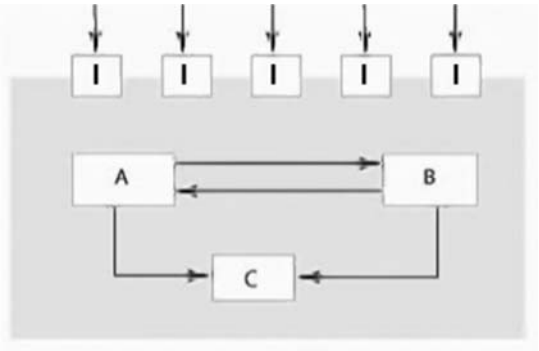
-- CONHECIMENTOS ESPECÍFICOS --

Julgue os próximos itens, relativos a conceitos, aplicações e tipos de testes de *software*.

51 No processo de testes para *software* customizado, deve haver pelo menos um teste para cada requisito; nessa validação, busca-se que o sistema execute corretamente de acordo com o uso esperado.

52 Em testes automatizados, para diminuir problemas de lentidão ao se acessar um banco de dados no teste de funcionalidade, pode-se substituir o banco de dados por um *mock objects*.

53 Considerando-se a figura a seguir — em que **I** significa interface, e os componentes **A**, **B** e **C** foram integrados para criar um subsistema —, é correto afirmar que, nesse cenário, os testes de componentes compostos devem mostrar se a interface de componente se comporta de acordo com sua especificação.



54 Em um desenvolvimento ágil, convém que os testes sejam automatizados para cada objeto no ciclo de vida do produto, o que inclui código-fonte e alterações de banco de dados.

55 Testes ágeis devem ser implementados quando há integração contínua; nesse caso, os testes devem ser aplicados somente ao final de cada *sprint*, de modo a validar se há integração de objetos distintos, ainda que isso não ajude a garantir que todo o sistema esteja funcionando corretamente.

56 O desenvolvimento orientado a testes (TDD — *test driven development*) agrega uma técnica de *design* e análise em que a funcionalidade de teste vem como um valor agregado, uma vez que os desenvolvedores tentam entender o objeto que estão prestes a construir, concentrando-se nos resultados esperados da funcionalidade.

A seguir, são apresentados dois arquivos, um escrito em XML e outro escrito em XSLT. Nesses arquivos, foram inseridos números seguidos de ponto (.) apenas para indicar as suas linhas. A tabela a seguir ilustra o resultado esperado quando se utilizam esses arquivos.

arquivo XML

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <garagem>
3.     <carro>
4.        <modelo>ModeloA</modelo>
5.        <marca>MarcaA</marca>
6.     </carro>
7.     <carro>
8.       <modelo>ModeloB</modelo>
9.       <marca>MarcaB</marca>
10.    </carro>
11.</garagem>
```

arquivo XSLT

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <xsl:stylesheet version="1.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
3. <xsl:template match="/">
4. <html>
5. <body>
6.    <table border="1">
7.      <tr>
8.        <th>Modelo</th>
9.        <th>Marca</th>
10.       </tr>
11.       <xsl:ABCD select="garagem/carro">
12.       <tr>
13.          <td><xsl:value-of
   select="modelo"/></td>
14.         <td><xsl:value-of
   select="marca"/></td>
15.       </tr>
16.       </xsl:ABCD>
17.     </table>
18.   </body>
19.   </html>
20.   </xsl:template>
21.   </xsl:stylesheet>
```

resultado esperado

modelo	marca
ModeloA	MarcaA
ModeloB	MarcaB

Tendo como referência as informações precedentes, julgue os itens seguintes, a respeito de XML e XSLT.

57 Para a obtenção do resultado esperado, a linha 3 do arquivo XSLT deve ser substituída pela linha a seguir, pois o atributo `match` deve conter o elemento `garagem` do arquivo XML.
`<xsl:template match="garagem">`

58 O arquivo XML continuaria sintaticamente correto mesmo se fossem excluídas as linhas 2 e 11.

59 Para que o arquivo XSLT fique sintaticamente correto, os caracteres `ABCD` devem ser substituídos por `get-value` nas linhas 11 e 16.

Acerca dos padrões UDDI, SOAP e REST, julgue os itens a seguir.

60 Para a implementação de serviços REST, é correto utilizar mecanismos padronizados, como o UDDI, para descobrir serviços oferecidos.

61 REST usa um modelo centrado em recursos de serviços encapsulados, em que cada recurso fornecido pelo serviço possui uma URL e todos os recursos oferecem suporte a uma interface uniforme.

62 A seguir, é apresentado um exemplo de UDDI para um serviço hipotético de nome `OperationX`.

```
<binding name="MyBinding" type="tns:abc" >
 <soap:binding style="rpc" transport="http://
schemas.xmlsoap.org/soap/http"/>
 <operation name="OperationX">
  <soap:operation soapAction=""/>
  <input>
     <mime:multipartRelated>
        <mime:part>
           <soap:body parts="part1 part2 ..."
   use="encoded"
namespace="http://mynamespace"
encodingStyle="http://schemas.xmlsoap.org/
soap/encoding"/>
        </mime:part>
        <mime:part>
           <mime:content  part="attch"  type="
text/html"/>
        </mime:part>
     </mime:multipartRelated>
  </input>
 </operation>
</binding>
```

63 O SOAP trata sobre a especificação para a troca de informação entre sistemas utilizando a XML, com suporte a interfaces orientadas a métodos e documentos.

A seguir, é apresentada parte de um arquivo em JSON, em que foram inseridos números seguidos de ponto (`.`) apenas para indicar as linhas.

```
1. [
2.  {
3.       "Fabrica": "FabricaB",
4.       "Pais": "PaisB",
5.       "Carros": [
6.          "CarroA",
7.          "CarroB"
8.       ]
9.  },
10.   {
11.         "Fabrica": "FabricaC",
12.         "Pais": "PaisC",
13.         "Carros": [
14.            "CarroC",
15.            "CarroD"
16.      ]
17.   }
18.]
```

Tendo como referência as informações precedentes, julgue os itens subsecutivos, a respeito de JSON.

64 A sintaxe das linhas 1, 5, 8, 13, 16 e 18 está errada, pois, em JSON, os vários valores em um *array* não devem ser listados entre colchetes, como ocorre nas linhas mencionadas, mas entre parênteses.

65 Considere que a linha 18 atual seja excluída e sejam incluídas, logo após a linha 17, as linhas mostradas a seguir, numeradas de 18 a 21; nesse caso, tem-se um exemplo de como inserir um método no arquivo JSON em questão.

```
(…)
17.  }
18. function inserecarro(nfabrica,ncarro) {
19.    new.nfabrica = "ncarro";
20.  }
21. ]
```

Com relação ao Plone 4, julgue os itens a seguir.

66 Por questões de segurança, não é possível fazer ligação (*link Object*) de arquivos binários nas páginas Plone.

67 O Plone 4 é compatível com as plataformas Linux e Windows e, ainda que não seja compatível com *Active Directory*, pode ser integrado com LDAP, que permite autenticação de grupo e de usuários.

68 Coleção e eventos são tipos de conteúdo que podem ser inseridos no Plone 4: uma coleção funciona como um diretório virtual similar ao OneDrive e ao Google Drive, em que são armazenados arquivos; os eventos são regras inseridas em páginas para alerta sobre alteração e inserção.

Julgue os itens a seguir, a respeito de e-Mag e padrões *web*, sob o ponto de vista das necessidades de acessibilidade e usabilidade.

69 Páginas *web* devem fornecer recursos de impressão amigável via utilização de uma CSS voltada para impressão, devendo-se também evitar a utilização de quadros (*frame*).

70 As URLs não devem funcionar sem o `www`, como no exemplo a seguir, para evitar erro de identificação do conteúdo do sítio (*doctype*).
`http://site.com.br/contato`

71 Convém dividir as áreas de informação da página usando-se, por exemplo, elementos do HTML5; nesse caso, pode-se utilizar o elemento `SECTION` várias vezes na mesma página, diferentemente dos elementos `FOOTER` e `HEADER`, restritos a única inserção por página.

72 Nos padrões *web*, as camadas deverão ser separadas de acordo com o objetivo para o qual elas foram desenvolvidas. Criar páginas em camadas lógicas é uma boa prática: xHTML, folhas de estilo CSS e JavaScript são voltadas, respectivamente, para as camadas de conteúdo, de apresentação visual do conteúdo e de comportamento dos elementos.

73 Para garantir que todos os recursos de tecnologia assistiva realizem a interpretação da informação, é importante utilizar os *landmarks roles* de ARIA, tais como `navigation` e `banner`, juntamente com os elementos estruturais do HTML 5, tais como `<nav>` e `<header>`.

A respeito de arquiteturas orientadas a serviços, *web services* e DevOps, julgue os próximos itens.

74 Em DevOps, ao submeter o código ao sistema de controle de versão, o desenvolvedor utiliza, entre outros, o teste de carga (estresse), que, basicamente, mede e avalia o tempo de resposta, o número de transações e outros requisitos sensíveis ao tempo.

75 Soluções orientadas a serviços devem ser compostas de serviços construídos ou como *web services* ou como componentes.

76 A técnica de integração contínua, de uso fundamental para DevOps, estabelece que o código seja compilado para cada mudança e que sejam executados testes automatizados minimamente confiáveis.

Acerca de desenvolvimento de sistemas *web*, julgue os itens a seguir.

77 A instrução `DOCTYPE` do HTML 5 é mais simples que a das versões anteriores HTML 4 ou XHTML 1.

78 No HTML 5, as *tags* de `link` e `script` usadas para referenciar arquivos de CSS e JavaScript não precisam informar o atributo `type`, porque, na sua ausência, o navegador assume que o arquivo é do tipo `text/css` ou `text/javascript`.

79 No CSS 3, o uso da propriedade `box-shadow` e da função `linear-gradient` possibilita a criação de formas, cores e efeitos das páginas diretamente no código, de modo que os resultados aparecem diretamente nos navegadores.

80 No HTML 5, os novos campos para formulários, como `email`, `search` e `range`, e os atributos, como `placeholder`, `pattern` e `required`, reduzem a necessidade de utilização de *plugins* para auxiliar a formatação dos elementos.

Acerca de gestão da qualidade, julgue os itens subsecutivos.

81 O objetivo do gerenciamento da rotina é tornar cada funcionário individualmente responsável por tudo o que executa, independentemente de fazê-lo em consonância, ou não, com o planejamento estratégico da organização.

82 Embora a qualidade total diminua a quantidade de erros nos processos, porque evita o desperdício de recursos, reduz o tempo da produção e diminui o estresse do trabalhador, ela gera mais custos ao processo como um todo.

83 A gerência da qualidade, além de gerar atitudes e controles que possibilitam a prevenção de problemas, garante que as atividades organizadas aconteçam conforme planejado.

84 Enquanto o TQM (Total Quality Management) está relacionado com a qualidade do ponto de vista do cliente, o TQC (Total Quality Control) relaciona-se com a qualidade total sob a ótica de toda a cadeia produtiva.

Julgue os itens seguintes, a respeito de controle da qualidade e suas principais ferramentas.

85 O diagrama de causa e efeito é utilizado pelo gestor de qualidade para demonstrar a relação entre as causas e os efeitos de um processo.

86 Diagrama de dispersão é uma ferramenta que permite a análise do relacionamento entre duas variáveis para traçar um perfil do comportamento dessa relação com base na concentração e no formato dos pontos.

87 O gráfico de controle pode ser utilizado para analisar se determinada variação está dentro de um padrão médio esperado.

88 O gráfico de Pareto parte do pressuposto de que 80% das causas advêm de 20% dos efeitos, princípio que é conhecido como a regra do 80/20.

Acerca de gestão da qualidade e suas ferramentas de análise e melhoria de processos, julgue os itens a seguir.

89 A ferramenta 5W2H é utilizada para a obtenção de critérios a serem usados na análise e seleção de um projeto ou investimento.

90 Um dos princípios em que se baseia a técnica de dinâmica de grupo conhecida como *brainstorm* é o atraso de julgamento, que possibilita a geração de muitas ideias antes de se decidir por uma.

91 Organograma é o mesmo que fluxograma e pode ser definido como um tipo de diagrama que representa a estrutura funcional de uma organização.

No que se refere a metodologias ágeis e experiência de usuário, julgue os itens que se seguem.

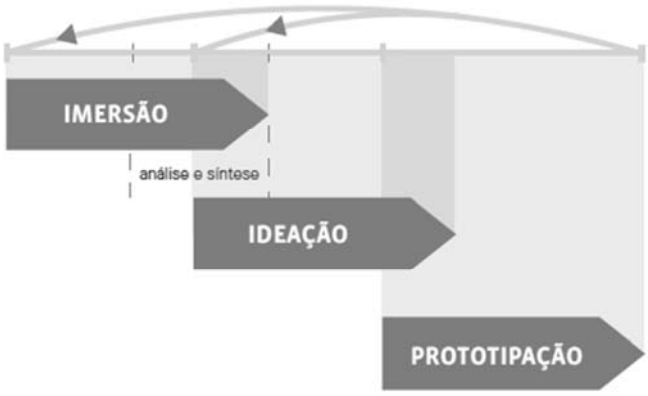
92 O *scrum master* é diretamente responsável por manter e priorizar o *backlog* do produto, além de colaborar com o time de desenvolvimento.

93 Na primeira fase do *design thinking*, chamada de imersão, a equipe de projeto aproxima-se do contexto do problema, do ponto de vista tanto da empresa quanto do usuário.

94 O responsável direto pelo *backlog* da *sprint* é o time de desenvolvimento, que decide sobre as adições e(ou) remoções e os ajustes de tarefas durante a execução da *sprint*; no entanto, se algum item for retirado, o dono do produto deve ser avisado o mais breve possível.

95 No Agile UX, *design*, negócios e desenvolvimento se unem para trabalhar em um processo cíclico, ao passo que, no Lean UX, eles trabalham em paralelo.

96 As etapas do *design thinking* ilustradas na figura a seguir possuem natureza linear, ou seja, as fases moldam o projeto e o problema em questão.



Julgue os próximos itens, a respeito da Estratégia de Governo Digital (EGD), instituída pelo Decreto n.º 10.332/2020.

97 Para a consecução dos objetivos estabelecidos na EGD, os órgãos e as entidades deverão elaborar o Plano de Transformação Digital, o Plano Diretor de Tecnologia da Informação e Comunicação e o Plano de Dados Abertos, os quais serão submetidos à aprovação do comitê de governança digital de cada órgão ou entidade.

98 Para o período de 2020 a 2022, é um dos objetivos da EGD a formação de equipes de governo com competências digitais, sendo uma de suas metas a capacitação de, no mínimo, dois mil profissionais em áreas do conhecimento essenciais para a transformação digital.

99 Integrar todos os estados à rede <gov.br> e ampliar a utilização de *login* único de acesso <gov.br> para mil serviços públicos digitais são algumas das iniciativas contidas nos objetivos da EGD para o período de 2020 a 2022.

Acerca de gerenciamento de projetos, julgue os itens a seguir.

100 No conjunto de conhecimentos em gerenciamento de projetos (PMBOK), os grupos de processos são as fases cronológicas pelas quais o projeto passa, e suas áreas estão divididas em seis áreas de conhecimento de gerenciamento de projetos.

101 No desenvolvimento ágil, a implantação contínua visa reduzir o tempo decorrido entre a gravação de uma linha de código e a disponibilização desse código aos usuários em produção.

102 A área de conhecimento *Project Scope Management* do PMBOK envolve o escopo do projeto, ou seja, o trabalho que está inserido no projeto, sendo as mudanças de escopo uma das principais causas de mudanças nesse projeto.

103 Os métodos ágeis de desenvolvimento de *software* têm duas unidades principais de entrega: lançamentos e iterações.

104 No gerenciamento ágil de projeto, o desenvolvimento orientado para teste de aceitação é uma descrição formal do comportamento de um produto de *software*, geralmente expressa como um exemplo ou cenário de uso.

Julgue os próximos itens a respeito de *Web writing* e UX *writing*.

105 UX *writing* é a arte de elaborar os textos e a linguagem falada voltada para guiar, encantar e reter os usuários de um produto digital.

106 UX *writing* e *Web writing* possuem escritas e objetivos iguais, são apenas nomes ou termos populares no mundo *front-end*, mas com as mesmas finalidades.

107 As habilidades principais que um *designer* de UX *writer* precisa ter incluem habilidades de escrita, estratégia de conteúdo, experiência de usuário, ferramentas de *design*, habilidades de comunicação, curiosidade e agilidade.

No que tange à arquitetura e tecnologias de sistemas de informação, julgue os itens seguintes.

108 Engenharia de *software* (ES) é um processo, expresso como o conjunto de todas as atividades relacionadas ao desenvolvimento, ao controle, à validação e à manutenção de um *software* operacional, abrangendo atividades técnicas e gerenciais.

109 A arquitetura distribuída apresenta algumas desvantagens em comparação ao modelo centralizado no que se refere a complexidade, segurança, capacidade de gerenciamento e imprevisibilidade.

110 Na arquitetura em três camadas, no ambiente cliente-servidor, a interface do usuário é armazenada na máquina cliente e o banco de dados é armazenado no servidor.

111 Um sistema distribuído pode ser representado pela arquitetura cliente-servidor, que forma a base para arquiteturas multicamadas, sendo as arquiteturas CORBA, JSON e REST algumas das alternativas para se implantar um sistema multicamadas.

Julgue os itens subsecutivos, a respeito de arquiteturas e servidores *web*.

112 Jackrabbit é um repositório de conteúdo e possui uma implementação em total conformidade para tecnologia Java.

113 No Apache httpd, a funcionalidade mais básica está incluída no núcleo do servidor; os recursos estendidos por meio de módulos devem ser compilados; módulos são estáticos; e o httpd deve ser recompilado para adicionar ou remover módulos.

114 O servidor Apache http permite o gerenciamento descentralizado de configuração por meio de arquivos especiais chamados de `.htaccess`, não sendo permitido o uso de outro nome de arquivo.

115 O servidor Apache http tenta manter um *pool* de *threads* de servidor sobressalentes ou ociosos, que estão prontos para atender às novas solicitações, de modo que os clientes não precisam esperar que novos *threads* ou processos sejam criados antes que suas solicitações possam ser atendidas, podendo essa configuração ser definida por meio da diretiva `StartServers`.

116 Uma implementação amplamente usada como servidor de aplicação Java é o Tomcat, cujo componente principal, chamado de Catalina, possui um conjunto de arquivos de configuração: o arquivo `web.xml` configura opções e valores que serão aplicados a todos os aplicativos e também contém as políticas de segurança do Tomcat para a classe Catalina.

117 Jetty é um servidor http de código aberto, escrito em Java, e um contêiner Java Servlet, facilmente integrado em dispositivos, ferramentas, estruturas, servidores de aplicativos e *clusters* e é caracterizado pelo tamanho, pela velocidade e pela escalabilidade.

Acerca de arquitetura de *software* e modelos, julgue os itens que se seguem.

118 MVC (*Model-View-Controller*) é um padrão de *design* de arquitetura que divide um aplicativo interativo em três componentes: modelo, visualização e controlador.

119 JCAPS é considerada uma plataformas de integração de aplicativos empresariais amplamente utilizada, sendo uma solução unificada e abrangente que se integra, otimiza e se conecta de forma fácil e rápida.

120 A função do *middleware* é ajudar a simplificar o desenvolvimento de aplicativos, oferecendo abstrações de programação comuns, mas não de maneira heterogênea; no entanto, não fornece sistemas operacionais e *hardware* essenciais, sendo necessária uma programação de baixo nível.

Espaço livre