Desafio Técnico

O desafio consiste na implementação para a solução de três problemas e, para cada um deles, será fornecido um código-fonte base contendo métodos pré-definidos que deverão ser editados. O código também acompanha um conjunto de testes unitários que deverão ser executados para garantir os requisitos mínimos de uma implementação correta.

Os problemas a serem solucionados são:

1. A classe “Luxfacta” recebe um parâmetro no seu construtor que é sempre um número inteiro. Com base nesse número “n”, escreva o corpo do método “say” de forma que:

- Se o número “n” for divisível por 3, o retorno deve ser “Lux”;
- Se o número “n” for divisível por 5, o retorno deve ser “Facta”;
- Se o número “n” for divisível por 3 e por 5, o retorno deve ser “LuxFacta”;
- Qualquer outra condição o retorno deve ser o próprio número “n”;

2. Uma palíndroma é uma palavra que se lê da mesma forma de trás para frente. Escreva o corpo do método isPalindrome que verifica se uma palavra é palíndroma, sem o uso de funções de reversão providas pela linguagem (ex: strrev para PHP, StringBuilder().reverse() para Java, Array.Reverse() para C#, etc.). A validação deve desconsiderar diferença entre maiúsculas e minúsculas (case insensitive).
Por exemplo, isPalindrome(“Deleveled”) deve retornar true.

3. A classe “Path” representa um sistema de arquivos abstrato. Escreva o corpo do método “cd” (change directory).

- O diretório raiz é “/”;
- O separador é “/”;
- O diretório pai é acessível através de “..”;
- O nome de um diretório contém apenas letras (A-Za-z);
- Quando um path for inválido, o método deverá lançar a exceção do tipo InvalidPathException;

Por exemplo, se o construtor da classe for chamado com o parâmetro “/a/b/c/d”, teremos (considerar os comandos chamados em sequência):

- “e/f”: deverá retornar “/a/b/c/d/e/f”;
- “../x”: deverá retornar “a/b/c/d/e/x”;
- “/d”: deverá retornar “/d”;
- “/?”: deverá lançar InvalidPathException;

O candidato poderá optar por realizar os desafios em PHP, Java, C# ou Javascript. Esperamos que a entrega da solução seja realizada em até três horas após o recebimento do desafio.

Entrega

Seu código fonte deverá ser gerenciado em um repositório git local. Você deverá entregar um .tar.gz ou .zip do seu projeto contendo o diretório .git/ e enviar como resposta ao e-mail que usamos para enviar o desafio.

NÃO COMPARTILHE ESSE DESAFIO OU A SUA SOLUÇÃO

Avaliação

Para a avaliação será considerada a simplicidade da solução, sua efetividade e conformidade com os requisitos. Atente aos seguintes pontos quando estiver desenvolvendo seu projeto:

- Os testes disponíveis devem servir de "norte" para que o próprio candidato tenha uma visão prévia se a solução adotada está correta e serão levados em conta na avaliação, porém não serão o único critério. Falha em todos os testes disponíveis, implicará na eliminação do candidato, porém o desafio será avaliado caso algumas falhas existam;
- Os testes estão acessíveis e podem ser analisados para auxiliar no desenvolvimento da solução;
- Se atente ao prazo esperado de entrega (3 horas), o tempo de entrega será considerado na avaliação, mas não será eliminatório. O candidato deverá ponderar entre atrasar um pouco a entrega ou realizar uma entrega com erros (caso sua solução apresente algum);
- Code Style: não preferimos jeito A ou jeito B, contanto que haja consistência ao longo do código.

Instruções para Execução dos Testes

Seguem abaixo as instruções de execução dos testes unitários:

- **PHP:**
  - Instalar o composer para gerenciamento de dependências (caso já não possua instalado);
  - Na pasta do projeto (diretório onde os arquivos composer.json e composer.lock estão localizados), executar o comando: composer install
  - Executar o phpunit (baixado pelo composer) passando a pasta de testes do projeto como parâmetro. Exemplo de execução a partir da pasta do projeto:
    - UNIX: ./vendor/bin/phpunit ./tests
    - WINDOWS: .\vendor\bin\phpunit .\tests
  - Os testes serão executados e o resultado exibido ao final:

    ```
    FAILURES!
    Tests: 22, Assertions: 22, Failures: 21.
    ```

  - Acima da mensagem status geral, será exibido o erro ocorrido para cada um dos testes executados.
- **Java:**
  - Caso a IDE que esteja utilizando não forneça uma opção para execução de testes:
    - Instalar o ant para a build e execução do jUnit;
    - Na linha de comando, abrir a pasta do projeto e executar: ant junit;
    - Após a execução será exibido um resumo e serão gerados os relatórios de testes executados na pasta "report".
- **C#**
  - Após a compilação do projeto, o usuário deverá abrir o Console do Gerenciador de Pacotes (no Visual Studio, clicando no menu superior **-> Ferramentas -> Gerenciador de Pacotes do NuGet -> Console do Gerenciador de Pacotes**).
  - Ao iniciar o gerenciador de pacotes, executar o comando **dotnet test**
  - Este comando irá iniciar a execução dos testes unitários, ao final dos testes, um relatório detalhado e um sumário serão apresentados, com os testes que foram bem sucedidos e os que não completaram com sucesso.