CONHECIMENTOS ESPECÍFICOS

A respeito de engenharia de *software*, julgue os itens a seguir.

61 O modelo *Waterfall* tem a vantagem de facilitar a realização de mudanças sem a necessidade de retrabalho em fases já completadas.

62 Manutenibilidade, confiabilidade, desempenho e usabilidade estão entre os principais atributos de um produto de *software*.

63 O processo de desenvolvimento de *software* é uma caracterização descritiva ou prescritiva de como um produto de *software* deve ser desenvolvido.

64 No modelo de prototipação, o processo de desenvolvimento de *software* é modelado como uma sequência linear de fases, enfatizando um ciclo de desenvolvimento de breve duração.

Julgue os itens que se seguem, relativos à arquitetura cliente/servidor.

65 Do ponto de vista da implementação, os servidores são classificados como interativos ou concorrentes. Em geral, os servidores que usam protocolos orientados a conexão são interativos; os que usam protocolos orientados a datagrama são concorrentes.

66 Um servidor interativo que receber solicitação de serviço retornará ao estado de espera após o tratamento e o envio do resultado desse serviço para o solicitante.

67 No servidor concorrente, para se tratar cada solicitação recebida, é criada nova instância do servidor, a qual pode ser um novo processo ou um *thread*.

68 A arquitetura cliente/servidor proporciona a sincronização entre duas aplicações: uma aplicação permanece em estado de espera até que outra aplicação efetue uma solicitação de serviço.

Acerca de arquitetura de aplicações para Internet e *web*, julgue os itens subsecutivos.

69 Nas arquiteturas de três níveis, a lógica de negócio é executada em um servidor de banco de dados localizado em um *host* distinto daquele que executa o servidor HTTP.

70 Na arquitetura *peer-to-peer* (P2P) pura, as aplicações podem assumir simultaneamente os papéis de cliente e de servidor, o que lhes confere grande escalabilidade e facilidade de gerenciamento.

71 Em uma arquitetura P2P híbrida, a transferência de dados entre as partes ocorre diretamente, porém há o registro em servidores centrais, o que facilita a busca de pares e de conteúdo.

72 Duas características comuns à maioria dos serviços *web* são o uso do protocolo HTTP — para envio das solicitações, processamento e envio de resultados em HTML — e o uso de *logs*, também no formato HTTP.

73 Nas arquiteturas cliente/servidor de dois níveis, a aplicação tem uma lógica de apresentação, executada no *browser*, mas a lógica de negócio deve ser executada em outra máquina diferente do servidor HTTP.

Com relação à arquitetura OLAP, julgue os itens seguintes.

74 Um membro derivado de uma dimensão é o de nível mais baixo na hierarquia.

75 *Drill up* e *down* são técnicas analíticas em que o usuário transita entre vários níveis de agregação dos dados, indo do mais sumarizado ao mais detalhado e vice-versa.

76 Os caminhos usados no *drilling* são definidos pelas hierarquias dentro das dimensões ou por outros relacionamentos, que podem ser dinâmicos entre dimensões ou dentro delas.

77 Um cubo, ou hipercubo, é um *array* multidimensional no qual um grupo de células de dados é organizado segundo as dimensões dos dados.

78 A dimensão, um atributo estrutural de um cubo, consiste em uma lista de membros, todos eles de um tipo de dados similar na percepção do sistema.

No que se refere à arquitetura SOA e *web services*, julgue os itens a seguir.

79 As transações são os blocos básicos sobre os quais as aplicações embasadas em SOA são construídas.

80 *Web services* são sistemas de *software* projetados para suportar interoperabilidade *host-to-host* sobre a rede.

81 A interoperabilidade é conseguida utilizando-se um conjunto de padrões proprietários embasado em XML.

82 A arquitetura SOA utiliza um paradigma *find-bind-execute* no qual os provedores cadastram seus serviços em um registro público; os consumidores acessam o registro em busca de serviços; e, se houver serviço disponível, o registro devolve ao consumidor um contrato e um endereço para aquele serviço.

83 Aplicações embasadas em SOA são distribuídas e *multi-tier* e possuem lógicas de apresentação e negócio e camadas de persistência.

Julgue os próximos itens, a respeito de orientação a objetos.

84 Encapsulamento consiste em publicar o estado interno de um objeto, exigindo que toda a interação seja executada por meio dos métodos do objeto.

85 O mecanismo de herança, identicamente ao de composição, faz com que uma classe herde o estado e o comportamento no sentido ascendente da hierarquia de classes.

86 Modularidade e reúso de código são alguns dos benefícios da orientação a objetos.

87 Uma classe que implementa uma interface compromete-se a prover o comportamento publicado por aquela interface.

88 Objetos constituem-se de estado e comportamento: o estado armazenado em campos ou variáveis, e o comportamento exposto por meio de métodos, que operam sobre o estado interno e servem como mecanismo primário de comunicação entre objetos.

```
class Bicycle {
int cadence = 0;
int speed = 0;
int gear = 1;
void changeCadence(int newValue) {
cadence = newValue;
}
void changeGear(int newValue) {
gear = newValue;
}
void speedUp(int increment) {
speed = speed + increment;
}
void applyBrakes(int decrement) {
speed = speed - decrement;
}
void printStates() {
System.out.println("cadence:"+cadence+"speed:
"+speed+"gear:"+gear);
}
}
class BicycleDemo {
public static void main(String[] args) {
Bicycle bike1 = new Bicycle();
Bicycle bike2 = new Bicycle();
bike1.changeCadence(50);
bike1.speedUp(10);
bike1.changeGear(2);
bike1.printStates();
bike2.changeCadence(50);
bike2.speedUp(10);
bike2.changeGear(2);
bike2.changeCadence(40);
bike2.speedUp(10);
bike2.changeGear(3);
bike2.printStates();
}
}
```

Considerando o trecho de programa Java mostrado acima, julgue os itens seguintes.

89 Ao final da execução desse trecho de programa, serão impressos os seguintes valores para `bike2`.
`cadence:50 speed:20 gear:3`

90 De acordo com o trecho de programa em questão, o mecanismo de herança foi bloqueado pela redefinição de atributos na subclasse criada.

91 O trecho de programa em apreço define instâncias da classe `Bicycle`.

92 Ao final da execução desse trecho de programa, serão impressos os seguintes valores para `bike1`.
`cadence:50 speed:10 gear:2`

```
<script language="php">
if($react == "delete_user") {
if($user) {
$query = "DELETE from login WHERE user='$user'";
$result = mysql_query($query, $mysql_link);
if(mysql_num_rows($result)) {
print("<strong>$user</strong> successfully
deleted<p>");
}
} else {
print("<strong>no users are available to delete
.</strong><p>");
}
}
elseif ($react =="add_user") {
if(($user) and ($pass)) {
$query = "INSERT into login VALUES (";
$query .= "0, SYSDATE(),'$username','$password' )";
mysql_query($query, $mysql_link);
} else {
print("<strong>either your user or password field
was left blank</strong><p>");
}
}
else {
print("<center>Administration Area - Choose your
option</center>");
}
</script>
```

Considerando o trecho de programa PHP apresentado acima, julgue os itens subsequentes.

93 Não há crítica aos dados providos pelo usuário.

94 Os pares usuário e senha são armazenados em um banco de dados.

95 As informações são cifradas antes de sua inserção no banco de dados.

Ferramentas CASE (*computer-aided software engineering*) são *softwares* utilizados para apoiar as atividades do processo de *software*, como, por exemplo, engenharia de requisitos, projeto, teste etc. Julgue os itens seguintes, acerca dessas ferramentas.

96 Geradores de referência cruzada, analisadores estáticos e analisadores dinâmicos são exemplos de ferramentas CASE de análise de programa.

97 Compiladores e processadores são exemplos de ferramentas CASE de gerenciamento de configuração.

98 Linguagens de nível muito alto e geradores de interface com o usuário são exemplos de ferramentas CASE de prototipação.

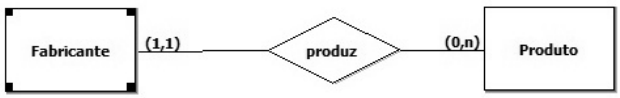
Medição é o processo por meio do qual números ou símbolos são atribuídos a entidades do mundo real, de forma a quantificá-las, ou seja, é o processo de obtenção de medidas para entidades do mundo real. Julgue os itens a seguir, relativos a análise de pontos de função (APF) e a análise por pontos de caso de uso (APCU).

99 A tabela de fatores de complexidade técnica permite avaliar o impacto de fatores como atualização *online*, volume de transações e entrada de dados *online*.

100 A APF, uma técnica de medição das funcionalidades fornecidas pelo *software*, de acordo com a percepção do usuário, baseia-se na avaliação dos requisitos lógicos; a APCU é uma técnica de medição dos casos de uso e, diferentemente da APF, possibilita, também, a identificação do esforço.

101 A contagem da APF considera fatores de complexidade ambiental e fatores de complexidade funcional; a contagem da APCU trata as características gerais de sistema.

102 São funções do tipo transação: entradas externas, saídas externas e consultas externas. Uma das principais diferenças entre as saídas externas e as consultas externas é que as primeiras devem conter alguma fórmula matemática ou cálculo, enquanto as consultas externas representam uma recuperação simples de dados.



Com base no modelo acima, julgue os itens que se seguem, referentes aos conceitos de modelagem de dados.

103 O Fabricante pode não produzir nenhum Produto.

104 O Fabricante pode produzir vários produtos, mas o Produto deve estar vinculado a um Fabricante.

105 O Produto pode existir sem estar vinculado a nenhum Fabricante, mas este tem de produzir pelo menos um Produto.

Tendo em vista que, de acordo com seu conceito corrente, a normalização visa refinar o modelo de dados em relação às anomalias de atualização, julgue os itens subsequentes, relativos a modelagem de dados e normalização.

106 A terceira forma normal será aplicada ao caso em que alguma entidade possua chave primária concatenada e haja algum atributo que apresente dependência parcial dessa chave.

107 A primeira forma normal afirma que cada ocorrência da chave primária deve corresponder a uma e somente uma informação de cada atributo, ou seja, a entidade não deve conter grupos repetitivos.

108 A segunda forma normal analisa a dependência transitiva, ou seja, se um conjunto de atributos depende de outro atributo que não pertence à chave primária.

Os requisitos de um sistema definem o que esse sistema deve fazer, bem como estabelecem as restrições de operação e implementação desse sistema. Acerca de requisitos funcionais e não funcionais, julgue os próximos itens.

109 Os requisitos não funcionais não estão relacionados diretamente às funções específicas fornecidas pelo sistema. Definições de desempenho, espaço e portabilidade são exemplos de requisitos não funcionais.

110 A frase a seguir é um exemplo de descrição de requisito funcional: A interface de usuário do sistema xxx deve ser implementada como simples HTML, sem *frames* ou *applets* Java.

111 Os requisitos funcionais descrevem o que o sistema deve fazer; dependem do tipo de *software* que está sendo desenvolvido, dos usuários e da abordagem geral utilizada pela organização para redigir os requisitos.

Considerando que a validação de requisitos permite demonstrar que os requisitos refletem o sistema que se deseja construir, julgue os itens subsequentes.

112 A revisão de requisitos objetiva verificar o documento de requisitos em busca de anomalias ou omissões; é um processo manual que envolve cliente e fornecedor.

113 Na prototipação, um modelo executável do sistema é apresentado aos usuários e clientes finais, que podem verificar se o modelo atende as suas necessidades reais.

114 Verificações de realismo buscam averiguar se os requisitos não estão conflitantes, enquanto verificações de consistência, considerando a tecnologia existente, visam analisar a viabilidade de implementação desses requisitos.

Sabendo que entrevistas e reuniões são técnicas utilizadas para a obtenção de requisitos de um *software*, julgue os itens a seguir.

115 A coleta inicial dos requisitos ocorre normalmente em reunião que participam os *stakeholders*, gerentes, desenvolvedores e todas as pessoas envolvidas com o projeto. As regras e agenda da reunião devem ser comunicadas para todos os participantes. Os tópicos a serem discutidos nessa coleta inicial incluem identificação do problema, necessidade e justificativa do novo produto de *software*.

116 As entrevistas podem ser classificadas como abertas ou fechadas. Nas entrevistas fechadas, não há roteiro predefinido. Na prática, é possível implementar uma combinação de entrevistas abertas e fechadas, visando obter uma compreensão mais ampla das necessidades dos *stakeholders* com relação ao sistema.

117 São considerados facilitadores para uma reunião: direcionar a discussão, encerrar a conversação e preparar um histórico, à medida que a reunião acontece, que vai ajudar na consolidação dos resultados e na identificação das próximas etapas.

Considerando que processo de *software* pode ser definido como um conjunto de atividades inter-relacionadas que transformam insumos (entradas) em produtos (saídas), julgue os itens que se seguem.

118 As áreas de processo medição e análise, gestão de acordo com fornecedores e planejamento do projeto pertencem ao nível de maturidade 3, gerenciado do CMMI-Dev; o nível de maturidade 2 do CMMI-Dev engloba as áreas de processo: gestão de requisitos, monitoramento e controle de projeto, garantia da qualidade de processo e produto e gestão de configuração.

119 A *extreming programming* (XP) é considerada um método ágil, em que todos os requisitos são expressos por meio de cenários. O ciclo de *release* em XP engloba: selecionar as histórias dos usuários para implementação na versão, dividir as histórias em tarefas, planejar a versão, desenvolver/construir e testar o *software*, liberar o *software* e avaliar o sistema.

120 São características de teste na XP: desenvolvimento *test-first*, desenvolvimento incremental de testes a partir de cenários, envolvimento do usuário no desenvolvimento e validação de testes e o uso de ferramentas de teste automatizadas.

121 O *framework scrum* engloba conceitos como *times scrum*, eventos com duração fixa (*time-boxes*), artefatos e regras. São exemplos de eventos que têm duração fixa: a reunião de planejamento da versão para entrega, a *sprint*, a reunião diária, a revisão da *sprint* e a retrospectiva da *sprint*.

122 O RUP é um modelo de processo que considera fases. Na fase de elaboração, é estabelecido um *business case* para o sistema e são definidas entidades externas que irão interagir com esse sistema.

123 Na fase do RUP de construção, são realizados a programação e o teste do sistema; a fase de transição corresponde à transferência do sistema da comunidade de desenvolvedores para a comunidade de usuários.

124 O CMMI-Dev e o MPS.BR têm como objetivos definir e aprimorar um modelo de melhoria e avaliação de processo de *software*. Ambos os modelos baseiam-se em conceitos de maturidade e processos.

Tendo em vista que as características de qualidade do produto de *software* permitem identificar uma série de fatores de influenciam na avaliação de um produto de *software*, julgue os itens de **125** a **127**.

125 A usabilidade representa a facilidade de utilizar o produto. Características como atratividade, compreensibilidade, eficiência de uso, facilidade de memorização e apreensibilidade, permitem orientar uma avaliação nesse contexto.

126 A característica portabilidade avalia a operação do produto em ambientes com características diferentes. Nesse sentido, podem-se avaliar aspectos como adaptabilidade, coexistência e instalabilidade.

127 A característica eficiência refere-se à capacidade de um *software* manter certo nível de desempenho quando estiver operando em determinado contexto de uso. Nesse contexto, é permitido haver a avaliação de aspectos como tolerância à falha e recuperabilidade.

Considerando que o modelo relacional representa o banco de dados como uma coleção de relações ou tabelas e que uma tabela é um conjunto não ordenado de tuplas ou linhas, julgue o item abaixo.

128 A relação entre linhas de tabelas de um banco de dados relacional é implementada por meio de chave. Em um banco de dados relacional, existem, no mínimo, dois tipos de chaves a considerar: a chave primária e a chave estrangeira. A chave primária é uma coluna ou uma combinação de colunas cujos valores distinguem uma linha das demais dentro de uma tabela, enquanto uma chave estrangeira é uma coluna ou uma combinação de colunas, cujos valores aparecem, necessariamente, na chave primária de outra tabela.

```
CREATE TABLE ITEM-DO_PEDIDO
(
Num_pedido          int  not null unique,
Codigo_produto      smallint   not null unique,
Quantidade           decinal,
FOREIGN KEY         (num_pedido)
                    REFERENCES PEDIDO,
                     (código_produto)
                    REFERENCES PRODUTO
);
```

Com base nos comandos SQL acima, julgue os itens a seguir.

129 A cláusula `REFERENCES` estabelece a restrição de chave entre as tabelas. A chave estrangeira já está definida juntamente com o registro da tabela.

130 Quando esses comandos forem executados, ocorrerão erros, pois as tabelas `PRODUTO` e `PEDIDO` deveriam ter sido criadas antes da execução desses comandos.