



CIDADE DE BAGÉ
INSTRUÇÕES GERAIS

- 1 - Este caderno de prova é constituído por 40 (quarenta) questões objetivas.
- 2 - A prova terá duração máxima de 04 (quatro) horas.
- 3 - Para cada questão, são apresentadas 04 (quatro) alternativas (a – b – c – d).
APENAS UMA delas responde de maneira correta ao enunciado.
- 4 - Após conferir os dados, contidos no campo “Identificação do Candidato” no Cartão de Resposta, assine no espaço indicado.
- 5 - Marque, com caneta esferográfica azul ou preta de ponta grossa, conforme exemplo abaixo, no Cartão de Resposta – único documento válido para correção eletrônica.


- 6 - Em hipótese alguma, haverá substituição do Cartão de Resposta.
- 7 - Não deixe nenhuma questão sem resposta.
- 8 - O preenchimento do Cartão de Resposta deverá ser feito dentro do tempo previsto para esta prova, ou seja, 04 (quatro) horas.
- 9 - Serão anuladas as questões que tiverem mais de uma alternativa marcada, emendas e/ou rasuras.
- 10 - O candidato só poderá retirar-se da sala de prova após transcorrida 01 (uma) hora do seu início.

BOA PROVA!

1. A compilação de uma página JSP resulta em um _____

A alternativa que preenche corretamente a lacuna é

- a) documento XHTML no lado do cliente.
- b) servlet Java no lado do servidor.
- c) managed bean no lado do servidor.
- d) documento HTML no lado do cliente.

2. Durante a fase de projeto de um sistema, definiu-se que ele será constituído de um modelo de 3 camadas.

Nesse modelo, em relação ao acesso a dados, as telas de cadastro

- a) podem acessar a camada de dados, principalmente porque essa camada armazena as informações de acesso aos dados por meio de telas de configuração de sistema.
- b) podem acessar a camada de dados, uma vez que a camada de negócio só deve ser invocada se os dados referentes à camada de apresentação fizerem parte de alguma regra de negócio de sistema.
- c) não podem acessar a base de dados, porque a descrição do modelo de dados está armazenada na camada de negócio e de transporte.
- d) não podem acessar a base de dados, uma vez que, para a camada de apresentação, a camada de dados é transparente e a camada de apresentação não possui informações sobre a localização da camada de dados.

3. Considerando a linguagem XML, afirma-se que

- a) uma tag de fechamento em XML sempre deve existir.
- b) as tags XML podem ser iniciadas com números.
- c) os atributos das tags XML não podem ser escritos entre aspas.
- d) as tags XML não são case sensitive.

4. No âmbito dos Web Services, o protocolo padrão OASIS que especifica um método para descoberta de diretórios de serviços é o

- a) UDDI.
- b) SOAP.
- c) WSDL.
- d) DOM/XML.

5. A segurança da Informação é a área do conhecimento dedicada à proteção da informação contra alterações indevidas, acessos não autorizados, ou a sua indisponibilidade.

Segundo a NBR ISO/IEC 27002-2005, a segurança das informações é caracterizada pela preservação dos seguintes princípios básicos, **EXCETO**:

- a) integridade.
- b) viabilidade.
- c) confiabilidade.
- d) disponibilidade.

6. Qual solução de J2EE é utilizada para comunicação entre uma aplicação J2EE e sistemas legados não escritos em Java?

- a) JAX-WS.
- b) JSF.
- c) JAAS.
- d) EJB.

- 7.** Assinale a alternativa que se refere ao padrão de projeto utilizado para garantir que uma classe seja instanciada somente uma vez.
- a) OneInstance.
 - b) SimpleObject.
 - c) Restrictor.
 - d) Singleton.
- 8.** Quanto à programação de aplicações para Web em Java, afirma-se que a ação
- a) forward é utilizada para transferência de controle de um componente Web para outro. Ela pode ser utilizada diversas vezes no código de um mesmo servlet, pois a ação é executada em segundo plano.
 - b) include tem o mesmo efeito que a ação forward, entretanto somente pode ser utilizada uma única vez, visto que o controle não volta ao servlet que a executou. Uma exceção é disparada caso seja executada novamente no mesmo servlet.
 - c) forward é utilizada para transferência de controle de um componente Web para outro. Entretanto, ela pode ser utilizada uma única vez no mesmo servlet. Uma nova chamada gera uma exceção.
 - d) include deve ser utilizada para incluirmos recursos web dentro de um componente Web. Entretanto, temos que tomar o cuidado de executá-la uma única vez, visto que várias chamadas geram uma exceção.
- 9.** Um ataque é uma tentativa deliberada de burlar serviços de segurança de um sistema. Um tipo de ataque passivo é a
- a) Falsidade.
 - b) análise de tráfego.
 - c) negação de serviço.
 - d) modificação de mensagem.
- 10.NÃO** é uma atribuição da Autoridade Certificadora
- a) Emissão de certificados digitais para pessoas físicas.
 - b) Determinação das políticas e dos procedimentos que orientam o uso de certificados.
 - c) Identificação dos usuários e submissão da solicitação do certificado ao Comitê Gestor da ICP.
 - d) Emissão e publicação da lista de certificados revogados.
- 11.**O arquivo de configuração do apache no Linux/Ubuntu é
- a) httpd.conf
 - b) hosts.conf
 - c) apache2.conf
 - d) conf.d
- 12.**Quais as permissões para o arquivo /www/var/index.html, para ser visualizado pela WEB?
- a) -rwxrwx--- 1 root root 177 Abr 4 19:38 index.html
 - b) drwxrwx-w- 1 root web 177 Abr 4 19:38 index.html
 - c) -rw-rw--- 1 root apache 177 Abr 4 19:38 index.html
 - d) -rwxrwxr-x 1 root root 177 Abr 4 19:38 index.html

13. Observe o comando SQL

```
CREATE TABLE PESSOA (  
    SECAO CHAR(1) NOT NULL,  
    ID INTEGER NOT NULL,  
    NOME VARCHAR(50) NOT NULL,  
    CONSTRAINT PK_PESSOA PRIMARY KEY(SECAO, ID)  
);
```

Considerando que a instrução SQL insert é utilizada para inserção de dados em tabelas, analise a sintaxe dos comandos:

- I. insert into pessoa values ('S',100,'OLAVO BILAC');
- II. insert into pessoa (secao,id,nome) values ('M',200,'JULIO DE CASTILHOS');
- III. insert into pessoa (secao,id,nome) select 'X',p.id,p.nome from pessoa p where p.secao <> 'X';

Está(ão) correto(s) a(s) afirmativa(s)

- a) I apenas.
- b) I e III apenas.
- c) II apenas.
- d) I, II e III.

14. Considere o seguinte comando SQL

```
CREATE TABLE ATOR (  
    ID INTEGER NOT NULL,  
    NOME VARCHAR(50) NOT NULL,  
    CONSTRAINT CLIENTE_PK PRIMARY KEY(ID)  
);
```

```
CREATE TABLE FILME (  
    ID INTEGER NOT NULL,  
    TITULO VARCHAR(50) NOT NULL,  
    CONSTRAINT FILME_PK PRIMARY KEY(ID)  
);
```

```
CREATE TABLE ESTRELA(  
    ID_ATOR INTEGER NOT NULL,  
    ID_FILME INTEGER NOT NULL,
```

```
CONSTRAINT ESTRELA_PK PRIMARY KEY (ID_ATOR, ID_FILME),  
CONSTRAINT ESTRELA_ATOR_FK FOREIGN KEY (ID_ATOR) REFERENCES ATOR(ID),  
CONSTRAINT ESTRELA_FILME_FK FOREIGN KEY (ID_FILME) REFERENCES FILME(ID)  
);
```

A instrução SQL que efetua a contagem de quantos filmes estrelados, por ator, com pelo menos um filme estrelado, é:

- a)

```
select nome, count(*) from ator, filme, estrela  
where  
  
ator.id = filme.id  
  
group by count(*)  
  
having count(*) > 1;
```
- b)

```
select nome, count(*) from ator, estrela  
where ator.id = estrela.id_ator  
  
group by count(*)  
  
having count(*) > 0;
```
- c)

```
select nome, count(*) from ator, estrela  
where ator.id = estrela.id_ator  
  
group by nome
```
- d)

```
select nome, count(*) from ator, filme, estrela  
where estrela.id_ator = ator.id and  
  
estrela.id_filme = filme.id  
  
group by nome  
  
having count(*) > 1
```

15. Na linguagem SQL, a subdivisão DCL possui os comandos:

- a) CREATE e ALTER.
- b) DROP, CREATE e ALTER.
- c) GRANT e REVOKE.
- d) BEGIN WORK, COMMIT e ROLLBACK.

16. As cláusulas de condição, para definir os dados que se desejam selecionar ou modificar, em uma consulta SQL são:

- a) from, where, group by, having, order by e distinct.
- b) from, where, and, or, not, group by, having, order by e distinct.
- c) from, where, group by, having e distinct.
- d) where, group by, having e distinct, not.

17.A instrução SQL, que exclui todos os dados de uma tabela de forma muito rápida (como, por exemplo, uma tabela chamada tableA), apaga os dados dentro da tabela e não a própria tabela com um log mínimo, é

- a) delete from tableA with noLog;
- b) truncate tableA;
- c) drop tableA;
- d) truncate table tableA;

18.Em transformações entre modelos de ER, para relacional em relacionamentos 1:1, cabe, como alternativa(s) preferida(s)

- a) tabela própria.
- b) adição de coluna e fusão de tabelas.
- c) fusão de tabelas.
- d) tabela própria e fusão de tabelas.

19.As classificações dos mecanismos de restrição de integridade em um SGBD (Sistema Gerenciador de Banco de Dados), garantidas automaticamente por um SGBD relacional, podem ser identificadas por

- a) chaves primárias.
- b) chaves primária e estrangeira.
- c) integridade de domínio, integridade de chave, integridade de vazio, integridade referencial.
- d) integridade de domínio, integridade de chave, integridade de vazio, integridade referencial e integridade de semântica.

20.O conceito de _____ serve para associar informações a ocorrências de entidades ou de relacionamentos.

Qual a palavra que preenche corretamente a lacuna?

- a) identificador
- b) cardinalidade
- c) domínio
- d) atributo

21.Com o objetivo de fornecer múltiplas visões do sistema a ser modelado, a UML 2.0 divide os diagramas em duas categorias: estruturais e comportamentais.

São exemplos desses diagramas, respectivamente

- a) diagrama de sequência e diagrama de comunicação.
- b) diagrama de casos de uso e diagrama de máquina de estados.
- c) diagrama de classes e diagrama de implantação.
- d) diagrama de casos de uso e diagrama de classes.

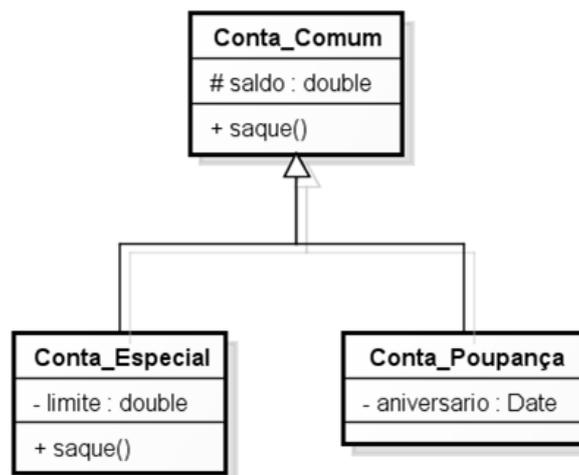
22.Na UML 2.0 o diagrama que determina as necessidades de hardware do sistema é o

- a) diagrama de componentes.
- b) diagrama de implantação.
- c) diagrama de objetos.
- d) diagrama de estrutura composta.

23. Segundo as características dos diagramas da UML 2.0, um diagrama de sequência

- representa os subsistemas ou submódulos englobados por um sistema de forma a determinar as partes que o compõem.
- concentra-se em como os elementos do diagrama estão vinculados e quais mensagens trocam entre si durante o processo.
- apresenta uma linguagem simples e de fácil compreensão para que os usuários possam ter uma ideia geral de como o sistema irá se comportar.
- preocupa-se com a ordem temporal em que as mensagens são trocadas entre os objetos envolvidos em um determinado processo.

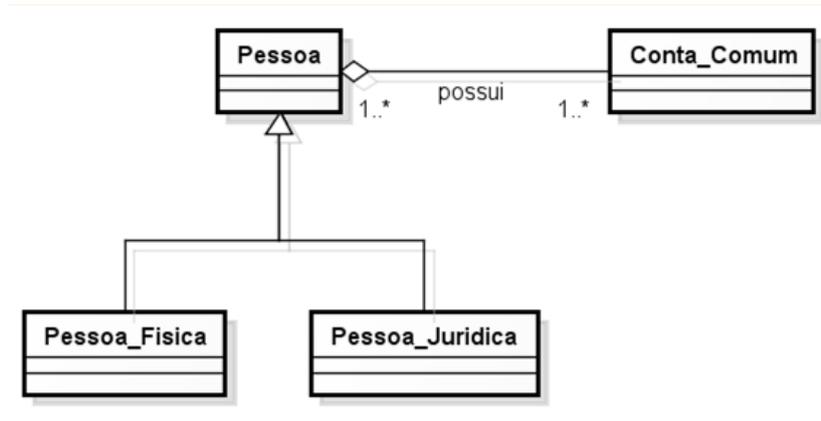
24. Considere-se o diagrama de classes UML abaixo, em que *Conta_Comum* e *Conta_Especial* possuem particularidades de saque. Embora as classes possuam métodos com o mesmo nome, as implementações desses métodos são diferentes.



Essa redeclaração de métodos caracteriza um exemplo de

- dependência.
- agregação.
- polimorfismo.
- generalização.

25. Analisando o diagrama de classes abaixo.



A alternativa que corresponde ao relacionamento entre as classes é

- a) Agregação (Pessoa, Conta_Comum), herança (Pessoa, Pessoa_Fisica).
- b) Composição (Pessoa, Conta_Comum), herança (Pessoa, Pessoa_Fisica).
- c) Agregação (Pessoa, Conta_Comum), composição (Pessoa, Pessoa_Fisica).
- d) Composição (Pessoa, Conta_Comum), herança (Pessoa, Pessoa_Fisica).

26. Para modelar aplicações baseadas no paradigma de Orientação a Objetos, utiliza-se UML, por ser uma linguagem visual e pela versatilidade de seus diagramas.

A seguir seguem características de alguns diagramas disponibilizados pela UML 2.0:

- I. Utilizado geralmente nas fases de levantamento e análise de requisitos e pode servir de base para outros diagramas.
- II. Demonstra o comportamento de um elemento por meio de um conjunto finito de transições de estado.
- III. Não se preocupa com a temporalidade do processo, concentrando-se em como os elementos do diagrama estão vinculados e quais mensagens trocam entre si.
- IV. Define a estrutura das classes utilizadas pelo sistema, determinando os atributos e/ou métodos que cada classe possui.

A partir das características listadas acima, determine a sequência correta quanto ao nome dos diagramas.

- a) I - Casos de Uso; II - Máquina de Estados; III - Sequência; IV - Componentes.
- b) I - Casos de Uso; II - Máquina de Estados; III - Comunicação; IV - Classes.
- c) I - Classes; II - Casos de Uso; III - Comunicação; IV - Máquina de Estados.
- d) I - Casos de uso; II - Máquina de estados; III - Sequência; IV - Classes.

27. Em um diagrama de classes, nos casos em que existam atributos relacionados a uma associação e que não possam ser armazenados por nenhuma das classes envolvidas utiliza-se

- a) classe associativa.
- b) composição.
- c) associação ternária.
- d) agregação.

28.Alguns relacionamentos da UML tentam demonstrar uma relação todo/parte entre os objetos associados. Há um tipo especial de associação, em que se tenta demonstrar que as informações de um objeto (chamado objeto-todo) precisam ser complementadas pelas informações contidas em um ou mais objetos de outra classe (chamados objetos-parte).

Esse tipo especial de associação é chamado de

- a) especialização.
- b) realização.
- c) agregação.
- d) generalização.

29.Em páginas web, eventos podem ser utilizados para disparar scripts. O evento *onblur* ocorre quando

- a) a página é carregada.
- b) um formulário é submetido.
- c) um elemento perde o foco.
- d) uma imagem é ofuscada.

30.Elementos de formulário possuem propriedades que precisam ser avaliadas quando se realiza a validação.

A propriedade *checked* aplica-se a

- a) caixas de seleção e caixas de verificação.
- b) botões de rádio e campos de texto.
- c) caixas de seleção e caixas de verificação.
- d) caixas de verificação e botões de rádio.

31.É sabido que o modelo de caixa da *Casting Style Sheet* (CSS) constrói o layout de um documento web, transformando cada elemento da árvore do documento em uma caixa cujas dimensões e cujos comportamentos são definidos pelos valores das propriedades CSS.

As propriedades que determinam as dimensões dessas caixas são

- a) margin, border, position, width e height
- b) margin, background, padding, width e height
- c) margin, border, padding, width e height
- d) margin, display, padding, width e height

A questão 32 baseia-se na Figura 1, que mostra um código escrito em uma linguagem de marcação para web. Nessa figura os números à esquerda representam cada linha de código fonte. Eles são meramente ilustrativos e não fazem parte do programa.

```

1  <!DOCTYPE html>
2  <html lang="pt-br">
3      <head>
4          <meta charset="utf-8">
5          <title> Título da página </title>
6          <link rel="stylesheet" href="/estilo/main.css">
7      </head>
8      <body>
9          <h1>Minha página</h1>
10     </body>
11 </html>

```

Figura 1: Código em uma linguagem de marcação para web.

32. Considerando a Figura 1, trata-se de um código criado na versão

- HTML 4.01 na codificação de caracteres UTF-8.
- HTML 5 na codificação de caracteres UTF-8.
- HTML 4.0 na codificação de caracteres UTF-8.
- xHTML 1.0 na codificação de caracteres UTF-8.

33. São exemplos de elementos para controle de formulário(s) em HTML:

- input, option, tr, label e select.
- input, option, label, td e select.
- input, button, fieldset, table e select.
- input, button, select, option e textarea.

A questão 34 baseia-se nas Figuras 2(a) e 2(b). Ambas apresentam classes implementadas na linguagem de programação Java e fazem parte do mesmo programa. Nessas figuras os números à esquerda representam cada linha de código fonte. Eles são meramente ilustrativos e não fazem parte do programa Java.

```

1 package soma;
2 public class Inteiro {
3     int i;
4 }
    
```

Figura 2(a): Programa em Java - Classe Inteiro.

```

1 package soma;
2 public class Soma {
3     public static void main(String[] args) {
4         int i;
5         Inteiro n = new Inteiro();
6         int soma = 1;
7         i = 2;
8         n.i = 4;
9         somarTres(i);
10        diminuirCinco(n.i);
11        soma += n.i * i;
12        System.out.println(soma);
13    }
14    public static void somarTres(int val){
15        val += 3;
16    }
17    public static void diminuirCinco(int val){
18        val -= 5;
19    }
20 }
    
```

Figura 2(b): Programa em Java - Classe Soma.

34. Considere a execução do programa Java mostrado nas Figuras 2(a) e 2(b), o valor da variável soma, quando o programa executar a linha 12 da Figura 2(b), é

- a) -1
- b) 8
- c) 7
- d) 9

As questões 35 e 36 baseiam-se nas Figuras 3(a), 3(b) e 3(c). A Figura 3(a) apresenta um método que implementa o tratamento do evento *ActionPerformed* do componente *ComboBox* da Figura 3(c). A Figura 3(b) apresenta a declaração de variáveis em escopo global. Ambas Figuras, 3(a) e 3(b), estão escritas na linguagem de programação Java e fazem parte da mesma classe, codificada de tal forma que permite gerar a interface gráfica da figura 3(c). Nas figuras 3(a) e 3(b), os números, à esquerda, representam cada linha de código fonte. Eles são meramente ilustrativos e não fazem parte do programa Java.

```

1 private void cbUFActionPerformed(java.awt.event.ActionEvent evt) {
2     lbSaida1.setText(Integer.toString(cbUF.getSelectedIndex()));
3     lbSaida2.setText((String)cbUF.getSelectedItem());
4 }
    
```

Figura 3(a): Programa em Java - Método cbUFActionPerformed.

```

1 private javax.swing.JComboBox cbUF;
2 private javax.swing.JLabel lbSaida1;
3 private javax.swing.JLabel lbSaida2;
4 private javax.swing.JLabel lbUF;
    
```

Figura 3(b): Programa em Java - Variáveis globais.

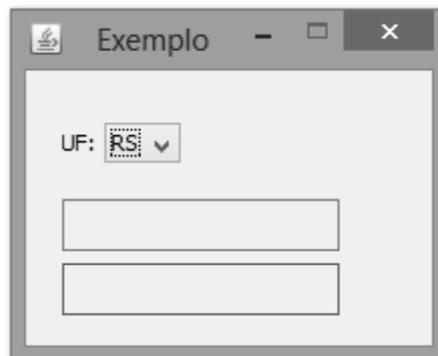


Figura 3(c): Interface gerada pelo programa Java.

35. Considerando na interface gráfica da Figura 3(c), UF como o componente lbUF, o retângulo ao lado de UF como o componente cbUF, que contém os valores RS e SC, sendo que RS é o primeiro da lista e apenas ele está exibido na Figura 3(c), e os demais retângulos como os componentes lbSaida1 e lbSaida2, de cima para baixo, lbSaida1 depois lbSaida2, ao clicar com o mouse no componente cbUF e selecionar RS, no componente lbSaida1 é exibido

- a) 0 e no componente lbSaida2 é exibido RS.
- b) 1 e no componente lbSaida2 é exibido RS.
- c) RS e no componente lbSaida2 é exibido RS.
- d) 1 e no componente lbSaida2 é exibido SC.

36. Considerando as Figuras 3(a) e 3(b), a alternativa correta é

- a) `lbSaida1` é um componente da biblioteca `awt` e, no parâmetro do método `setText` da instrução da linha 2, Figura 3(a), é realizado uma operação de *typecast*.
- b) `lbSaida1` é um componente da biblioteca `swing` e, no parâmetro do método `setText` da instrução da linha 2, Figura 3(a), é realizado uma operação de *typecast*.
- c) `lbSaida1` é um componente da biblioteca `swing` e, no parâmetro do método `setText` da instrução da linha 2, Figura 3(a), é realizado uma operação de *parse*.
- d) `lbSaida1` é um componente da biblioteca `awt` e, no parâmetro do método `setText` da instrução da linha 2, Figura 3(a), é realizado uma operação de *parse*.

A questão 37 baseia-se na Figura 4. Essa figura apresenta um método para inserir dados em um arquivo do tipo texto (`txt`), implementado na linguagem de programação Java. Nessa figura os números à esquerda representam cada linha de código fonte. Eles são meramente ilustrativos e não fazem parte do programa Java.

```
1 public static void inserir(String output) throws FileNotFoundException, IOException{
2     BufferedWriter out = new BufferedWriter(new FileWriter("teste.txt", true));
3     BufferedReader in = new BufferedReader(new FileReader("teste.txt"));
4     String entrada = in.readLine();
5     if(entrada != null){
6         out.newLine();
7         out.write(output);
8         out.close();
9     }
10    else{
11        out.write(output);
12        out.close();
13    }
14    in.close();
15 }
```

Figura 4: Programa em Java - Método inserir.

37. Com base na figura 4, a variável `out`, na linha 2, recebe uma referência para um objeto que estabelece um fluxo de

- a) Bytes para o arquivo `teste.txt`.
- b) Caracteres para o arquivo `teste.txt`.
- c) Objetos para o arquivo `teste.txt`.
- d) Inteiros para o arquivo `teste.txt`.

As questões 38 e 39 baseiam-se nas Figuras 5(a), 5(b), 5(c) e 5(d). Todas apresentam classes implementadas na linguagem de programação Java e fazem parte do mesmo programa. Nessas figuras os números à esquerda representam cada linha de código fonte. Eles são meramente ilustrativos e não fazem parte do programa Java.

```

1  package animal;
2  public abstract class Animal {
3      public abstract void falar();
4  }
    
```

Figura 5 (a): Programa em Java - Classe Animal.

```

1  package animal;
2  public class Cachorro extends Animal {
3      @Override
4      public void falar(){
5          System.out.println("Au, Au, Au ...");
6      }
7  }
    
```

Figura 5(b): Programa em Java - Classe Cachorro.

```

1  package animal;
2  public class Gato extends Animal{
3      @Override
4      public void falar(){
5          System.out.println("Miau, Miau, Miau ...");
6      }
7  }
    
```

Figura 5(c): Programa em Java - Classe Gato.

```

1  package animal;
2  public class Principal {
3      public static void main(String[] args){
4          Animal[] arrayDeAnimais = new Animal[2];
5          arrayDeAnimais[0] = new Cachorro();
6          arrayDeAnimais[1] = new Gato();
7          for(int i=0; i <= 1; i++){
8              arrayDeAnimais[i].falar();
9          }
10     }
11 }

```

Figura 5(d): Programa em Java - classe Principal.

38. Considerando apenas as classes das Figuras 5(a), 5(b) e 5(c), afirma-se que a classe

- a) Gato é a superclasse e o relacionamento dela com as demais classes é de composição.
- b) Animal é a superclasse e o relacionamento dela com as demais classes é de composição.
- c) Animal é a superclasse e o relacionamento dela com as demais classes é de herança.
- d) Cachorro é a superclasse e o relacionamento dela com as demais classes é de herança.

39. Considerando todo o programa, ilustrado nas Figuras 5(a), 5(b), 5(c) e 5(d), ele representa um exemplo do conceito de

- a) agregação.
- b) polimorfismo.
- c) herança múltipla.
- d) abstração.

A questão 40 baseia-se na Figura 6, que apresenta um método implementado na linguagem de programação Java. Escrito para um programa que acessa uma base de dados, mais especificamente a tabela tbContatos. Nesta figura os números à esquerda representam cada linha de código fonte. Eles são meramente ilustrativos e não fazem parte do programa Java.

```
1 public Contato consultar(long id, Contato cto){
2     String sql = "SELECT * FROM tbcontatos WHERE id = " + id;
3     try (Statement stmt = this.connection.createStatement()) {
4         try (ResultSet rset = stmt.executeQuery(sql)); {
5             rset.next();
6             cto.setId(rset.getLong("id"));
7             cto.setNome(rset.getString("nome"));
8             cto.setTelefone(rset.getString("telefone"));
9             return cto;
10        } catch (SQLException ex) {
11            System.out.println("SQLException em consultar!!!" + "Erro
12                detectado: " + ex.getMessage());
13            return null; }
14    } catch (SQLException ex) {
15        System.out.println("SQLException em consultar!!!" + "Erro detectado: " +
16            ex.getMessage());
17        return null; }
18 }
```

Figura 6: Trecho de um programa Java - método consultar.

- 40.** Tendo em vista o método ilustrado na Figura 6 e considerando que as classes utilizadas em seu desenvolvimento pertencem à biblioteca de códigos da linguagem Java 7, afirma-se que o objeto stmt pertence à classe
- ResultSet da biblioteca Java 7, e o objeto rset pertence à classe Statement desta mesma biblioteca.
 - Connection da biblioteca Java 7, e o objeto rset pertence à classe Statement desta mesma biblioteca.
 - Connection da biblioteca Java 7, e o objeto rset pertence à classe ResultSet desta mesma biblioteca.
 - Statement da biblioteca Java 7, e o objeto rset pertence à classe ResultSet desta mesma biblioteca.