Programação Orientada a Objetos - COM230 - Turma 012

Página Inicial
Avisos
Cronograma
Atividades
Fóruns
Calendário Lives
Collaborate
Notas

Menu das Semanas
Semana 1
Semana 2
Semana 3
Semana 4
Semana 5
Semana 6
Semana 7
Semana 8
Orientações para realização da prova
Orientações para realização do exame

Documentos e informações gerais

Gabaritos
Referências da disciplina
Repositório de REA's

Revisar envio do teste: Semana 4 - Atividade Avaliativa

Usuário	LIZIS BIANCA DA SILVA SANTOS
Curso	Programação Orientada a Objetos - COM230 - Turma 012
Teste	Semana 4 - Atividade Avaliativa
Iniciado	02/11/23 15:01
Enviado	02/11/23 15:11
Data de vencimento	03/11/23 05:00
Status	Completado
Resultado da tentativa	10 em 10 pontos
Tempo decorrido	10 minutos
Instruções	Olá, estudante!

1. Para responder a esta atividade, selecione a(s) alternativa(s) que você considerar correta(s);
2. Após selecionar a resposta correta em todas as questões, vá até o fim da página e pressione "Enviar teste".
3. A cada tentativa, você receberá um conjunto diferente de questões.

Pronto! Sua atividade já está registrada no AVA.

Resultados exibidos: Todas as respostas, Respostas enviadas, Respostas corretas, Comentários, Perguntas respondidas incorretamente

Pergunta 1

1,42 em 1,42 pontos

Na programação orientada a objetos, é comum o uso de classes e métodos genéricos, visando reduzir a quantidade de códigos de conversão de tipos de dados.

Considerando esse contexto, assinale a alternativa que apresenta, corretamente, o uso dos "Generics" em linguagem Java.

Resposta Selecionada: b. List<Carro> carros = new ArrayList<Carro>();

Respostas:
a. List lista = new ArrayList(); lista.add("dois");
b. List<Carro> carros = new ArrayList<Carro>();
c. Object[] pessoas = new Object[Pessoa];
d. Pessoa pessoa = new Pessoa();
e. list = Arrays.asList(letters);

Comentário da resposta: **JUSTIFICATIVA**
Com uso de "Generics", a lista pode somente ter objetos do tipo "Carro", não aceitando objetos de outro tipo, assim, é possível assegurar qual tipo será retornado pela lista, caso contrário, o erro será percebido somente em tempo de execução.

Pergunta 2

1,42 em 1,42 pontos

Considere o código a seguir, em linguagem Java (qualquer versão), que apresenta a utilização de um tipo genérico para uma lista de nomes.

```
public static void main(String[] arga) {

    List<String> nomes = new ArrayList<String>();
    nomes.add("José");
    nomes.add("João");
    nomes.add("Maria");
    System.out.println(nomes.get(0));

  }
```

Em relação a esse código, é possível afirmar que:

Resposta Selecionada: d. o uso do tipo genérico é observado na notação <String>.

Respostas:
a. qualquer tipo de dado poderia ser adicionado à lista, por ser genérica.
b. o tipo genérico aplicado à lista é "nomes".
c. o uso do tipo genérico é restrito ao uso de listas e coleções.
d. o uso do tipo genérico é observado na notação <String>.
e. o uso do tipo genérico pode ser observado nos comandos ".add".

Comentário da resposta: **JUSTIFICATIVA**
Na programação orientada a objetos, o uso de classes e métodos genéricos pode reduzir o código de programação e prevenir erros, como é o caso do código apresentado. Utiliza-se a notação <TipoDeDados> com o "Generics", em linguagem Java.

Pergunta 3

1,42 em 1,42 pontos

Um "Iterator" é um objeto que pode ser usado para percorrer uma estrutura de dados do tipo "Collection", como é o caso de "ArrayList" e "HashSet". O uso do "Iterator" é um recurso muito importante para a linguagem Java.

Considerando o uso do "Iterator", analise as afirmativas a seguir e assinale V para a(s) verdadeira(s) e F para a(s) falsa(s).

I. ( ) A função de objetos do tipo "Iterator" é permitir percorrer e remover elementos de uma coleção.
II. ( ) Toda coleção tem um método que retorna um "Iterator", esse recurso faz parte da classe "Collection" e suas filhas.
III. ( ) Um exemplo da aplicação da interface "Iterator" em Java é o uso do recurso "For-Each".
IV. ( ) O método "hasNext()" retorna "true", se há elementos a serem lidos na coleção que está sendo percorrida.

Assinale a alternativa que apresenta a sequência correta.

Resposta Selecionada: d. V, V, F, V.

Respostas:
a. F, V, V, V.
b. F, F, F, V.
c. V, F, V, F.
d. V, V, F, V.
e. V, V, F, F.

Comentário da resposta: **JUSTIFICATIVA**
A afirmativa I é verdadeira, pois a interface "Iterator" é utilizada para percorrer e remover elementos de uma coleção, por meio de três métodos:

- hasNext(): retorna "true", se há elementos a serem lidos no iterador;
- next(): retorna o próximo elemento do iterador;
- remove(): remove o último elemento obtido pela chamada de next().

A afirmativa II é verdadeira, pois todos os objetos herdados da classe "Collection" também herdam o "Iterator". A afirmativa III é falsa, pois o "For-each" não é um exemplo de aplicação da interface "Iterator". Apesar do For-each, quando aplicado a uma coleção do arcabouço de coleções, usar internamente um iterador, esse iterador não está exposto ao usuário. Além disso, o "For-each" pode ser usado para percorrer também Arrays simples e nesse caso não usa iteradores. A afirmativa IV é verdadeira, pois o método "hasNext()" é utilizado para verificar a existência de elementos a serem percorridos em um objeto "Collection".

Pergunta 4

1,42 em 1,42 pontos

O Java Iterator também é conhecido como o cursor universal de Java, pois é apropriado para todas as classes do *framework* Collection. Ele também auxilia nas operações como READ e REMOVE. Quando comparamos a interface do Java Iterator com a interface do iterador de enumeração, podemos dizer que os nomes dos métodos disponíveis no primeiro são mais precisos e fáceis de usar.

Assinale a alternativa correta a respeito do Java Iterator, a seguir.

Resposta Selecionada: c.
Se um usuário estiver trabalhando com um *loop for*, ele não poderá modernizar (adicionar/remover) a coleção. Por outro lado, se usar o Java Iterator, poderá simplesmente atualizar a coleção.

Respostas:
a.
Nas novas versões, não é necessária a criação de uma instância da interface Java Iterator, pois a biblioteca já vem predefinida, facilitando ainda mais o seu desenvolvimento.
b.
O Java Iterator possui todas as operações relacionadas ao CRUD, ou seja, criação, leitura, atualização e exclusão, suportando diversas operações em paralelo com confiabilidade.
c.
Se um usuário estiver trabalhando com um *loop for*, ele não poderá modernizar (adicionar/remover) a coleção. Por outro lado, se usar o Java Iterator, poderá simplesmente atualizar a coleção.
d.
O Java Iterator preserva a iteração nas duas direções, ou seja, trata-se de um iterador bidirecional, atualizando tanto do modelo para a base de dados quanto da base para o modelo.
e.
Java Iterator e Collections são coisas distintas: o primeiro trabalha com a tratativa de dados simples; o segundo consegue realizar operações com uma grande quantidade de dados.

Comentário da resposta: **JUSTIFICATIVA**
O Java Iterator realiza atualizações mais facilmente do que as simples operações em *loop*. No entanto, possui limitações em relação às operações realizadas pelo CRUD, além de suportar operações em uma única direção. É necessária a criação de uma instância da interface Java Iterator, uma vez que a biblioteca não vem predefinida como padrão.

Pergunta 5

1,48 em 1,48 pontos

Leia o trecho a seguir.

"Todo tipo primitivo tem uma classe empacotadora de tipo correspondente (no pacote java.lang). Essas classes chamam-se *Boolean*, *Byte*, *Character*, *Double*, *Float*, *Integer*, *Long* e *Short*. Elas permitem manipular valores de tipo primitivo como objetos. [...] Mas podem manipular objetos das classes empacotadoras de tipo, porque cada classe, em última análise, deriva de Object" (DEITEL; DEITEL, 2016, p. 539).

DEITEL, P., DEITEL, H. **Java:** como programar. 10. ed. São Paulo: Pearson Education do Brasil, 2016.

Sobre tipos e classes, analise as asserções a seguir e a relação proposta entre elas.

I. A interface List é implementada por várias classes, inclusive pelas classes ArrayList, LinkedList e Vector. O *autoboxing* ocorre quando valores de tipo primitivo são adicionados aos objetos dessas classes.

POIS

II. As variáveis do tipo primitivo armazenam apenas referências a objetos, em que as classes ArrayList e Vector são implementações de *arrays* redimensionáveis de List.

A respeito das asserções, assinale a alternativa correta a seguir.

Resposta Selecionada: d. As asserções I e II são proposições verdadeiras, e a II é uma justificativa correta da I.

Respostas:
a. A asserção I é uma proposição verdadeira, e a asserção II é uma proposição falsa.
b. A asserção I é uma proposição falsa, e a asserção II é uma proposição verdadeira.
c. As asserções I e II são proposições verdadeiras, mas a II não é uma justificativa correta da I.
d. As asserções I e II são proposições verdadeiras, e a II é uma justificativa correta da I.
e. As asserções I e II são proposições falsas.

Comentário da resposta: **JUSTIFICATIVA**
A asserção I é uma proposição verdadeira. Como List é uma interface, objetos do tipo lista não podem ser criados. Assim, sempre será necessária uma classe que implemente a List para criar um objeto. A asserção II também é uma proposição verdadeira e justifica corretamente a I, pois essas variáveis relacionadas à ArrayList e à List apontam referências apenas para poderem ser implementadas, independentemente do tipo que será trabalhado na codificação.

Pergunta 6

1,42 em 1,42 pontos

A interface List fornece uma maneira de armazenar uma coleção de objetos. Analise as afirmações a seguir e escolha a alternativa correta.

I. A interface List é uma interface filha de Collection.
II. A interface List é uma coleção ordenada de objetos na qual valores duplicados podem ser armazenados.
III. A List não preserva a ordem de inserção e não permite o acesso posicional ou a inserção de elementos.
IV. São métodos da interface List implementados pela classe ArrayList: Add, Remove, Clear e Contains.

Resposta Selecionada: Apenas I, II e IV estão corretas.

Respostas:
Apenas II está correta.
Apenas I, II e IV estão corretas.
Apenas I, II e IV estão corretas.
Apenas I e IV estão corretas.
Apenas III está correta.

Comentário da resposta: A resposta correta é: "Apenas I, II e IV estão corretas."

Justificativa
Em II há um erro conceitual, porque uma das características da List é preservar a ordem de inserção e permitir o acesso posicional, assim como a inserção de novos elementos.

Pergunta 7

1,42 em 1,42 pontos

Assinale a alternativa correta quanto ao uso de Iterator na linguagem Java.

I. Quanto a um Iterator, sabemos que com seu uso é possível percorrer qualquer coleção que visita seus elementos, como se ela fosse uma estrutura sequencial, atuando como um ponteiro para o próximo elemento.
II. Um iterator não é obtido da própria coleção ao usar o método iterator(), pois para cada interface ou classe de uma coleção um iterador precisa ser implementado.
III. O método iterator retorna um objeto iterator posicionado depois do primeiro objeto da coleção.
IV. O método delete() do interator remove o último item retornado pelo método next().

Resposta Selecionada: Apenas I está correta.

Respostas:
Apenas III e IV estão corretas.
Apenas II está correta.
Apenas II e III estão corretas.
Apenas I está correta.
Apenas I, II e III estão corretas.

Comentário da resposta: A resposta correta é: "Apenas I está correta."

Justificativa
Em II, há um erro, pois na realidade o iterador é obtido da própria coleção e não é necessário implementar isso para cada classe. Em III, o posicionamento é feito antes do primeiro objeto da coleção. Em IV, não há método delete() e sim remove().

Quinta-feira, 15 de Agosto de 2024 21h56min31s BRT

← OK