****** Documento Confidencial ******

Introdução

Olá,

Gostaríamos de agradecer por participar do nosso processo seletivo para a vaga de **Desenvolvedor(a) Full Stack** e parabenizar por ter avançado para a etapa de avaliação técnica.

O objetivo desta etapa é avaliar a sua capacidade de raciocínio lógico, arquitetura de *software*, organização do código, metodologia de desenvolvimento e velocidade. Por isso, gostaríamos de simular uma situação real, onde você tem todos os recursos que quiser para resolver o problema dentro do tempo que precisar.

Regras

Você tem liberdade quase total para resolver o desafio, assim como numa situação real do mundo profissional. Claro, algumas pequenas regras são necessárias e contamos com a sua cooperação e código de honra para respeitá-las:

Pode

- Usar sua IDE preferida
- Google
- Stackoverflow

Não pode

- Copiar e colar soluções inteiras (plágio não é legal! 😔)
- Consultar outro desenvolvedor (pair programming não vale! 😕)

Fica clara a idéia, né? Vale tudo, desde que seja você mesmo(a) que esteja fazendo.

Logística

Assim que você receber este documento, o relógio começa a andar. Leve o tempo que quiser para entregar na melhor qualidade possível. O ponto ótimo entre qualidade de código e velocidade ficará à seu critério. O relógio para no momento em que a resposta chega na nossa caixa de e-mail.

A sua resposta deverá ter um arquivo zip ou link para download contendo:

- Código

- Arquivos listando dependências (requirements.txt, Pipfile, package.json, etc)
- Documentação (opcional)
- Testes (opcional)
- Deploy (opcional recomendamos os serviços da AWS)

Critérios de Avaliação

A solução proposta será avaliada em relação à:

- Corretude
- Completude
- Performance
- Arquitetura
- Organização do código
- Tempo total para envio da solução final

Desafio

O desafio consiste em construir uma aplicação web composta de frontend e backend para resolver o seguinte problema:

> Duas pessoas, Marcelo e Carla, trabalham em uma fazenda de maçãs na qual existem N árvores. As árvores estão alinhadas em uma fileira e são numeradas de 1 a N. Marcelo planeja coletar maçãs de **K** árvores consecutivas e Carla de **L** árvores consecutivas. Eles querem escolher segmentos disjuntos, ou seja, que não se sobrepõem para não interferirem na coleta do outro. Qual o maior número de maçãs que os dois juntos podem coletar?
>
> Escreva uma função:
>
> `def get_max_apples(A, K, L)`
>
> que, dado um vetor A consistindo de N inteiros denotando o número de maçãs em cada árvore, e inteiros K e L, respectivamente, o número de árvores que Marcelo e Carla escolhem coletar, retorne o número máximo de maçãs coletadas ou `-1` se não existem intervalos que permitam a coleita.
>
> Por exemplo, dado um vetor `A = [3, 4, 1, 7, 8, 5]`, `K = 2`, `L = 3`, sua função deve retornar 27, já que Carla escolherá as árvores 4, 5 e 6 e coletar 7 + 8 + 5 = 20 maçãs. Marcelo, por sua vez, escolherá as árvores 1 e 2, coletando assim 3 + 4 = 7 maçãs.
>
> Dado o vetor `A = [1, 3, 5]`, `K = 2`, `L = 2` a função deverá retornar -1, pois não existe como Marcelo e Carla escolherem dois intervalos disjuntos de tamanho 2.

Frontend

O Frontend deve ser escrito em Javascript, preferencialmente utilizando o framework Vue.JS. Ele deve ser composto de uma tela inicial que permite a entrada dos dados do vetor de árvores `A` e do tamanho dos intervalos de Marcelo e Carla, `K` e `L`.

O Frontend deve enviar os dados para o Backend e mostrar a resposta em uma segunda tela, com alguma representação gráfica do resultado mostrando as árvores, com os respectivos números de maçãs em cada, e os dois intervalos selecionados. Não avaliaremos o design da solução, portanto, a representação gráfica pode ser como achar mais fácil de fazer, desde que seja de fácil entendimento. Por exemplo, uma simples tabela pode dar conta do recado.

Backend

O Backend deve ser escrito em Python e recomendamos:

- Serverless, por ser um framework simples para micros serviços (caso decida por uma abordagem em micro serviços)
- Flask, por ser um framework simples e praticamente sem configuração necessária

A comunicação com o Frontend pode ser feita utilizando endpoints REST ou GraphQL. É importante que o Backend receba os parâmetros do Frontend e retorne as informações necessárias para a visualização correta da solução. Também é importante que a função contendo o algoritmo esteja desacoplada dos endpoints para permitir testes de unidade.

Conclusão

Esperamos que este desafio seja encarado com interesse e seriedade mas que também seja intelectualmente estimulante. Se você achar que está perdendo muito tempo resolvendo-o, provavelmente está! Nossa intenção não é criar um problema, esse teste foi feito para que divirta-se.

Qualquer dúvida pode ser tirada através do e-mail **lucas.martins@balko.com.br**. Deixe comentários no seu código explicando qualquer suposição feita e levaremos em consideração.

Boa sorte e divirta-se!

Att,

Equipe Balko