**LISTA GESTÃO DE CONFIGURAÇÃO E MUDANÇAS**

**1) Na gestão de mudanças, quais são as características dos tipos de mudança abaixo?**

Mudança padrão é uma mudança que é pré-autorizada pelo Gerenciamento de Mudanças e que se tornou rotineira, já tendo um script de procedimento para execução. Por este motivo, geralmente o fluxo para execução desta alteração é mais ágil.

Mudança normal é uma mudança para a qual não existe um script já pronto, e precisa passar pelo fluxo mais extenso para ser autorizada e planejada antes de sua execução.

Mudança emergencial é aquela que precisa ser implementada rapidamente para resolver falhas (incidentes). Neste caso, nem sempre será possível realizar todos os testes. Este tipo de mudança é tratado pelo Comitê Consultivo de Mudanças Emergenciais.

**2) Quem são os participantes de um comitê consultivo de mudanças?**

Fornecedores, gerente de problemas e gerente de nível de serviço.

**3) Dê 3 exemplos de regras usuais para que os desenvolvedores realizem mudanças nas funcionalidades de um software.**

Não realizar commit na branch master.

Comentar os commits.

Definir um responsável por realizar a resolução de conflitos.

Antes da realizar um merge é necessário que um responsável ou uma equipe aprove as modificações.

**4) Cite 4 exemplos de ferramentas de controle de versão de código-fonte.**

Git, Subversion, Mercurial e CVS.

**5) O que é integração contínua?**

Integração contínua é uma prática de desenvolvimento de software onde membros de um time integram seu trabalho frequentemente. Cada integração é verificada por um *build* automatizado para detectar erros o mais cedo possível.

**6) Quais são os benefícios da integração contínua?**

Os benefícios da integração contínua é que esta abordagem reduz problemas de integração e permite que o software seja desenvolvido de forma rápida e coesa.

**7) Cite 2 exemplos de ferramentas de software que podem dar apoio à gestão de mudanças.**

Jira e Trac.

**8) Cite 2 exemplos de ferramentas de software para a realização de integração contínua.**

Travis CI e Bamboo.

**9) No que consiste fazer o gerenciamento de *releases*?**

Consiste no empacotamento de um sistema para promovê-lo de desenvolvimento para QA (Quality Assurance) e, posteriormente, para produção.

**10) Cite 3 princípios do gerenciamento de *releases*.**

Releases devem ser identificados por um identificador (ID) de versão imutável.

Releases devem ser empacotados com todas as suas dependências.

O empacotamento de releases deve ser automatizado e desenhado para evitar erros humanos.

**11) Explique como é feita a identificação de um *release* utilizando o versionamento semântico, e qual é o significado de cada parte desta identificação.**

A identificação de um release utilizando o versionamento semântico é feita por uma sequência de 3 números: MAJOR.MINOR.PATCH.

O identificador MAJOR deve ser alterado somente quando as alterações tornam a API incompatível com versões anteriores.

O identificador MINOR deve ser alterado quando as alterações são realizadas para adicionar funcionalidade, mantendo a compatibilidade com versões anteriores.

O identificador PATCH deve ser alterado quando as alterações são realizadas para corrigir bugs, mantendo a compatibilidade com versões anteriores.

**12) Dê um exemplo de identificação de um *release* utilizando o versionamento semântico.**

1.2.14

**13) Cite um benefício obtido com a utilização de uma ferramenta como o Codenvy.**

O Codenvy é uma ferramenta de desenvolvimento de código que possui como um dos principais benefícios a possibilidade de poder ser acessada remotamente de qualquer local, dispensando também a necessidade da instalação de uma IDE em uma máquina local.

**14) Cite um benefício obtido com a utilização de uma ferramenta como o Heroku.**

O Heroku é uma plataforma de serviço em nuvem que possui como um dos principais benefícios a possibilidade de hospedar uma aplicação em ambiente web de maneira que a mesma fique disponível para o acesso de qualquer pessoa.