APLICAÇÕES DE LINGUAGEM DE PROGRAMAÇÃO ORIENTADAS À OBJETOS 7968-90_43701_R_E1_20232 CONTEÚDO

Revisar envio do teste: QUESTIONÁRIO UNIDADE III

Usuário	
Curso	APLICAÇÕES DE LINGUAGEM DE PROGRAMAÇÃO ORIENTADAS À OBJETOS
Teste	QUESTIONÁRIO UNIDADE III
Iniciado	
Enviado	
Status	Completada
Resultado da tentativa	
Tempo decorrido	
Resultados exibidos	

Pergunta 1

0,25 em 0,25 pontos

Model-View-Control, isto é, MVC, corresponde a um padrão de arquitetura de *software* que tem como objetivo separar as lógicas de negócio e de apresentação de modo que se permita o desenvolvimento, o teste e a manutenção de ambos de forma isolada. Sabe-se que o *model* (modelo) é utilizado para a gerência da informação (camada de persistência), enquanto o *view* (visão) apresenta os dados de maneira adequada aos utilizadores (usuários) e o *controller*
(controle) é responsável, dentre outras coisas, pela validação e filtragem da entrada de dados, ou seja, é a camada que concentra a lógica da aplicação. A partir de tais exposições, pode-se aprofundar as explicações acerca do MVC afirmando corretamente que:

Resposta Selecionada: b.
No MVC, a camada de visão gerencia o conteúdo de apresentação, definindo como os dados devem ser apresentados ao usuário, mas também encaminhando as ações do usuário para a camada controladora que poderá validar os dados.

Respostas:

a.
Na arquitetura MVC, a camada do modelo representa tanto os dados da aplicação, quanto as regras do negócio que orientam o acesso e a modificação deles. Portanto, o modelo mantém o controle das funcionalidades gerais e apresenta os dados na camada de visualização.

b.
No MVC, a camada de visão gerencia o conteúdo de apresentação, definindo como os dados devem ser apresentados ao usuário, mas também encaminhando as ações do usuário para a camada controladora que poderá validar os dados.

c.
A camada de controle da aplicação se preocupa com a exibição da informação, isto é, como e onde ela será mostrada, e quais recursos gráficos serão utilizados.

d.
Considerando-se a arquitetura de três camadas do modelo MVC, entende-se que a persistência da informação deve ficar junto com o controle e com a validação dos dados.

e.
Na camada de controle centraliza-se o armazenamento, leitura e geração de dados de um Banco de Dados e, por isso, utiliza-se a camada de apresentação como intermediária com a camada lógica.

Comentário da resposta: Resposta: B
Comentário: vimos que pelo Pattern MVC, para que possamos organizar melhor o sistema a ser construído, divide-se (organiza-se) o projeto em camadas, a fim de melhorar tanto o processo de criação, quanto o de manutenção do sistema. São três as camadas que trata esse Pattern MVC, como as camadas "model", "view" e "controller" de forma que:
- A camada "model" é onde colocamos as classes, ou os elementos, que vão realizar ou auxiliar na persistência das informações do sistema.
- A camada "view" é onde colocamos as classes, ou os elementos, que vão mostrar na tela alguma coisa ao usuário, ou seja, que vão montar e apresentar no monitor uma tela, ou uma janela de informações ao usuário que está utilizando o sistema.
- A camada "controller" ou "control" é aquela que conterá as classes ou elementos que permitem o controle das informações tanto que chegam ao usuário, quanto das que vão para o BD. Nesse caso, são os elementos responsáveis pela validação das informações que tramitam entre as camadas.

Pergunta 2

0,25 em 0,25 pontos

Sobre Design Patterns, qual a alternativa **incorreta**?

Resposta Selecionada: d.
Os Design Patterns são métodos específicos de classes preparadas especialmente para criação de imagens e desenhos.

Respostas:

a.
Os Design Patterns surgiram das melhores práticas, em que se reuniram as melhores e mais eficazes soluções, transformando-as em padrões e *Frameworks*.

b.
Os Design Patterns garantem maior eficiência e eficácia, compreensão do sistema (para os desenvolvedores), facilidade na manutenção, na utilização e na reutilização de elementos de programação.

c.
Os Design Patterns, ou Padrões de Projetos, são soluções generalistas para problemas recorrentes durante o desenvolvimento de um *software*.

d.
Os Design Patterns são métodos específicos de classes preparadas especialmente para criação de imagens e desenhos.

e.
Os Design Patterns surgiram da necessidade de geração, manutenção ou implementação de sistemas com mais agilidade, rapidez e preços competitivos.

Comentário da resposta: Resposta: D
Comentário: os Design Patterns podem ser considerados regras de programação que quando seguidas permitem uma criação (ou uma manutenção) mais organizada e eficaz de sistemas orientados a objetos. Algumas dessas regras foram surgindo à medida que sua utilização melhorava o processo de criação de sistemas, tornando mais eficaz a sua produção, de forma que se tornaram uma forma padrão de se desenvolver sistema.

Pergunta 3

0,25 em 0,25 pontos

Sobre Design Pattern, qual a alternativa correta?

Resposta Selecionada: e. Os Design Patterns possuem impacto direto no trabalho dos desenvolvedores, tornando-o mais dinâmico.

Respostas:

a. Os Design Patterns são características de sistemas, que tornam a utilização pelos usuários mais segura.

b.
Como a utilização dos Design Patterns exige uma documentação mais detalhada, isso torna o desenvolvimento de sistemas um pouco mais demorado e por isso tende a ser mais caro.

c.
Os Design Patterns, ou Padrões de Projetos, são elementos que, por serem complexos de se implantar, dificultam a utilização dos sistemas pelos usuários.

d.
Para o funcionamento dos Design Patterns, utiliza-se a memória disponível nos HDs ao invés da memória RAM e, portanto, sua utilização acaba tornando o sistema um pouco mais lento ao usuário.

e. Os Design Patterns possuem impacto direto no trabalho dos desenvolvedores, tornando-o mais dinâmico.

Comentário da resposta: Resposta: E
Comentário: um dos objetivos dos Design Patterns é tornar o trabalho dos desenvolvedores de sistemas mais dinâmico e eficaz. Esses padrões de desenvolvimento não estão relacionados ao que o cliente quer com o sistema, já que este pode propor qualquer coisa, dependendo da sua necessidade, mas de qualquer forma possuem impacto direto com o trabalho dos desenvolvedores sobre aquele sistema, de acordo com as solicitações dos clientes. Da mesma forma, trabalhar com Design Pattern não tem impacto direto com a segurança do sistema, que fica a cargo da experiência do desenvolvedor, o que não impede os desenvolvedores de utilizarem os Design Patterns para desenvolverem toda a lógica de segurança do sistema. Quanto à memória utilizada pelo sistema, é a mesma utilizada na funcionalidade normal de um sistema criado na linguagem Java, utiliza normalmente tanto a memória RAM (que a JVM utiliza) quanto o HD, onde estão gravadas as classes do sistema.

Pergunta 4

0,25 em 0,25 pontos

Sobre o Design Pattern DTO, qual a alternativa correta?

Resposta Selecionada: a. O DTO é um Pattern que está relacionado à transferência de informações entre objetos.

Respostas:

a. O DTO é um Pattern que está relacionado à transferência de informações entre objetos.

b.
O Pattern DTO define o tratamento de datas, como, por exemplo, a diferença de formatação existente entre a formatação padrão inglesa (MM, DD, YYYY) e a brasileira (DD, MM, YYYY).

c.
Para o Pattern DTO, ao criarmos um método numa classe, podemos inserir quantos parâmetros forem necessários, mesmo que sejam muitos parâmetros, sejam eles de qualquer um dos tipos existentes.

d.
No Pattern DTO, definem-se as datas em que serão feitas as transferências das operações, já que sua sigla significa "Data Transfer Operation".

e.
Ao utilizarmos o Pattern DTO, não podemos colocar objetos representando classes como parâmetro, já que um parâmetro de um método não aceita objetos em memória.

Comentário da resposta: Resposta: A
Comentário: o Pattern DTO dá preferência à utilização de objetos como parâmetros de métodos, de forma que esse objeto conterá tantos atributos quanto os dados necessários na transferência de informação entre métodos. Significa que ao passarmos informações entre métodos (quando um método invoca outro método), a não ser que sejam poucas as informações a serem transmitidas entre eles, deve-se preencher um objeto com essas informações e então enviar o objeto todo como parâmetro do método. O que devemos entender é que, quando se envia um objeto como parâmetro de um método, está se enviando na verdade o "endereço" de localização daquele objeto na memória, a fim de que o método que recebeu o objeto possa localizá-lo e utilizar diretamente os dados que estão guardados naquele objeto.

Pergunta 5

0,25 em 0,25 pontos

Sobre o Hibernate, qual a alternativa correta?

Resposta Selecionada: d.
Para se utilizar o Hibernate, a configuração da conexão com o Banco de Dados deve estar descrita no arquivo "persistence.xml".

Respostas:

a.
A desvantagem de se trabalhar com o Hibernate é que a utilização desse *framework* dificulta a alteração de Banco de Dados, caso haja a necessidade, imaginando que o cliente trabalhe com mais de um tipo de SGBD (como MySQL, SQL Server, Oracle, entre outros).

b.
O Hibernate é uma ferramenta de consulta e leitura de Banco de Dados, que não pode ser utilizado para persistir informação.

c.
Para que possamos utilizar o *framework* Hibernate, a classe que representa as entidades acessadas do Banco de Dados deve conter elementos que mapeiam a classe, os quais são conhecidos como "apontamentos" e que são termos que iniciam com o símbolo "%".

d.
Para se utilizar o Hibernate, a configuração da conexão com o Banco de Dados deve estar descrita no arquivo "persistence.xml".

e.
O Hibernate exige que todas as *queries* necessárias de acesso aos dados do Banco de Dados sejam geradas pelo desenvolvedor enquanto o sistema está sendo desenvolvido.

Comentário da resposta: Resposta: D
Comentário: o Hibernate é uma "ferramenta de persistência objeto/relacional" de alta performance que permite realizar consultas e alterações nas informações do Banco de Dados. É uma solução muito flexível e poderosa. Com ela, precisamos apenas fazer o mapeamento de classes Java em relação às tabelas do Banco de Dados (utilizando-se do que chamamos de "anotações" que são aqueles elementos de código que iniciam com o símbolo "@"). As facilidades são tantas na sua utilização de forma geral, que permite reduzir significativamente o tempo de desenvolvimento de sistemas. Como o Hibernate abstrai o código SQL, toda classe da camada JDBC, além de todo o código SQL, é gerado automaticamente em tempo de execução. Desta forma, o Hibernate gera sozinho o SQL que servirá para um determinado Banco de Dados (aquele que está configurado no arquivo "persistence.xml" do projeto, já que cada SGBD fala uma espécie de "dialeto" diferente da linguagem SQL, de forma que as diferenças são por vezes significativas). Essa característica possibilita a troca de Banco de Dados (do SGBD) sem ter que alterar as *queries* das quais o sistema utiliza, já que isso fica como responsabilidade da própria ferramenta. Para se lidar com as tabelas (e até gerá-las se necessário), é preciso configurar o JPA, ou seja, configurar as propriedades do banco (como em qual Banco de Dados vamos gravar nossas Tarefas, quais são o *login* e senha do usuário de acesso ao banco de dados, de forma que o JPA necessita dessas configurações, a serem colocadas no arquivo "persistence.xml" do projeto). Desta forma, para configurar a conexão com o banco de dados e permitir o acesso, o Hibernate precisa saber como se conectar a ele, de forma que isso é feito pelo arquivo "persistence.xml", que deve ficar dentro de uma pasta nomeada como "META-INF", localizada na raiz do projeto.

Pergunta 6

0,25 em 0,25 pontos

Sobre o Design Pattern DAO, qual a alternativa correta?

Resposta Selecionada: c.
O Pattern DAO define e reúne as classes (e, consequentemente, quando em execução, os objetos) que estão relacionadas à persistência dos dados.

Respostas:

a. As classes criadas a partir do Pattern DAO devem ser colocadas na camada de controle de um MVC.

b.
O Pattern DAO define que toda comunicação entre os objetos deve priorizar a utilização de classes nos parâmetros dos métodos.

c.
O Pattern DAO define e reúne as classes (e, consequentemente, quando em execução, os objetos) que estão relacionadas à persistência dos dados.

d.
No Pattern DAO, definem-se as datas em que serão realizados os acessos às operações de Banco de Dados, já que sua sigla significa “Date Access Operation”.

e.
As classes criadas sobre o Pattern DAO devem prever a forma com que o usuário irá visualizar as informações lidas de um Banco de Dados.

Comentário da resposta: Resposta: C
Comentário: o Pattern DAO, sigla que significa "Data Access Object", cuja tradução indica que se trata dos objetos responsáveis pelo acesso aos dados, ou seja, são os objetos responsáveis por realizar a persistência das informações, comunicando-se diretamente com o Banco de Dados. De uma forma geral, quando se está desenvolvendo um sistema Java para isso, as variáveis, que definem as classes que representam as entidades do Banco de Dados, possuem por convenção a sigla DAO no seu nome, como: a classe que irá conter o código que irá manipular os dados da tabela Aluno (junto ao sistema) será a classe AlunoDAO, assim como a que irá manipular os dados da tabela Produto será a classe ProdutoDAO etc. Segundo o Pattern MVC, a camada de Modelagem (camada "model") é a camada responsável pela persistência das informações. Seguindo essa ideia, as classes DAO, quando geradas num sistema construído utilizando-se do Pattern MVC, deverão ficar localizadas nesta camada ("model").

Pergunta 7

0,25 em 0,25 pontos

Sobre as anotações existentes nas classes que utilizam o Hibernate, qual a alternativa correta?

Resposta Selecionada: b.
A anotação "@Entity" define a classe como uma entidade do banco de dados, tornando seus objetos passíveis de serem persistidos.

Respostas:

a.
A anotação "@Pk" define o atributo ao qual está precedendo, como a primary key da tabela referenciada como Entidade.

b.
A anotação "@Entity" define a classe como uma entidade do banco de dados, tornando seus objetos passíveis de serem persistidos.

c.
A anotação "@IncrementedValue", colocada junto à anotação de Id, define que esse valor será autoincrementável pelo próprio banco de dados.

d.
As anotações ou, ainda, esses termos precedidos pelo símbolo "@" somente são usados quando estamos utilizando o *framework* Hibernate.

e. Uma anotação é sempre inserida logo após as linhas de código às quais está se referindo.

Comentário da resposta: Resposta: B
Comentário: a fim de mapear uma classe que representa uma Tabela do Banco de Dados, adicionam-se "anotações" no código. É um recurso da linguagem Java que permite inserir "metainformações" que serão lidas e utilizadas por *frameworks* e bibliotecas, assim como o Hibernate, por exemplo, a fim de que tomem decisões baseadas nessas configurações. Uma dessas anotações é a anotação "@Entity" colocada logo antes da declaração de uma classe, indicando que os objetos gerados a partir dessa classe tornarão suas informações "persistíveis" no banco de dados. A anotação @Id colocada logo antes de um dos atributos da classe indica que esse atributo é a chave primária da tabela que a classe representa. A anotação "@GeneratedValue", quando colocada antes do atributo que representa a chave primária da tabela, indica que seu valor será gerado automaticamente, ou seja, terá a característica de "autoincremento". A anotação "@Temporal" é utilizada para se trabalhar com "data e hora". Cada uma dessas anotações precisa que os devidos "imports" estejam presentes na classe, já que as classes que permitem sua utilização pertencem à biblioteca (ao pacote) *javax.persistence*.

Pergunta 8

0,25 em 0,25 pontos

O Pattern MVC pode ser entendido como uma forma de separar as responsabilidades entre os componentes de uma aplicação, seja ela *desktop* ou para *web*. Desta forma, em um *site* criado utilizando-se do MVC, é correto afirmar que:

Resposta Selecionada: a.
As classes que montam janelas são representadas junto aos componentes da camada View e as classes de entidades do banco de dados junto aos componentes da camada Model.

Respostas: a.
As classes que montam janelas são representadas junto aos componentes da camada View e as classes de entidades do banco de dados junto aos componentes da camada Model.

b.
As classes responsáveis pelo acesso a dados e que atendem ao padrão DAO (*Data Access Object*) são representadas junto aos componentes do tipo Controller.

c. As páginas *web* e as classes que montam janelas são representadas nos componentes View e Model.

d.
As classes que implementam regras de negócio e os arquivos com as páginas *web* são representadas no componente controller.

e.
Num sistema criado sob os moldes do Pattern MVC, não pode haver comunicação entre os componentes da camada View e os da camada Model, sem que se passe pela camada controller.

Comentário da resposta: Resposta: A
Comentário: o Pattern MVC sugere uma divisão em camadas (pacotes) para o grupo de classes criadas no sistema, em que as classes são inseridas quando geradas no projeto. No pacote "model" são colocadas as classes ligadas à persistência dos dados, como as classes geradas sob o Pattern DAO. No pacote "view" são colocadas as classes ligadas à construção das telas que o usuário visualizará ao utilizar o sistema. No pacote "control" são colocadas as classes que intermediarão as classes dos outros pacotes, o que não significa que as classes da camada "view" não possam acessar diretamente as classes da camada "model", se assim se fizer necessário.

Pergunta 9

0,25 em 0,25 pontos

Considere uma aplicação *web* em desenvolvimento utilizando Java com páginas *web* e o *design* Pattern MVC. Nesse contexto, é correto afirmar que:

Resposta Selecionada: e.
Uma classe "Cliente" que representa a tabela de clientes do banco de dados e que possui diversos atributos além dos seus respectivos métodos getters e setters deve ser definida na camada MODEL do MVC.

Respostas:

a.
Uma classe "ClienteDAO" que possui métodos para acessar o banco de dados e executar instruções SQL deve estar definida na camada CONTROLLER do MVC.

b.
Uma página *web*, que gera no *browser* um formulário de cadastro do cliente, deve ser representada no componente CONTROLLER do MVC.

c. Uma página *web* nunca poderá acessar diretamente uma classe do tipo DAO (*Data Access Object*).

d. As classes de montagem dos formulários de preenchimento geralmente estão definidas na camada MODEL do MVC.

e.
Uma classe "Cliente" que representa a tabela de clientes do banco de dados e que possui diversos atributos além dos seus respectivos métodos getters e setters deve ser definida na camada MODEL do MVC.

Comentário da resposta: Resposta: E
Comentário: as classes que representam as entidades do banco de dados (como as tabelas) são criadas sempre encapsuladas (com os atributos privados e seus respectivos métodos "getters" e "setters"), em que seus atributos representam os campos (ou as colunas) da tabela no banco de dados. Essas classes encapsuladas que representam uma entidade do banco de dados (como uma tabela, por exemplo) são conhecidas como "JavaBeans". Esses "JavaBeans", por serem representantes de entidades do banco de dados, costumam ficar localizados na camada "model" do MVC, assim como as classes ligadas ao Pattern DAO.

Pergunta 10

0 em 0,25 pontos

Uma forma de configurar o *framework* Hibernate é por meio do arquivo "persistence.xml". Analise cada um dos itens a seguir verificando se as propriedades neles descritas podem ser configuradas nesse arquivo XML.
I - O dialeto que o Hibernate utilizará para a montagem dos comandos SQL.
II - O nome completo da classe do *driver* JDBC.
III - O nome e a senha do usuário que permitirão estabelecer a conexão com o banco de dados.
IV - A URL de conexão com o banco de dados.

Desta forma, qual das opções indica corretamente os itens que possuem as propriedades possíveis de serem configuradas?

Resposta Selecionada: d. Somente as propriedades dos itens III e IV.

Respostas:

a. Somente as propriedades dos itens I e III.

b. Somente as propriedades dos itens II e IV.

c. As propriedades de todos os itens.

d. Somente as propriedades dos itens III e IV.

e. Somente as propriedades dos itens I, II e III.

← OK