Nível Superior

UNIVERSIDADE PÚBLICA COMPROMISSO SOCIAL
UFBA

UFRB
UNIVERSIDADE FEDERAL DO RECÔNCAVO DA BAHIA

Concurso Público para Servidor Técnico-Administrativo UFBA e UFRB 2009

Analista de Tecnologia da Informação

UNIVERSIDADE FEDERAL DA BAHIA • PROGRAD/SSOA
Rua João das Botas, nº 31 - Canela • CEP: 40110-160
Salvador - Bahia - Brasil • Telefax: (__71) 3283-7820
www.concursos.ufba.br • ssoa@ufba.br

INSTRUÇÕES

Para a realização desta prova, você recebeu este Caderno de Questões.

1. Caderno de Questões

- Verifique se este Caderno de Questões contém a prova de Conhecimentos Específicos **referente ao cargo a que você está concorrendo**:

  CONHECIMENTOS ESPECÍFICOS — Questões de 101 a 130

- Qualquer irregularidade constatada neste Caderno de Questões deve ser imediatamente comunicada ao Fiscal de sala.
- Neste Caderno, você encontra apenas um tipo de questão: objetiva de proposição simples. Identifique a resposta correta, marcando na coluna correspondente da Folha de Respostas:

  V, se a proposição é verdadeira;
  F, se a proposição é falsa.

ATENÇÃO: Antes de fazer a marcação, avalie cuidadosamente sua resposta.

LEMBRE-SE:

- A resposta correta vale 1 (um), isto é, você **ganha** 1 (um) ponto.
- A resposta errada vale –0,75 (menos setenta e cinco centésimos), isto é, você **não ganha** o ponto da questão que errou e ainda **perde**, em cada resposta errada, 0,75 (setenta e cinco centésimos) dos pontos ganhos em outras questões que você acertou.
- A ausência de marcação e a marcação dupla ou inadequada valem 0 (zero). Você **não ganha nem perde nada**.

2. Folha de Respostas

- Você terá uma única Folha de Respostas para a Prova de Conhecimentos Gerais e para esta Prova de Conhecimentos Específicos.
- NÃO AMASSE, NÃO DOBRE, NÃO SUJE, NÃO RASURE ESSA FOLHA DE RESPOSTAS.
- A marcação da resposta deve ser feita preenchendo-se o espaço correspondente com caneta esferográfica de tinta **PRETA**. Não ultrapasse o espaço reservado para esse fim.

- **O tempo disponível para a realização das duas provas e o preenchimento da Folha de Respostas é de 5 (cinco) horas no total.**

PROVA DE CONHECIMENTOS ESPECÍFICOS

ANALISTA DE TECNOLOGIA DA INFORMAÇÃO

QUESTÕES de 101 a 130

INSTRUÇÃO:

Para cada questão, de **101** a **130**, marque na coluna correspondente da Folha de Respostas:

V, se a proposição é verdadeira;
F, se a proposição é falsa.

A resposta correta vale 1 (um); a resposta errada vale –0,75 (menos setenta e cinco centésimos); a ausência de marcação e a marcação dupla ou inadequada valem 0 (zero).

Questão 101

As **Abordagens Evolucionárias** de desenvolvimento de *software* permitem determinar, de forma precisa, o número de ciclos necessários para a construção do produto.

Questão 102

O produto resultante de cada iteração do processo de *software Rational Unified Process* — RUP — é uma versão executável do sistema.

Questão 103

Em sistemas grandes e complexos, a determinação das conexões entre os requisitos torna-se geralmente uma tarefa difícil, sendo que as **Tabelas de Rastreamento** constituem uma ferramenta que ajuda a minimizar essa dificuldade.

Questão 104

Numa arquitetura cliente-servidor do tipo **Cliente-Magro**, as alterações no *software* da aplicação obrigam a sua reinstalação em cada computador cliente.

Questão 105

No processo de *software* baseado em componentes, cada componente projetado para **reuso** é uma entidade executável independente, que deve manipular exceções.

Questão 106

A probabilidade total de falha de um sistema que possui N componentes distintos e independentes é dada pelo produto das probabilidades de falha desses componentes.

Questão 107

A programação de **N-versões** utilizada em sistemas críticos caracteriza-se pela implementação de uma série de versões de código referentes a uma mesma especificação de *software*, as quais são executadas de forma paralela em computadores distintos.

Questão 108

Os componentes de um **executivo de tempo-real** típico são o relógio de tempo real, o manipulador de interrupções, o escalonador, o gerenciador de recursos e o processador.

Questão 109

Diante da possibilidade real de os usuários de sistemas interativos cometerem erros que configuram ações destrutivas, o projeto de **interfaces** deve contemplar mecanismos que permitam reverter os danos causados por essas ações.

Questão 110

Os testes de *software* **Caixa-Branca** examinam o comportamento interno do componente de *software.*

Questão 111

A **Associação de Inclusão** geralmente é usada quando existe um serviço, situação ou rotina comum a mais de um Caso de Uso.

Questão 112

O **Estereótipo <<*boundary*>>** identifica uma classe que serve de comunicação entre atores externos ao sistema.

Questão 113

O operador UNION combina os resultados de duas ou mais consultas em um único *result set*, retornando todas as linhas pertencentes a todas as consultas envolvidas na execução, sendo que, para evitar a ocorrência de linhas duplicadas no *result set* final, o filtro DISTINCT deve ser usado.

Questão 114

Em um Diagrama de Gráfico de Estados, um **Estado de História** representa o registro do último subestado em que um objeto se encontrava, quando, por algum motivo, o processo foi interrompido.

QUESTÕES de 115 a 118

Considerem-se os comandos SQL, a seguir:

```
IF EXISTS (SELECT 1 FROM SYSOBJECTS
    WHERE TYPE = 'U' and NAME = 'Pessoa')
   DROP TABLE Pessoa
Go

CREATE TABLE Pessoa (Codigo CHAR(5)         NOT NULL,
                     Nome   VARCHAR(50) NOT NULL,
                     Data_Nasc                DATETIME,
    CONSTRAINT PK_COD_PESSOA PRIMARY KEY (Codigo))

INSERT INTO Pessoa VALUES('11111', 'Paloma Freitas', '20-12-1999')
INSERT INTO Pessoa VALUES('44444', 'Ricardo Prado',  '20-12-2006')
INSERT INTO Pessoa VALUES('55555', 'Milena Souza',   '20-12-2006')
INSERT INTO Pessoa VALUES('22222', 'Telma Pachiolle','20-12-2000')
INSERT INTO Pessoa VALUES('33333', 'Victor Marques', '20-12-2004')
INSERT INTO Pessoa VALUES('66666', 'Juliana Rocha',  '22-01-1978')
INSERT INTO Pessoa VALUES('88888', 'Márcio Lopes',   '05-04-1975')
INSERT INTO Pessoa VALUES('77777', 'Tânia Regina',   '31-12-2007')
```

Questão 115

A Cláusula "DROP TABLE Pessoa" apagará do banco a tabela **Pessoa**, caso ela já exista no mesmo, no entanto, em situação oposta, uma mensagem de erro será exibida pelo SQL.

Questão 116

A execução do comando "SELECT * FROM Pessoa" exibirá todos os registros inseridos na tabela **Pessoa**, na mesma ordem em que foram inseridos.

Questão 117

A execução do comando SQL mostrado a seguir

```
SELECT Codigo AS Código,
       Nome,
       DATEDIFF (yy,Data_Nasc,GETDATE()) AS 'Idade(Anos)'
FROM PESSOA WHERE DATEDIFF (yy,Data_Nasc,GETDATE()) > 3
ORDER BY DATEDIFF (yy,Data_Nasc,GETDATE())
```

exibirá o seguinte resultado:

Código	Nome	Idade (Anos)
33333	Victor Marques	5
22222	Telma Pachiolle	9
11111	Paloma Freitas	10
66666	Juliana Rocha	31
88888	Márcio Lopes	34

(5 row(s) affected)

Questão 118

A execução do seguinte comando SQL

```
SELECT Codigo AS Código,
       Nome,
       DATEDIFF (yy,Data_Nasc,GETDATE()) AS Idade
FROM PESSOA WHERE DATEDIFF (yy,Data_Nasc,GETDATE()) BETWEEN 9 and 31
ORDER BY Data_Nasc DESC
```

exibirá o seguinte resultado:

Código	Nome	Idade (Anos)
11111	Paloma Freitas	10

(1 row(s) affected)

Questão 119

Em um relacionamento, a **cardinalidade mínima** de uma entidade com participação parcial é **um**.

Questão 120

Um *thread* pode ceder seu tempo de processamento a outro *thread* de prioridade mais baixa.

Questão 121

Um *thread deamon* impede o programa de terminar.

Questão 122

Atribuir *null* à referência de um objeto marca esse objeto para a coleta de lixo, caso não haja outra referência ao objeto.

Questão 123

O bloqueio do objeto que ocorre com a execução dos métodos *synchronized* poderia levar a um *deadlock*, se os bloqueios não fossem liberados nunca.

Questão 124

O bloco *finally* em uma instrução *try – catch – finally* sempre será executado quer ocorra ou não uma exceção no bloco *try*.

Questão 125

FlowLayout, *BorderLayout* e *GridXYLayout* são gerenciadores de *layout* da interface gráfica.

Questão 126

ActionListener e *MouseListener* são algumas das interfaces ouvintes de eventos do pacote java.awt.event.

QUESTÕES de 127 a 130

```
import java.util.ArrayList;
import java.util.Iterator;

public class Teste {
  public int w;
}
class Teste2 extends Teste {
  public int y;
}

class Teste3 {
  public static Teste2  z;

  public static void main(String x[]) {

      ArrayList<Teste>  lista = new ArrayList<Teste>();
      Iterator it;

      Teste   a = new Teste();
      Teste2  b = new Teste2();
      Teste3  c = new Teste3();

      lista.add(a);
      lista.add(b);
      lista.add(c);

      for (Teste o : lista) {
        System.out.println(o.y);
      }
  }
}
```

Questão 127

O trecho de código apresentado contém vários erros, sendo que um deles ocorre porque o objeto “it” é utilizado no comando “for”.

Questão 128

A coleção **lista** somente pode aceitar objetos da classe **Teste** e de seus descendentes na chamada do método ***add***.

Questão 129

O comando *System.out.println(o.y)* será executado sem problemas, já que “y” é um atributo de uma classe descendente de **Teste**.

Questão 130

Uma maneira de inicializar o objeto “it” é it = lista.iterator();

Universidade Federal da Bahia