabela: Montadora			
Código	**Nome**	**Cidade**	**Email_suporte**

As chaves primárias das tabelas *Proprietario*, *Carro* e *Montadora* são, respectivamente, {Num}, {Placa} e {Código}, e estão representadas por um sublinhado. As colunas *NumProp* e *CodigoMontadora* da tabela Carro são chaves estrangeiras referentes às colunas *Num*, da tabela Cliente, e *Codigo*, da tabela Montadora, respectivamente. Considere que existem muitos dados na base (não representados na figura) e que todos os atributos (colunas das tabelas) são atômicos e monovalorados. Assinale a alternativa correta que contém o comando SQL, cujo resultado contenha o nome e o telefone de todos os proprietários que possuem um carro de cor 'Azul' da montadora que tem o nome de 'Fiat'.

(A) select Proprietario.NomeProprietario, Proprietario.Telefone from Proprietario, Carro, Montadora where (Proprietario.Num in (select NumProp from Carro where Cor='Azul')) AND (Carro.CodigoMontadora in (select Codigo from Montadora where Nome like 'Fiat')) AND (Carro.NumProp = Montadora.Codigo).

(B) select P.NomeProprietario, P.Telefone from Proprietario P, Carro C, Montadora M where (M.Nome = 'Fiat') AND (C.Cor = 'Azul').

(C) select NomeProprietario, Telefone from Proprietario where (Cor = 'Azul') AND (Nome = 'Fiat').

(D) select Proprietario.NomeProprietario, Proprietario.Telefone from Proprietario, Carro, Montadora where (Montadora.Nome = 'Fiat') AND (Carro.Cor = 'Azul').

(E) select P.NomeProprietario, P.Telefone from Proprietario P, Carro C, Montadora M where (P.num = C.NumProp) AND (C.CodigoMontadora = M.Codigo) AND (M.Nome = 'Fiat') AND (C.Cor = 'Azul').

44. Com relação ao código exibido logo em seguida, assinale a alternativa correta.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
 "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" >
<head>
<meta name="description" content="Informações sobre a UFGD" />
<meta name="keywords" content="UFGD, Dourados, Históriat" />
<meta http-equiv="Content-Type" content="text/html;charset=ISO-8859-1" /> <title>UFGD</title>
</head>
<body>
<h1>Universidade Federal da Grande Dourados </h1>
<p> A <strong>Universidade Federal da Grande Dourados</strong> (<strong>UFGD</strong>) é uma instituição pública que se localiza na cidade de Dourados, Mato Grosso do Sul, Brasil. </p>
<h2>História</h2>
<p>Inaugurado em <em>20 de dezembro de 1970</em>, o então <strong>Centro Pedagógico de Dourados</strong>, que inicialmente deveria abrigar o curso de Agronomia, começou a funcionar em fevereiro de <em>1971</em>, promovendo o primeiro vestibular para os cursos de Letras e Estudos Sociais (Licenciatura). </p>
<p>Com a criação da UFGD, em 1 de agosto de 2005, os Cursos existentes no Centro Pedagógico de Dourados passaram a fazer parte da nova instituição. </p>
</body>
</html>
```

(A) O código acima é considerado válido e em conformidade com os padrões da W3C (*World Wide Web Consortium*).
(B) Trata-se de um código de página Web XHTML não validado pelos padrões da W3C por não conter a primeira marcação de um documento XML, ou seja, não contém a Processing Instructions (PI). A PI identifica a versão de XML e o conjunto de caracteres utilizados no documento como, por exemplo, <?xml version="1.0" encoding="iso-8859-1"?>.
(C) O código XHTML apresentado está em conformidade com os padrões internacionais W3C, pois, ao ser submetido ao validador disponibilizado pela W3C (http://validator.w3.org/), não indicará erros (*errors*) ou alertas (*warnings*).
(D) O código apresentado faz uso de HTML, XHTML e XML, que estão entre os padrões Web definidos pelo W3C. Entretanto, ao fazer uso desses padrões em conjunto, o código acima está violando as regras do W3C, pois os mesmos são incompatíveis entre si.
(E) O código HTML apresentado não está em conformidade com os padrões internacionais W3C, pois, ao ser submetido ao validador disponibilizado pela W3C (http://validator.w3.org/), não indicará erros (*errors*), mas alertas (*warnings*) devido ao uso de etiquetas em desuso ou não recomendadas.

45. Com relação ao W3C e seus padrões, analise as sentenças apresentadas.

I. HTML, XHTML, XML e CSS estão entre os padrões web definidos pelo W3C.
II. O W3C é uma comunidade internacional que desenvolve padrões com o objetivo de garantir o crescimento da web.
III. Tanto XML, quanto XHTML e HTML são linguagens de marcação que combinam texto (dados), informação sobre o texto (semântica) e formatação (aparência) para disponibilizar informações na Internet.
IV. Entre os exemplos de padrões de acessibilidade, estão o WCAG (*Web Content Accessibility Guidelines*), recomendado pelo W3C, e o padrão brasileiro e-MAG (Modelo de Acessibilidade de Governo Eletrônico), sendo obrigatória a observância deste último nos *sites* e portais do governo brasileiro.

Assinale a alternativa que contém as informações verdadeiras.

(A) Apenas I.
(B) Apenas I e II.
(C) Apenas I, II e III.
(D) Apenas I, II e IV.
(E) Apenas I, III e IV.

46. Com relação ao desenvolvimento de aplicações Web usando Java, assinale a alternativa correta.

(A) No projeto de aplicações Web em Java, existem dois modelos de projeto comumente utilizados: Model 1 e Model 2. O Model 2 usa o padrão de projeto MVC (Model-View-Controller) para separar a apresentação do conteúdo. *Servlets* podem ser usados no Model 2, cumprindo o papel de *view*.
(B) O JSP é uma tecnologia Java para desenvolver aplicações Web e se adapta perfeitamente ao modelo de projeto Model 2, cumprindo o papel de *controller*.
(C) No Model 1, realiza-se uma solicitação (*request*) para um JSP (ou *Servlet* e JSP) que manipula todas as responsabilidades para o atendimento da solicitação. Tais responsabilidades incluem o processamento da solicitação, a validação dos dados e a manipulação da lógica de negócios, gerando uma resposta (*response*). O Model 1 é simples e normalmente utilizado em aplicações pequenas.
(D) *Enterprise Java Beans* (EJB) é um dos componentes da plataforma J2EE (*Java 2 Enterprise Edition*). É um componente servidor que é executado no *container* do servidor de aplicação. Para acessar um EJB, é necessário definir sua interface de acesso, que pode ser *Local Interface* (permite acesso de outros componentes no mesmo servidor de aplicações) ou *External Interface* (permite acesso de componentes externos ao servidor).
(E) São quatro os tipos de componentes EJBs: *Session, Entity, Message Driven* e *Container*.

47. Com base nas definições de *eXtensible Markup Language* (XML) e XML *Schema Definition* (XSD), leia o código a seguir.

```
papers.xsd
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
 <xs:element name="acceptedworks">
  <xs:complexType>
   <xs:sequence>
    <xs:element maxOccurs="10" minOccurs="2" ref="paper"/>
   </xs:sequence>
  </xs:complexType>
 </xs:element>
 <xs:element name="paper">
  <xs:complexType>
   <xs:sequence>
    <xs:element maxOccurs="unbounded" ref="author"/>
    <xs:element ref="title"/>
    <xs:element ref="abstract"/>
   </xs:sequence>
   <xs:attribute name="submitdate" use="required" type="st.date"/>
  </xs:complexType>
 </xs:element>
 <xs:simpleType name="st.date">
  <xs:restriction base="xs:string">
   <xs:pattern value="\d{2}-\d{2}-\d{4}"/>
  </xs:restriction>
 </xs:simpleType>
 <xs:element name="author">
  <xs:complexType>
   <xs:attributeGroup ref="att.person"/>
  </xs:complexType>
 </xs:element>
 <xs:attributeGroup name="att.person">
  <xs:attribute name="firstname" use="required"/>
  <xs:attribute name="lastname" use="required"/>
 </xs:attributeGroup>
 <xs:element name="title" type="xs:string"/>
 <xs:element name="abstract" type="xs:string"/>
</xs:schema>
```

```
Event1.xml
<?xml version="1.0" encoding="UTF-8"?>
<acceptedworks xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation=" papers.xsd">
<paper submitdate="15-06-2010">
<author firstname="João" lastname="Silva"/>
<author firstname="José" lastname="Souza"/>
<title>Visão geral sobre XML</title>
<abstract>Este artigo trata dos conceitos de XML. O que descrevemos aqui é tão verdade quanto 2+2=4. </abstract>
</paper>
<paper submitdate="01-06-2010">
<author firstname="Alfredo" lastname="Martins"/>
<author firstname="José" lastname="Souza"/>
<title>HTML4 x HTML5</title>
<abstract>Uma discução sobre as mudanças ocorridas com o surgimento do HTML5 em comparação com o HTML4. </abstract>
</paper>
</acceptedworks>
```

```
Event2.xml
<?xml version="1.0" encoding="UTF-8"?>
<acceptedworks xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:noNamespaceSchemaLocation=" papers.xsd">
<paper submitdate="02\12\2009">
<title>Matemática na infância.</title>
<abstract>Aplicabilidade de problemas para desenvolver o raciocínio matemático em crianças menores de 5 anos.</abstract>
</paper>
<paper submitdate="10\04\2009">
<author firstname="Patrícia" lastname="Buarque"/>
<title>Pedras Pantaneiras</title>
<abstract>Levantamento dos tipos de pedras preciosas e semi preciosas encontradas na planície pantaneira.</abstract>
</paper>
</acceptedworks>
```

A partir dessas informações, assinale a alternativa correta.

(A) Não é possível validar os arquivos XML (Event1.xml e Event2.xml), citados no código, com o esquema *papers.xsd*, por se tratar de estruturas incompatíveis entre si.

(B) Os documentos XML (Event1.xml e Event2.xml) não são documentos bem-formados, pois utilizam codificação de caracteres indevida e, portanto, são documentos inválidos.

(C) Ambos os documentos XML (Event1.xml e Event2.xml) são documentos bem-formados e válidos. Eles seguem as regras das linguagens de marcação e estão em conformidade com o esquema *papers.xsd*.

(D) Ambos os documentos XML (Event1.xml e Event2.xml) são documentos bem-formados, todavia, apenas Event2.xml é válido, uma vez que Event1.xml possui data com formato inválido.

(E) Ambos os documentos XML (Event1.xml e Event2.xml) são documentos bem-formados, porém, apenas Event1.xml é válido, uma vez que Event2.xml não está em conformidade com o esquema *papers.xsd*, porque o atributo *submitdate* não segue as regras do tipo simples *st.date* e há um elemento *paper* que não possui autor.

48. Ajax é uma tecnologia utilizada para tornar o navegador mais interativo com o usuário, utilizando solicitações assíncronas de informações. Em relação às tecnologias que compõem o Ajax e suas funções, assinale a alternativa verdadeira.

(A) Ajax utiliza linguagem *JavaScript*; XML e XSLT, para transferências dos dados; DOM, para interações dinâmicas; e os objetos *XMLHttpRequest* e *XMLHttpResponse*, para chamadas assíncronas dentro de uma apresentação XHTML/CSS ou HTML/CSS.

(B) Ajax utiliza linguagem *JavaScript*; XML, para transferências dos dados; DOM, para interações dinâmicas; e os objetos *HttpRequest* e *HttpResponse*, para chamadas assíncronas dentro de uma apresentação XHTML/CSS ou HTML/CSS.

(C) Ajax utiliza linguagem JavaScript; XML, para transferências dos dados; XSLT, para interações dinâmicas; e os objetos HttpRequest e *HttpResponse*, para chamadas assíncronas dentro de uma apresentação HTML/CSS.

(D) Ajax utiliza linguagem *JavaScript*; XML e XSD, para transferências dos dados; DTD e DOM, para interações dinâmicas; e os objetos *XMLHttpRequest* e *XMLHttpResponse*, para chamadas síncronas dentro de uma apresentação XHTML/CSS.

(E) Ajax utiliza linguagem *Java*; XML e XSD, para transferências dos dados; XSLT, para interações dinâmicas; e os objetos *XMLHttpRequest* e *XMLHttpResponse*, para chamadas síncronas dentro de uma apresentação XHTML/CSS.

49. Considerando *Hibernate* e sua linguagem de consulta, assinale a alternativa correta.

(A) O *Hibernate* é um *framework* para o mapeamento objeto-relacional, que tem por objetivo diminuir a complexidade no desenvolvimento de *softwares* baseados no paradigma orientado a objeto que necessitam utilizar banco de dados relacional. Apesar de escrito em linguagem *Java*, o *Hibernate* possui versões compatíveis com muitas linguagens de programação orientadas a objetos, como, por exemplo, C#, .Net e Ruby.

(B) A HQL (*Hibernate Query Language*) é uma linguagem de consulta para o *Hibernate*, semelhante a SQL. No entanto, a HQL é uma linguagem de consulta puramente orientada a objeto, incluindo os paradigmas de herança, polimorfismo e encapsulamento.

(C) A linguagem de consulta HQL foi incluída no *Hibernate* em substituição à EJB/JPA QL (*EJB Query Language*), que possui sintaxe muito complexa e de difícil aprendizado.

(D) No *Hibernate*, é possível realizar consultas utilizando tanto a SQL quanto a HQL. Com ambas as linguagens, as consultas podem apenas ser realizadas sobre as classes de persistência do *Java*, ao invés de tabelas no banco de dados, o que aumenta a distância entre o desenvolvimento das regras de negócio e o banco de dados.

(E) HQL é uma linguagem de consulta para o *Hibernate*, semelhante a SQL. No entanto, a HQL é uma linguagem de consulta híbrida que permite consultas relacionais e orientadas a objeto.

50. JSF (*JavaServer Faces*) é um *framework* para desenvolver aplicações Web de forma visual e ágil, que incorpora características do padrão MVC (Model-View-Controller) e de um modelo de interfaces gráficas baseado em eventos. JSF foi desenvolvido como resultado da experiência e maturidade adquiridas com o Model 1, Model 2 e *Struts*. Sobre JSF, assinale a alternativa verdadeira.

(A) O JSF possui suporte a internacionalização e acessibilidade. Tal *framework* ainda possui um conjunto padrão de componentes de interface de usuário que possibilita validação padronizada e segue um modelo de eventos do lado do cliente (*client-side event model*).

(B) Uma aplicação JSF tem como principal arquivo de configuração o faces-config.xml, que está localizado no diretório WEB-INF da aplicação e possui formato XML. Esse arquivo é responsável por definir os *servlets* utilizados e os mapeamentos (*servlet-mapping*) entre os *servlets* e as requisições que cada um deve atender (URI).

(C) O arquivo WEB-INF/web.xml é o descritor do contexto de aplicação Web, segundo a especificação *Java* Servlet/J2EE, sendo necessário, independente do uso de algum *framework*. O arquivo *web.xml* é responsável por descrever os elementos e sub-elementos que fazem parte do projeto, tais como regras de navegação, componentes gerenciados e configurações de localização.

(D) O JSF permite a inserção, por meio de uma IDE (*Integrated Development Environment*), de Folhas de estilo (CSS) e comandos em *JavaScript*, mas não possui suporte à tecnologia Ajax (*Asynchronous JavaScript and XML*).

(E) O JSF possui *BackingBeans*, que podem armazenar dados que serão mostrados no navegador e precisam ser configurados no arquivo *faces-config.xml*. Os *BackingBeans* possuem três tipos de escopo distintos, sendo: re*q*uest, *session* e *application*.

Rascunho