PROGRAMADOR DE ELITE

COMO SAIR DO ABSOLUTO ZERO E TRABALHAR NAS MELHORES EMPRESAS DO MUNDO

MATEUS DANTAS, PEDRO NASCIMENTO, PHYLLIPE MEDEIROS

Primeira edição, 2019

www.programadordeelite.com

Programador de Elite

Como sair do absoluto zero e trabalhar nas melhores empresas do mundo

Introdução

Milhares de brasileiros constantemente buscam um emprego dos sonhos - e esse nem sempre é aquele de melhor remuneração. Muitas vezes, procuram algo desafiador, relevante para a vida de outras pessoas e que proporcione oportunidades de crescimento e um bom ambiente de trabalho. Afinal, ninguém quer chegar no serviço já com a intenção de voltar para casa.

Então, por que não unir tudo isso? Já pensou em trabalhar ganhando mais de 500 mil reais em projetos que impactam a nossa sociedade e ainda assim ter um horário de trabalho flexível, ou até mesmo, quem sabe, uma pausa para jogar vídeo-game? E que tal benefícios como trabalhar de casa, remotamente?

Você sonha trabalhar em grandes empresas de tecnologia como Facebook, Google, Microsoft e Apple? Já imaginou ser entrevistado por uma delas, mas não sabe por onde começar ou como se preparar?

Pelo fato da tecnologia estar presente em praticamente tudo que envolve o nosso cotidiano, a nossa dependência só continua a crescer. Por exemplo, quando acreditamos que nada mais pode ser criado, empresas nos surpreendem e estudam como desenvolver carros que dirigem sozinhos. Com isso, o salário e a demanda por

profissionais da área já atingiram um patamar histórico e a previsão é que só continuem crescendo.

Este livro revela que o mercado de tecnologia não se resume apenas aos gigantes da área, como Apple e Microsoft, pois existem diversos outros tipos de empresas, com diferentes nichos de atuação para os desenvolvedores.

Saber da existência dessas firmas não é o suficiente! Muitas pessoas desconhecem o caminho necessário para conseguir uma entrevista em empresas do exterior. Além disso, há uma grande dificuldade na hora de se prepararem para essas entrevistas.

Os autores possuem mais de 10 anos de experiência com programação e trabalham em grandes firmas como Google, Snapchat e Facebook. Além disso, compartilham a experiência de centenas de pessoas entrevistadas no currículo e todos são movidos por uma paixão em comum: ajudar outros programadores (e aspirantes a programadores) a alcançarem seus sonhos.

A motivação de escrever este livro veio ao ajudar amigos pessoais que desejavam imensamente trabalhar em grandes empresas de tecnologia, mas achavam essa possibilidade muito difícil e não sabiam por onde começar.

Com este exemplar, o leitor entenderá que trabalhar em boas empresas de tecnologia não é um sonho distante. Você também compreenderá como funciona o processo de seleção e saberá como se preparar para as entrevistas. E, se esse for o seu primeiro contato com a programação, vamos ajudá-lo a sair do absoluto zero e alcançar os seus objetivos.

É importante lembrar que o sucesso depende muito da sua dedicação e nunca esquecer que, enquanto você prorroga sua preparação, apesar do mercado crescer cada vez mais, milhares de outras pessoas estão se preparando para alcançarem seus sonhos. Por isso, comece agora! Leia este livro e se prepare para as entrevistas, pois isso pode mudar a sua vida.

O futuro

Você consegue imaginar a sua vida sem a existência da tecnologia?

Desde a invenção do primeiro computador, em meados de 1938, na Alemanha, até a criação dos mais sofisticados smartphones nos dias de hoje, o ser humano pauta praticamente todas as esferas do seu cotidiano em volta da tecnologia. Ela está presente no ar, na água, na comida, nos negócios, na comunicação, no transporte e em praticamente tudo que envolve o nosso dia a dia.

Antigamente, com a ausência da tecnologia, enviar uma carta para se comunicar com um familiar que estava morando longe levaria dias para ser entregue. Atualmente, você consegue imaginar sua vida sem mensagens de texto?

Hoje, mais de 2 bilhões de pessoas usufruem de um conhecimento instantâneo, utilizando tecnologias que cabem no bolso da calça! Todos os smartphones possuem um poder de processamento maior do que os computadores que guiaram os astronautas à lua.

A tecnologia mudou as nossas vidas e vai continuar mudando; cientistas estudam como levar vida para outro planeta e empresas de aviação buscam construir um avião hipersônico que viajará cinco vezes mais rápido que a velocidade do som, portanto, não existem limites para o que ainda pode ser alcançado!

E que tal fazer parte dos que podem revolucionar esse mercado? A computação não é só mais o futuro, mas sim o presente. Salário e demanda por desenvolvedores ou Engenheiros de *Software* atingiram um patamar histórico e a previsão é de que isso só continue crescendo.

Marc Andreessen, um dos investidores de maior sucesso do Vale do Silício com empresas como Facebook, Instagram e Twitter em seu portfólio, escreveu um polêmico artigo em 2011, no *The Wall Street Journal*, quando afirmou que o *software* estava dominando o mundo e, eventualmente, todas as indústrias iriam se render à tecnologia. Apesar de ter repercutido bastante nos meios de comunicação daquele ano, praticamente oito anos depois podemos perceber que essa afirmação não somente continua verdadeira, como vem ganhando força com o passar dos anos.

Mais e mais indústrias seculares estão sendo revolucionadas e viradas de ponta-cabeça pela chegada do *software*. O Uber e o

AirBnb são dois dos mais diversos exemplos de firmas que causaram uma ruptura histórica nas indústrias de transporte privado e locação de imóveis, fenômeno nunca antes visto. A explicação é simples: **o *software* é democrático e melhora a vida de todo mundo.**

Assim como o *software*, o mercado de computação e TI em geral também é bastante democrático. Para se preparar para a entrevista de qualquer emprego da área, basta ter acesso à internet, um computador e saber como estudar. É uma receita simples e que, se bem aplicada, é altamente eficiente. Se você alguma vez já ouviu que é preciso passar por um curso de computação ou ser um gênio da área de exatas para se dar bem nessa área, pode ter certeza que isso não é verdade. Empresas de *software* estão cheias de engenheiros autodidatas ou alunos com as mais inesperadas formações, tendo muitas vezes não terminado uma graduação ou sequer passado por uma universidade. E não se engane: esses engenheiros de formação variada não ficam para trás dos alunos que passaram por bons cursos de computação e afins. O intuito deste livro é mostrar para você, que se sente inspirado pela forma com que a tecnologia vem melhorando a vida das pessoas, a se preparar e conquistar os seus sonhos nesse mercado que está em constante ascensão e não tem prazo de validade.

Nos próximos capítulos, vamos apresentar dicas valiosas de como você pode se preparar e mudar a sua mentalidade para ser bem sucedido em qualquer tipo de entrevista feita por uma empresa de *software*.

O conteúdo deste livro é o resultado de anos de experiência de pessoas que já passaram pelas corporações mais renomadas do mundo e decidiram utilizar tudo que aprenderam como entrevistados e entrevistadores para ajudar você a conquistar o emprego dos seus sonhos

Motivação

Com mais de 10 anos de experiência profissional combinada, trabalhando em multinacionais, *startups* e consultorias remotas, os autores do livro já ajudaram, direta e indiretamente, na preparação de dezenas de pessoas. Embora a maioria tenha conseguido alcançar seus objetivos e hoje sejam profissionais de sucesso, todos sabem que o começo foi difícil e repleto de dúvidas.

Alguns dos problemas e dúvidas mais comuns que foram compartilhados com a gente talvez sejam familiares para você. Vejamos:

Por onde eu devo começar?

Não sou formado em computação, nunca programei na vida e não sou excepcional em exatas. Tenho alguma chance?

Eu quero trabalhar em uma* big tech*, mas pouquíssimas pessoas se classificam. O que fazer para aumentar as minhas chances?

Como me preparar para a entrevista? Que tipos de perguntas são feitas? Qual linguagem de programação devo usar?

Consegui uma oferta de emprego, e agora? Como negociar para obter uma proposta melhor?

Essas são as dúvidas mais comuns de pessoas que se preparam para uma entrevista de emprego no mercado de computação. Se você tem dúvidas semelhantes ou iguais a essas, saiba que não está sozinho. O caminho até o sucesso, muitas vezes, não é completamente claro e definido, mas se você continuar tentando, você vai chegar lá. O mais importante de tudo é dar o primeiro passo e continuar persistindo, independente das adversidades que possam acontecer no meio do caminho.

Determinação e persistência são uma daquelas palavras de impacto que todo mundo sabe falar, mas são poucos os que de fato conseguem incorporá-las. Afinal de contas, de uma forma ou de outra, todos estão em busca do seu lugar ao sol, mas nem todo mundo está disposto a pagar o preço, buscar conteúdo e fazer o que for necessário para conquistar os seus objetivos. Se você chegou até

este livro, significa que já deu um grande passo. Ir atrás do conhecimento necessário é fundamental para a realização dos seus sonhos.

Quando decidimos escrever este livro, nós sabíamos que não seria fácil. Além do tempo reduzido, devido ao trabalho integral, nenhum dos autores possui experiência como escritor. Apesar de ser um empecilho na nossa jornada, esses nunca foram fatores limitantes, pois acreditamos sempre que podemos aprender durante o processo. Por acreditar e já ter passado por todo esse percurso que detalhamos aqui neste livro, nós sabíamos que era tudo questão de aplicar a mesma metodologia de sucesso que utilizamos para chegar até onde chegamos; ou seja, buscar o conhecimento necessário, aplicá-lo de forma consistente e contínua, e ainda assim, aprender com os nossos erros.

No nosso caso em específico, fomos atrás de livros, artigos e o que mais você possa imaginar a respeito de como escrever um livro. Colocar em prática foi a tarefa mais difícil: escrevemos e reescrevemos diversos trechos aqui encontrados, até chegar nesta versão final que, hoje, você está lendo.

Pode parecer simples, mas esse processo de buscar conhecimento, colocá-lo em prática e aperfeiçoá-lo é extremamente

eficaz e pode ser utilizado em qualquer aspecto da sua vida. Ao longo do livro, você vai perceber que a preparação para entrevistas de programação em sua simplicidade se resume a isso.

Iremos abordar e mostrar em detalhes quais os diferentes tipos de oportunidades no mercado e um passo a passo de como se preparar especificamente para cada uma delas.

Ao final, você vai ter todo o *framework* necessário para se dar bem em qualquer empresa no mercado de computação, seja ela no Vale do Silício, na cidade onde mora ou até remotamente.

Sobre os autores

Pedro Nascimento

Pedro Nascimento é carioca e atualmente mora em Londres, na Inglaterra. Começou a programar aos 13 anos de idade desenvolvendo websites e não parou mais. Mestre em engenharia elétrica pela Coppe-UFRJ, com ênfase em inteligência artificial e aprendizado de máquina. Formou-se em engenharia eletrônica e matemática aplicada pela Universidade Federal do Rio Janeiro (UFRJ), onde desenvolveu suas habilidades em diversas áreas além da computação, tais como processamento de sinais, eletrônica, robótica e matemática pura. Desde jovem demonstrava talento em olimpíadas científicas, ganhando medalhas em diversas, principalmente nas olimpíadas de informática e matemática. Sua paixão pela computação se intensificou ainda durante ensino médio, quando se formou como técnico de informática pelo CEFET-RJ. Possui anos de experiência em programação competitiva e profissional, tendo trabalhado em diversos projetos durante todo o seu tempo na universidade. Atualmente é engenheiro de *software* na Google em Londres, onde desenvolve sistemas e lidera projetos que servem a milhões de pessoas.

Mateus Dantas

Programando há mais de 12 anos, Mateus Dantas, que sempre teve uma grande paixão por computadores e *software* em geral, viu os primeiros frutos do seu trabalho logo na adolescência, ao conquistar diversas premiações nacionais e internacionais em competições de programação. Em 2013, após conquistar a nota mais alta na seletiva para a Olimpíada Internacional de Informática, fez parte da comitiva de 4 alunos que representaram o Brasil na competição. Em 2014, integrou, juntamente com Phyllipe Medeiros, o time que foi medalha de Bronze na Maratona Brasileira de Programação e, consequentemente, classificado para representar o Brasil e a Universidade Federal de Campina Grande na fase mundial. Após o sucesso obtido em várias competições, decidiu se preparar e fazer entrevista para as mais diversas empresas de *Software*. Realizou dois estágios no Facebook, um na Google, um na TFG (uma das maiores empresas de jogos *mobile* do mundo) e um estágio remoto para uma empresa do Vale do Silício. Atualmente, trabalha como Engenheiro de *Software* no Snapchat, em Londres, na Inglaterra. Depois de ajudar diversas pessoas, de forma direta e indireta, a conquistarem o emprego dos seus sonhos, Mateus decidiu

juntamente com Pedro e Phyllipe, reunir em um livro toda a sua experiência e as suas melhores dicas para ajudar o maior número de pessoas possíveis.

Phyllipe Medeiros

Phyllipe Medeiros é paraibano, nascido em João Pessoa, formado em Ciência da Computação pela Universidade Federal de Campina Grande (UFCG) e começou a programar *bots* e *scripts* para o mIRC aos dez anos de idade. Participou de diversas competições de programação, tendo logrado êxito e recebido várias medalhas de ouro. Em 2010, dentre milhares de candidatos conquistou, a primeira medalha de ouro a nível nacional da sua universidade. Representou o Brasil por duas vezes em mundiais de programação e em um deles ganhou o prêmio de melhor time da América Latina. Sempre motivado em ajudar, criou um grupo de estudos na UFCG, almejando uma melhor performance dos alunos da instituição perante competições de programação. Aos 18 anos de idade, assinou seu primeiro contrato com o Facebook e foi eleito um dos melhores estagiários da empresa, sendo convidado a jantar na casa de Mark Zuckerberg. Na metade da graduação, aos 19 anos, já possuía um contrato efetivo com a rede social para trabalhar em Londres e,

assim que se formou, com a nota mais alta dentre todos os formandos, mudou-se para o Reino Unido. Atualmente, lidera outros engenheiros para escalar a infraestrutura da empresa e tem como paixão ajudar brasileiros a trabalharem em grandes empresas.

O mercado de trabalho

O mercado de tecnologia é amplo tanto na quantidade de vagas disponíveis quanto na variedade em termos de oportunidades. Aproximadamente cerca de 100 milhões de *startups* são fundadas anualmente no mundo. Isso é equivalente a quase 3 *startups* por segundo. Ainda que a grande maioria não consiga obter o resultado esperado e feche as portas em seu estágio inicial, esse número demonstra que o mercado de computação se expande muito mais rápido do que se pode imaginar, resultando em uma considerável quantidade e variedade de oportunidades.

Quem tem experiência na área sabe que existem diversas formas de um engenheiro de *software* atuar no mercado, seja como profissional autônomo ou remoto, dentro de grandes empresas e até mesmo em *startups*. Embora sejam relativamente diferentes uns dos outros, todos esses papéis podem ser igualmente bem sucedidos, desde que executados da forma correta.

Sob certas circunstâncias, algumas *startups* e empresas grandes permitem que seus engenheiros trabalhem de forma remota, facilitando a entrada de pessoas que, porventura, não tenham total disponibilidade presencial. *Startups*, grandes empresas e trabalhos

remotos serão o nosso foco daqui em diante, pois são essas oportunidades que exigem um grau maior de preparo para possíveis entrevistas.

A escolha de qual caminho seguir é única e exclusivamente sua. No entanto, cada um dos três caminhos aqui abordados e apresentados exige uma estratégia de preparação diferente.

Uma boa preparação é fundamental para uma entrevista exitosa. Entretanto, existem inúmeros casos de pessoas que acabam não se preparando corretamente, pois acreditam que saber programar é o suficiente para passar em qualquer entrevista.

Ainda que o cargo para o qual você vai ser contratado ou o tipo de trabalho que será exercido seja geralmente o mesmo em todas as empresas, cada uma delas define de maneira diferente o que esperar de cada candidato.

Algumas preferem entrevistas mais práticas, voltadas para o desenvolvimento de mini projetos; outras optam por entrevistas mais voltadas para algoritmos e estrutura de dados. Porém, caso você não esteja familiarizado com nenhum desses termos, não se preocupe, pois, ao longo deste livro, iremos detalhar o que é preciso saber sobre cada um deles e como se preparar para qualquer entrevista que você tenha a oportunidade de fazer.

As empresas de tecnologia

Grandes Empresas

Quando se fala de tecnologia e computação, é impossível não mencionar empresas como Google, Apple, Facebook, Microsoft e Amazon. Trabalhar para essas empresas é o sonho de uma grande parte de muitos profissionais, principalmente engenheiros de *software*. Juntas, essas firmas possuem um valor de mercado de mais de três trilhões de dólares, ou aproximadamente 12 trilhões de reais na cotação atual de mercado. Para se ter uma noção da representatividade desses valores, o Reino Unido possui um produto interno bruto ou PIB de cerca de 2 trilhões e 800 bilhões de dólares. Isso significa dizer que juntas, as referidas empresas são mais valiosas do que tudo que é produzido no Reino Unido.

Esses valores de mercado astronômicos não aconteceram por acaso. Cada uma dessas empresas revolucionou e revoluciona o mercado de maneira significativa, através de inovações tecnológicas constantes. Cada uma impulsiona o avanço da sociedade como um todo. Assim como a Ford revolucionou a indústria da produção de carros no século passado, essas firmas hoje estão revolucionando o avanço da tecnologia.

Para conseguir estar sempre inovando e impulsionando o progresso humano, é necessário um alto investimento de capital em processos, pesquisas e principalmente em pessoas. Sem que haja pessoas altamente capacitadas e preparadas, não é possível construir ou manter um negócio nesse patamar, pois são elas que surgem com novas ideias, que dão vida a produtos, influenciando positivamente a sociedade.

Empresas como as citadas anteriormente não só investem incessantemente na contratação de engenheiros, como os remuneram com salários, ações e bônus bem acima da média de mercado.

Ser muito bem pago não é a única característica que atrai engenheiros: horários flexíveis, ambiente de trabalho inclusivo e a possibilidade de trabalhar em projetos que vão moldar a sociedade com algumas das mentes mais brilhantes do mundo são alguns dos benefícios que fazem dessas empresas os melhores lugares para se trabalhar no mundo. Além disso, a oportunidade de se entender as melhores práticas e processos por trás de empresas que são consideradas extremamente bem sucedidas em larga escala vai te dar acesso a um tipo de conhecimento que provavelmente nenhuma outra empresa pode te oferecer.

E por colocar tanto poder nas mãos dos seus engenheiros, essas empresas geralmente possuem um elevado nível de exigência em relação a suas entrevistas de emprego. Resolver problemas com mais eficiência, de forma mais clara, além de uma boa comunicação são algumas das características que se esperam de engenheiros que costumam ser bem sucedidos nessas entrevistas.

Um dos mitos que infelizmente leva muita gente a desistir no meio do caminho ou sequer tentar é de que a concorrência por essas vagas é muito alta, tornando quase impossível as chances de sucesso. Antes de continuarmos a leitura, é necessário que você entenda que isso não é verdade. De fato, a concorrência é alta, mas o seu sucesso só depende única e exclusivamente de você. Na grande maioria dos casos, o comitê que decide a contratação analisa apenas o desempenho nas entrevistas, ou seja, basta atingir o nível de exigência estabelecido pela empresa que você vai receber uma oferta.

O processo de aprendizagem até chegar em um nível em que você esteja pronto para qualquer entrevista deve ser feito passo a passo. Pular qualquer etapa significa que as suas chances de sucesso serão diminuídas, pois cada etapa ignorada é uma habilidade a menos que você não terá na hora de ser entrevistado.

Esse passo a passo é o foco do nosso livro. Ele inclui dicas de preparação de currículo, o que você deve aprender, como e onde estudar e até mesmo dicas de como se portar durante a entrevista. Esse processo inteiro pode ser seguido por qualquer pessoa, de qualquer formação, seja ela de exatas, humanas ou qualquer outra área que se possa imaginar.

Nós acreditamos plenamente que caso você siga disciplinadamente cada uma das etapas que vamos detalhar nos próximos capítulos, será apenas uma questão de tempo até que você obtenha sucesso, qualquer que seja a empresa escolhida.

Startup

Quando se fala em *startup*, muitos imediatamente associam essa ideia a uma empresa comandada por um grupo de jovens que trabalham em algum aplicativo ou site descolado e prometem desestabilizar alguma próxima grande indústria. Mas realisticamente falando, o que de fato é uma *startup*?

Na verdade, não há definição correta sobre o que é uma startup. Isso mesmo, por incrível que pareça, não existe um consenso sobre o conceito de *startup*. Alguns costumam definir como

um estado de espírito. Outros costumam fazer essa definição de acordo com a receita, tempo de mercado ou até mesmo com a quantidade de funcionários presentes na companhias.

A nosso ver, *startup* é um negócio que busca desenvolver produtos ou serviços inovadores, com um alto potencial de crescimento, de forma repetível e escalável.

No Brasil, possuímos diversas empresas que se encaixam nesse perfil. Alguns exemplos dessas empresas são: VTEX, TFG, QuintoAndar, 99 Taxi, Nubank e Loggi.

Ainda que algumas sejam melhor estabelecidas que outras, trabalhar em uma *startup* pode proporcionar a qualquer profissional a oportunidade de perceber o impacto das trajetórias de crescimento de cada uma delas. Esse é um dos prazeres que poucas empresas podem te oferecer. Afinal de contas, por mais que empresas grandes como Microsoft ofereçam uma maior segurança, o seu impacto na história delas vai ser completamente diluído entre as dezenas de milhares de pessoas que já trabalham lá.

Um outro aspecto interessante é a forma de remuneração. Algumas, especialmente as que se encontram nos estágios mais iniciais, não possuem tanto capital e, portanto, não apresentam condições de oferecer salários e bônus exorbitantes, como as

empresas mais consolidadas costumam pagar. Para resolver esse impasse, é comum as *startups* oferecerem uma parte da remuneração em algum plano de ações da empresa. O mais comum hoje em dia é o de *stock options*.

Stock options é um plano de opção de compra de ações no qual a empresa fornece ao funcionário a opção de comprar suas ações a um certo valor pré-determinado e geralmente depois de certo período de tempo.

A verdade é que enquanto a empresa não for vendida ou não abrir o capital, as ações que você possui dela não tem valor nenhum no mercado - salvo algumas exceções de *startups* que te permitem vender as ações que você adquiriu de volta para ela mesma. Em resumo, enquanto a *startup* em que você trabalha não fizer um *exit* (geralmente abrir o capital ou ser comprada por outra empresa), dificilmente você vai ver a cor do dinheiro.

O outro lado dessa moeda é que um *exit* costuma ser bastante generoso - financeiramente falando - para os funcionários que entraram antes dele. Existe uma história bastante interessante sobre isso. Por volta de 2005, um grafiteiro americano chamado David Choe foi contratado pelo Facebook para decorar os murais do escritório pelo então presidente Sean Parker. Como forma de

pagamento pelo seu trabalho, David foi oferecido receber cerca de 60 mil dólares em dinheiro vivo ou em ações da companhia. Inicialmente, David se mostrou um pouco receoso a aceitar as ações, até porque ele provavelmente sabia que existia o risco de nunca mais ver a cor daquele dinheiro, caso a rede social não fosse uma empresa bem sucedida. Depois de muita conversa com o presidente, David decidiu finalmente aceitar o pagamento em ações. Provavelmente você já deve estar imaginando o quão bem David se deu, pois quando o Facebook abriu o capital em maio de 2012, as ações que ele recebeu no valor de 60 mil doláres passaram a valer 200 milhões de dólares, ou seja, um retorno de quase 3000 vezes sobre o valor inicial. E se David, que fez um simples trabalho de consultoria para a empresa, se deu tão bem, você deve imaginar como estão os engenheiros que trabalharam no Facebook desde aquela época.

Assim como em empresas grandes, dinheiro não é o único motivo pelo qual engenheiros decidem trabalhar para uma *startup*. Por não ter tantos recursos quanto em empresas maiores, geralmente *startups* concentram mais poder e responsabilidade nas mãos dos seus engenheiros, ou seja, o escopo dos projetos em que você vai trabalhar é mais abrangente e o impacto causado por eles é

maior e mais visível se comparado com empresas maiores. Toda essa liberdade geralmente está atrelada à expectativa de que você consiga se adaptar ao ritmo acelerado de crescimento da empresa. *Startups* geralmente precisam se adaptar e estar preparadas às rápidas mudanças do mercado, sob pena de serem esmagadas pela brutal concorrência do mercado como um todo.

No geral, coisas como títulos de emprego, estruturas hierárquicas ou até mesmo requisitos de um projeto mudam com mais frequência do que um filtro de uma máquina de café.

Essa constante mudança pode ser meio frustrante para alguns, especialmente se você está acostumado a corporações onde as políticas seguem uma ordem linear de progresso. Mas para se dar bem em uma *startup*, é preciso que você se acostume com o caos. Todo esse caos, além de servir como uma fonte de motivação diária, também vai fazer com que você aprenda habilidades que nenhum outro tipo de empresa vai poder te ensinar. Por exemplo, aprender a fazer *pitch* e vender o produto em que você trabalha para potenciais clientes é uma das diversas habilidades multidisciplinares além do escopo de desenvolvimento de software que é bastante comum de se adquirir nesse tipo de ambiente.

Além disso, aprender a lidar com a falha como um passo para o sucesso talvez seja o que de mais importante você possa levar consigo para o resto da sua vida. O processo de evolução e aprendizado em uma *startup* muito se parece com o de tentativa e erro. Aprender a falhar, mas falhar rápido e pivotar na direção de algo que pode funcionar é essencial para a sobrevivência em um ambiente tão caótico e competitivo. E esses ensinamentos valiosos que você pode aprender em uma *startup* podem ser aplicados em qualquer outra área da sua vida, seja ela no seu desenvolvimento pessoal ou caso você decida abrir uma empresa própria no futuro.

Devido a esse ambiente acelerado e não usual, *startups* costumam ter um processo de entrevista que se adapta às necessidades de momento da empresa. Por exemplo, se a firma necessita de engenheiros que saibam uma certa tecnologia, então é comum que elas adaptem o seu processo para focar em candidatos que já possuem esse conhecimento específico. Já que toda tecnologia exige um certo tempo de aprendizado e *startups* geralmente não possuem nem tempo nem recursos disponíveis, faz mais sentido para elas contratarem pessoas que já tenham esse conhecimento específico. Então, em muitos casos, é comum que seja exigido que você já saiba uma certa linguagem de programação,

um certo *framework* ou saiba trabalhar em alguma área específica, como desenvolvendo aplicativos, *web* ou *backend*.

Por não ser um processo tão previsível como os demais tipos de empresa, vamos focar neste livro em te ajudar a escolher um conjunto de *startups* ideal para você, como identificar o que é exigido e como se preparar especificamente para cada uma dessas empresas. Ao final deste livro, esperamos que você tenha o conhecimento necessário para ser bem sucedido em qualquer que seja a *startup* escolhida.

Trabalho remoto

Você já deve ter ouvido falar em “trabalhar de casa”, mas o termo “trabalho remoto” se refere a qualquer tipo de trabalho que é executado sem que ninguém precise se locomover para um escritório.

Isso mudou graças à era digital. Hoje é possível terminar projetos e se comunicar com o seu time sem estar na mesma sala ou cidade. A chave para isso tudo é apenas uma boa conexão com a internet.

Trabalho remoto funciona mesmo?

Certamente! E oportunidades na área vêm crescendo cada vez mais. Estudos mostram que:

- O crescimento dessas vagas aumentou exponencialmente nos últimos anos, claramente mostrando que empresas estão obtendo bons resultados.

- 77% dos empregados remotos são mais produtivos.

Diversas corporações oferecem oportunidades de emprego totalmente remota. Um exemplo disso é o quinto escritório da Stripe (empresa americana de pagamentos que vale mais de 100 bilhões de reais), o qual é totalmente remoto.

Quais são as diferenças?

Trabalhar remotamente, assim como tudo na vida, tem suas vantagens e desvantagens. Aqui mostraremos algumas delas.

Pró: Trabalhar de qualquer cidade

Você já viu aquele emprego dos sonhos, mas que infelizmente não era na cidade em que você morava? Trabalhar remotamente te dá a

liberdade de trabalhar em qualquer lugar, e até mesmo viajar o mundo trabalhando!

Pró: Estresse reduzido

Trabalhar remotamente pode reduzir o estresse consideravelmente. A sensação de privacidade pode ser um grande benefício para pessoas que se sentem melhor sozinhas.

Um dos maiores fatores que contribuem para o estresse envolvem problemas com a locomoção para o trabalho. Para empregados que vivem numa cidade grande, o trânsito ou até mesmo problemas com transporte público podem afetar seu dia a dia na empresa.

Contra: Risco de se sentir isolado

Uma das desvantagem é que pode ser difícil para empregados remotos desenvolverem uma afinidade com outros colegas. Não existem oportunidades para uma conversa rápida enquanto esperam pelo café ou de até mesmo a possibilidade de almoçar juntos.

Os diferentes papéis em uma empresa

A arte de desenvolver *software* é feita de forma diferente e varia de empresa para empresa. Vale dizer que o processo que é seguido em uma, sendo bem sucedido em determinado ambiente e situação, pode falhar miseravelmente em outras circunstâncias. São essas diferenças que, geralmente, tornam tão difícil reduzir o desenvolvimento do *software* a uma única função ou responsabilidade.

Dito isso, é bastante comum que isso provoque uma certa generalização. Qualquer pessoa que trabalha nessa área será reconhecida como um desenvolvedor ou programador, tão somente. Teoricamente falando, essa definição não está equivocada, pois, afinal de contas, quem trabalha produzindo *software* precisa, em dado tempo, desenvolver ou programar algum tipo de código.

Todavia, assim como na medicina - onde cada médico pode se especializar em determinado ramo, como cardiologia ou pediatria, por exemplo, desenvolvedores de *softwares* também podem ser especialistas. Das várias especialidades, podemos citar os Engenheiros de *Software*, de Produto, *Site Reliability Engineer* (SREs) e até mesmo trabalhar em *backend* e *frontend*

Ainda que existam essas diferenças, algumas coisas não mudam com o passar do tempo. Vale dizer que sempre vai haver a necessidade de se entender um problema, traduzi-lo em uma solução, testá-la, implantá-la e depois mantê-la. Alguns desses processos podem variar, ou ter mais ênfase, dependendo do papel que você esteja assumindo ou da própria empresa em que você trabalhe. Um engenheiro de *software* de testes, citando caso análogo, pode se especializar e voltar atenção mais no processo de testar a solução; já um engenheiro de *software frontend* especializa-se mais em resolver problemas voltados para o cliente.

Certas pessoas podem assumir todos, alguns ou apenas uma dessas funções. Apesar disso, existe uma necessidade real de uso em cada uma dessas responsabilidades. Afinal de contas, todos têm o seu propósito na empreitada assumida.

Inicialmente, pode parecer um pouco assustador se é a primeira vez que você mantém contato com este mundo. Mas não se preocupe! Explicaremos adiante, em detalhes, no que consistem cada uma dessas funções, quais suas diferenças em termos de entrevista, a fim de que você possa escolher a que melhor se adeque ao seus interesses, no intuito de beneficiar sua evolução profissional, em busca de uma carreira de sucesso.

Software Engineer (SWE)

Fábricas não podem construir sistemas críticos que possam colocar em risco a vida das pessoas, como aeronaves, reatores nucleares e sistemas médicos, sem contar com a ajuda do engenheiro de *softwares*. Todo esse processo precisa ser estudado para que os custos sejam estimados, funcionários contratados e os riscos de falhas minimizados.

Em áreas críticas como aviação, área espacial, medicina e até mesmo entretenimento (em um parque de diversões, por exemplo), o risco de uma falha de *software* pode colocar milhares de vidas em risco. Um engenheiro de *software* tem a habilidade em antecipar e evitar esses problemas antes que eles aconteçam.

Existem diversas formas de atuação no mundo do *software*. Abaixo, estão elencados os tipos mais comuns:

Frontend

O engenheiro de *frontend* é responsável por criar conteúdo com o qual os usuários possam interagir e ter uma boa experiência. Quando abrimos um site como o Facebook e começamos a pesquisar por amigos, todas essas funcionalidades visualizadas foram criadas por esse tipo de engenheiro. Isso significa que tudo

que visitamos, clicamos ou até mesmo os formulários que preenchemos foi criado por eles.

Parte das responsabilidades deste desenvolvedor incluem melhorar a experiência do usuário, dar vida ao conceito criado por um *designer*, criar ferramentas que possam ser utilizadas pelos usuários, criar um site que seja perfeito em seu celular, entre outras funcionalidades. Por isso, este tipo de profissional acaba focando no código que interage diretamente com o cliente e nos serviços que auxiliam a comunicação direta com o mesmo. Ele é o responsável pela parte “visível” do produto. Em geral, esse é o programador que entende de tecnologias como JavaScript, HTML, CSS, Dart e React. Esse tipo de programador que, muitas vezes, trabalha diretamente com os *designers* responsáveis por criar o *layout* da interface com o cliente.

Ao desenvolver um novo produto ou criar uma nova *startup*, é fundamental possuir em sua equipe um engenheiro de *frontend*. Afinal, todos os usuários querem uma boa experiência e a primeira impressão é a que fica.

Backend

Quando você abre o Google e começa a digitar sua pesquisa, é incrível como ela sabe até mesmo o que queremos digitar, né? O

Google autocompleta seu texto e assim que você pressiona *enter*, rapidamente todos os resultados estão à sua frente.

Embora você esteja interagindo com uma interface criada por um desenvolvedor *frontend*, existe todo um sistema por trás, responsável por prover todos os dados que serão visualizados por você.

O engenheiro *backend* trabalha na parte que não é diretamente “visível” para o usuário. Muitas vezes, ele é responsável por criar componentes críticos do sistema que serão utilizados pelo engenheiro *frontend*.

Inúmeros tipos de sistemas podem ser considerados sistemas de *backend*, por exemplo:

- O Instagram possui uma parte em seu *backend* que escolhe as fotos mais relevantes para você. Esse serviço é então integrado na aplicação e juntos essas funcionalidades são responsáveis por proverem uma boa experiência ao usuário final.
- Serviços de autenticação do cliente. Ninguém quer que seu e-mail possa ser acessado por outras pessoas! Esses serviços são responsáveis por dar as permissões corretas de busca, inserção e seleção do usuário no produto como um todo.

Esse tipo de engenheiro em geral possui conhecimento mais avançado de redes de computadores, sistemas operacionais e sistemas distribuídos. No entanto, a variedade de habilidades é grande e depende da experiência específica de cada pessoa.

Full stack

Em diversos estágios de uma empresa ou produto, você vai precisar fazer mudanças internas no *backend* e na interface com a qual o usuário interage. Imagine que você está criando um novo aplicativo com seus amigos e precisa terminar o mais rápido possível. Isso significa que por mais que você esteja trabalhando na interface com o usuário, nada vai te impedir de consertar erros no *backend*. Afinal, você está preocupado com a entrega e ninguém vai querer ficar na zona do conforto em um momento como este.

Podemos dizer que esse é o programador que é capaz de trabalhar tanto em *frontend* quanto em *backend*. Ele possui, em geral, uma capacidade de entender o sistema de forma completa. Além disso, ele deve entender como a interface interage com os sistemas internos, e vice-versa.

Em empresas grandes de tecnologia, esse é o tipo preferido de engenheiro. Afinal, quem não quer contratar alguém que possa fazer por dois? Em *startups*, esse tipo de trabalho é fundamental.

Essas firmas precisam de engenheiros que se adaptem rapidamente a novos desafios, mesmo que isso signifique alternar entre *frontend* e *backend*!

O time específico no qual você trabalha e os projetos do momento são o que vai definir de forma mais clara se você irá trabalhar mais no *frontend* ou no *backend*. É esperado que todos os engenheiros sejam capazes de aprender e executar projetos em larga escala em ambas frentes.

Security Engineer

O objetivo de um engenheiro de segurança é proteger as redes e os sistemas de uma empresa contra ataques cibernéticos. Ninguém vai querer realizar um pagamento em um site que já foi invadido por *hackers*. Trabalhar com segurança digital tem suas vantagens. Praticamente tudo que usamos no dia a dia tem algum componente relacionado à tecnologia. Por isso, as oportunidades nessa carreira são extraordinárias. Engenheiros de segurança são extremamente disputados por empresas de tecnologias devido a sua escassez, esse tipo de engenheiro em geral domina habilidades em diversas áreas tais como: redes de computadores, sistemas operacionais e programação em geral.

Engenheiros de segurança são responsáveis por testar e auditar *softwares*, além de monitorar redes e sistemas a fim de descobrirem falhas ou intrusos. Suas responsabilidades são, por exemplo: desenvolver um conjunto de padrões e práticas de segurança, criar novas maneiras de resolver problemas de

segurança já em produção e até mesmo simular ataques na própria empresa.

Production Engineer / Site reliability engineer (SRE)

Em grandes empresas de tecnologia, quando um serviço fica fora do ar ou algo fora do normal em geral acontece com um serviço, esse é o engenheiro responsável por fazer com que tudo volte ao normal. Ele garante que os serviços rodem de forma estável, sem muitos problemas. Qualquer alteração brusca no comportamento de um sistema é monitorado por esses engenheiros. Mesmo que não faça parte do time que escreveu o sistema, o SRE tem um grande conhecimento de como monitorar o sistema e rapidamente consegue investigar as possíveis causas que estariam levando o sistema à falha.

Engineering Manager

Normalmente, *engineering managers* são responsáveis, dentre outras funções, por criarem o plano para o desenvolvimento de novos produtos, recrutamento e gestão de pessoas. Esse

engenheiro dedica a maior parte do seu tempo nessas tarefas, diferentemente da maior parte dos engenheiros que focam mais na parte de programação e desenvolvimento do produto. Ainda assim, é comum que alguns desses profissionais escolham por nunca se tornarem *managers*, ou seja, optam por continuar trabalhando como contribuidores individuais, independente de sua senioridade na corporação.

O *engineering manager* deve sempre pensar em alocar/criar projetos que se alinhem com as habilidades e objetivos de carreira dos seus funcionários. O alinhamento dos interesses da empresa com o interesse do funcionário é fator primário para fazer com que os engenheiros se sintam motivados e o sucesso do projeto seja amplificado. A visão de conseguir alinhar esses interesses é o que muitas vezes diferencia um *engineering manager* sênior e júnior.

Programador de Elite: O início de uma jornada

“O sucesso é caminhar de falha em falha sem perder o entusiasmo”

Winston Churchill

Inicialmente, é preciso esclarecer uma coisa: **aprender a programar é difícil!** Mas não encare como um obstáculo, adquirir uma habilidade é sempre uma jornada que não vem de graça, demanda tempo e esforço dos que estão aptos a enfrentá-la, ou você acha que o Cristiano Ronaldo aprendeu a jogar bola de um dia para o outro? Ou que um campeão de natação olímpica aprendeu a nadar através de um tutorial no YouTube? Certamente não!

Bom, se você continuar lendo, é um bom sinal. Significa que não tem medo das dificuldades e está disposto a enfrentá-las para atingir os seus objetivos. Nós também somos assim, e certamente todo programador de elite também é.

Como falamos, aprender a programar é de fato uma jornada. Mas não se sinta sozinho, pois nós estamos aqui para trilhar esse caminho lado a lado com você. Veja o programador de elite como um guia e um mentor que vai te ajudar a escolher o seu próprio caminho

e servir como um norte nos momentos de dúvidas e frustrações que surgirão durante essa jornada.

Agora, já sabendo que pode contar conosco, vamos ao que interessa.

Aprender a programar é um processo mental que ocorre de forma semelhante a aprender qualquer outra habilidade. Geralmente, acontece em quatro diferentes estágios de aprendizado:

- **Inconsciente incompetente**: Nesse estágio, você não possui a habilidade e também não está ciente de que precisa tê-la. O lado bom é que se chegou até aqui, certamente tem consciência que precisa aprender a programar.
- **Consciente incompetente:** Nesse estágio, você ainda não possui a habilidade, mas já reconheceu que precisa tê-la e já está fazendo o necessário para aprendê-la. Durante essa fase, falhar vai ser inevitável e até desejável. Serão os erros que vão remodelar as conexões no seu cérebro até você conseguir programar sem muita ou com pouca ajuda.
- **Consciente competente:** Esse estágio é interessante, pois nele você já tem a habilidade, mas ainda precisa de esforço mental para executá-la. Provavelmente, já sabe programar, mas ainda precisa de ajuda ou ainda não é auto-suficiente. Mais

adiante, explicaremos como conseguir dar o salto para o próximo e tão esperado estágio de um bom programador.

- **Inconsciente competente:** Nesse estágio, você já tem a habilidade e consegue executá-la de forma praticamente inconsciente, da mesma forma que consegue ler ou escrever. É aqui que moram os programadores de elite. Programar, para eles, passa a ser tão trivial quanto beber um copo de água ou ler um jornal. Porém, não se assuste: todo mundo pode chegar nesse estágio com tempo suficiente. Nosso objetivo é ensiná-lo o suficiente para você poder chegar aqui com as próprias pernas e, a partir daí, ter a liberdade de poder escolher o seu futuro como um programador.

Identificar em qual estágio você se encontra é essencial para saber quais objetivos devem ser traçados e como orientar os seus estudos. Por exemplo, se você ainda encontra-se na fase consciente incompetente, então é preciso focar em aprender os assuntos básicos de programação (Condicionais, Laços, Funções, etc), além de direcionar a sua prática para projetos simples. Já na fase consciente competente, você pode começar a se aprofundar em assuntos mais avançados, como algoritmos e estrutura de dados,

design de sistemas, além de praticar com projetos mais complexos. Se até agora nada disso fez sentido para você, não se preocupe, existe um capítulo dedicado para cada uma dessas fases, onde explicaremos de forma mais aprofundada o que é primordial estudar, como praticar e quais objetivos devem ser traçados, de acordo com o estágio em que você se encontra.

Antes de progredir para os próximos capítulos é interessante entender a importância e a necessidade de passar por todas essas fases sem pular etapas, afinal de contas, todo programador de elite um dia já foi um inconsciente incompetente. Portanto, não precisa se preocupar se achar que está demorando demais para evoluir de estágio, cada pessoa aprende no seu tempo e corre a sua própria maratona. O que importa, no final, é buscar uma evolução diária, mesmo que a considere marginal.

Aprendendo a programar

O consciente incompetente

Se aprender a programar é uma maratona, esse estágio é certamente a largada. É aquele momento em que as coisas ainda não estão muito claras em relação ao que vai acontecer e qual caminho seguir, mas, de toda forma, está disposto a continuar e sabe que o objetivo é ultrapassar a linha de chegada e ganhar o direito de se considerar um programador de elite. É sempre bom lembrar que essa maratona não possui atalhos. Você vai ter que seguir todos os passos.

Podemos considerar esse primeiro estágio como um aquecimento. Você vai aprender o básico de programação, como se manter motivado, como focar no que é importante e, principalmente, aprender a aprender sozinho. Tornar-se auto suficiente e conseguir aprender sozinho é a principal habilidade de todo programador de elite.

A liberdade de aprendizado vai te permitir trabalhar onde quiser e com o que quiser, porém sem colocar a carroça na frente dos bois.

Vamos iniciar com o básico, que consiste em ensinar os fundamentos de programação e toda a parte comportamental para

conseguir manter o foco e a motivação necessárias durante essa jornada.

E como talvez você já saiba, o lado bom da programação é que a quantidade de recursos disponíveis na internet é praticamente ilimitada, ou seja, independente do tópico que você esteja estudando, algum especialista já escreveu algo sobre ou gravou uma aula a respeito desse assunto na internet. Apesar de não existir um único lugar em que você consegue encontrar todos esses tópicos, alguns sites e blogs na internet conseguem organizar muito bem os principais assuntos que serão necessários para ganhar tração e sair de fato do absoluto zero. Mais adiante, vamos apresentar alguns desses recursos e como melhor utilizá-los, objetivando otimizar os seus estudos.

Antes de entrar de fato na parte técnica e mostrar quais assuntos deverão ser estudados, achamos válido passar algumas dicas relacionadas ao foco e à motivação. Afinal de contas, de nada vai adiantar escrever um livro, que ensina tudo bem detalhado, sem procurar orientá-lo a criar disciplina para estudar.

Um dos comportamentos que costuma acontecer com frequência é a sobrecarga de recursos. É um pouco contraditório, mas existem tantos recursos disponíveis na internet sobre

praticamente tudo que algumas pessoas costumam se sentir perdidas, sem direção e, consequentemente, sem saber por onde começar. E isso de fato faz sentido, apesar da riqueza de recursos, falta um pouco de organização no que se encontra na internet. E não é só organização em relação a ordem em que estudar cada tópico, mas também em relação ao próprio tópico. Por exemplo, você pode achar um artigo que te explica que para aprender a programar você deve começar com a linguagem de programação Python e ao mesmo tempo encontrar outro artigo que te diz que você deve começar por Java. E indo mais além, você consegue encontrar diversos outros artigos que explicam Python e Java de uma forma completamente diferente. E aí, por onde você começa e qual desses caminhos você deve confiar? A resposta é: Não existe certo ou errado sobre como começar, o que importa é você começar de alguma forma!

Vamos ensiná-lo um caminho que pode ser seguido, pois nós já passamos por ele e ajudamos outras pessoas a fazer o mesmo. Entretanto, não se prenda tão somente aos nossos ensinamentos. Há um universo de programação enorme, além do programador de elite e, embora o nosso livro seja completo em relação ao que é

preciso para se tornar um grande programador, é de suma importância que você continue sempre aprendendo mais e mais.

O lado bom é que se você chegou até aqui, de uma forma ou de outra, já demonstrou que tem sede por conhecimento e está disposto a sair da zona de conforto para aprender algo novo e tão complexo quanto programação.

Durante essa jornada, um dos primeiros problemas que provavelmente você precisará enfrentar será o de conseguir encontrar motivação diária, até porque, como falamos no início, aprender a programar leva tempo, ou seja, é preciso enfrentar uma longa jornada.

Geralmente o padrão que enxergamos em quem acaba se desmotivando no meio do caminho é que essas pessoas terminam por focar em um objetivo muito grande e muito distante, ignorando evoluções menores, que são conquistadas diariamente e, dessa maneira, vão perdendo a motivação, pois acham que não estão evoluindo.

E é por isso que se você está começando, saber fracionar bem os seus objetivos vai ser muito importante. Por exemplo, se você acabou de começar a aprender uma nova linguagem de programação, não faz sentido o seu primeiro objetivo ser construir

um jogo 3D super complexo. Seus objetivos precisam ser compatíveis com a fase em que você se encontra. Ora, um primeiro objetivo que faria mais sentido seria somar dois números ou mostrar o seu nome na tela de um computador por exemplo, que é muito mais realístico e alinhado com quem está entrando no mundo da programação.

Uma dica interessante que podemos dar para quem está iniciando é imaginar tudo isso como um jogo, onde você se encontra em uma fase A, quer evoluir para uma fase B, mas para isso precisa enfrentar diversos obstáculos menores durante o caminho. Fazer isso vai tornar esse processo um pouco mais simples e mais divertido de se participar. Mas enfim, vamos ao que interessa. A seguir vamos falar de algumas dicas comportamentais que vão te ajudar a se manter no eixo durante esse trajeto. Não se preocupe em lembrar de todas elas agora, mas sempre que se sentir um pouco perdido, revisitar o que vamos escrever pode te ajudar a manter o foco e direção no que é importante.

Definir objetivos menores

O ponto importante de definir objetivos menores é poder ver o seu progresso acontecer de uma forma mais clara. Traçar objetivos

maiores vai diluir a sua percepção de evolução e te dar a sensação de que você continua estagnado.

Antes de tudo, crie uma lista de pequenos objetivos atingíveis, como assistir uma vídeo aula, ler mais uma página deste livro ou de qualquer outro ou escrever 10 linhas de código e se comprometa a fazê-lo diariamente. Conseguir completar essa lista vai te dar uma sensação de dever cumprido e vai tornar palpável todo o progresso que você conseguiu atingir. E não importa se você leu apenas uma página de um livro ou assistiu 10 minutos de uma vídeo aula, se você traçou isso como meta e conseguiu atingir, naturalmente você vai sentir o prazer de dever cumprido.

Além disso, é importante que você constantemente tenha consciência do seu progresso e celebre o mesmo. Pode ser que às vezes você não consiga ver isso de forma tão clara, mas de uma coisa você pode ter certeza: todo dia que você escreve ao menos uma linha de código, você está naturalmente melhorando. Uma dica importante que podemos dar é manter um diário onde você mantém anotado tudo o que você aprendeu e o que você está tendo dificuldade no momento. É bastante provável que em um ou dois meses você consiga enxergar que não está mais tendo esse mesmo problema.

A importância de fazer o que você gosta

Você muito provavelmente já ouviu falar dessa frase: “Escolha um trabalho que você ame e não terás que trabalhar um único dia da sua vida”. Por mais que não acreditamos que isso seja 100% verdade, afinal de contas, nem sempre você vai conseguir trabalhar só com o que ama, a essência dessa frase é de você fazer o que gosta pois isso vai gerar mais prazer e consequentemente, te manter mais motivado.

Todo bom programador passou por problemas de motivação no começo e, geralmente, o que conseguimos notar é que boa parte dessas pessoas que conseguiram passar por essa barreira, começaram a programar por *hobby* e utilizavam a programação para criar coisas do seu próprio interesse. Por exemplo, é bem comum você ouvir a história de um grande programador que começou pois queria desenvolver um jogo próprio, uma modificação de um jogo existente ou até um website do seu interesse. E é por isso que quando recebemos a pergunta sobre qual linguagem de programação começar ou o que é melhor aprender, geralmente

respondemos que você deve seguir o que você mais tem interesse e o que você acha mais fácil por consequência.

Então principalmente nesse início, não se preocupe com linguagem de programação, tecnologia ou o que começar a desenvolver. Escolha algo que você goste bastante ou que tenha curiosidade e procure criar uma versão simples disso. Por exemplo, se você gosta de jogos, pense em um jogo que seja simples o suficiente e esteja dentro da sua realidade de momento. Mas tenha cuidado, é importante ter em mente que você ainda está em fase de aprendizado, então o projeto inicial que você escolher deve ser adequado a essa realidade. Afinal de contas, não faz sentido você decidir desenvolver o próximo *League of Legends* ou qualquer outro jogo complexo, pois eles demandam muito tempo e um conhecimento avançado de programação.

A síndrome do impostor

Por incrível que pareça, às vezes os programadores podem ser os seus próprios inimigos. É impactante e ao mesmo tempo libertador quando você descobre que é bastante comum que programadores, inclusive os de alto escalão, duvidem de si mesmo e

muitas vezes tenham a sensação de que chegaram até onde estão por sorte. Por exemplo, é bastante comum que em algum momento você ache que todos ao seu redor sabem muito mais e são bem melhores do que você, mas isso realmente não é verdade.

E sabemos disso pois já passamos por todas essas situações e hoje que já estamos mais estabilizados, conseguimos analisar todo esse passado e inferir que isso não passou de uma ilusão. Por exemplo, quando começamos a trabalhar para o Facebook e Google, era comum encontrar as estrelas do mundo da programação, como os criadores do C++, Haskell e de sistemas operacionais, ou seja, pessoas responsáveis por praticamente moldar a computação como um todo, além de conhecer diversos estudantes de universidades renomadas mundialmente, como Harvard e MIT. E em situação como essas é bastante natural que você desenvolva o sentimento de inferioridade e não pertencimento.

E aqui vem a nossa dica de como ultrapassar essa barreira: **consistência**. Não deixamos isso nos abalar de forma alguma, continuamos fazendo o que sempre fizemos, que é programar. E não somente isso, utilizamos todo esse sentimento de inferioridade como um combustível para aprender e trabalhar mais do que o restante dos programadores ali presentes.

Então resumindo tudo, toda vez que você se sentir assim, pegue o seu computador e programe mais um pouco. Utilize todo esse sentimento negativo para te impulsionar e consequentemente transformar em algo positivo. Lembre-se, a consistência é a chave para o sucesso de todo programador.

A importância da língua inglesa

Uma das poucas verdades **absolutas** e **inquestionáveis** que o mundo inteiro da programação consegue concordar é de que o inglês é **completamente obrigatório**. Então, todas as vezes que nos perguntam se é necessário aprender inglês, sempre respondemos sim, independente da situação e da pessoa.

Podemos passar o restante do livro inteiro escrevendo motivos, mas acho que o fato de que a grande maioria dos recursos relacionados a programação estejam em inglês já deveria ser explicação o suficiente. E não só isso: se você não souber inglês, vai acabar limitando o seu escopo de trabalho a algumas empresas brasileiras e, consequentemente, ser excluído de todas as outras que constituem a grande maioria do mercado de programação.

A seguir, vamos te dar algumas dicas do que fazer para melhorar na língua, mas não se limite somente a isso. Recomendamos sempre que você procure algum especialista para te guiar no estudo da língua.

- **Assistir filmes e seriados com legenda em inglês:** Essa é uma dica que funciona muito bem. De uma forma ou de outra, você vai forçar a sua mente a se acostumar com essa nova língua. Faça isso gradativamente e de preferência com filmes que sejam mais simples de entender para que você possa acompanhar o contexto do que está se passando.
- **Ler livros em inglês:** Semelhante a assistir filmes e séries, ler livros em outra língua tem o mesmo efeito na sua mente. Aqui, você pode consultar um dicionário ou o próprio Google Tradutor caso você esteja empacado em alguma palavra ou frase.
- **Praticar a escrita:** Escrever bastante na língua vai te ajudar a memorizar melhor o vocabulário e entender como as palavras se conectam para formar frases. Além disso, lembre-se de que o código de programação deve ser escrito em inglês. Logo, praticar isso escrevendo outras coisas vai te dar uma certa vantagem.

- **Aplicativos:** Os mais conhecidos como o Duolingo e o Memrise podem te ajudar também. Geralmente, eles possuem cursos inteiros que você pode seguir a qualquer momento sem sair do conforto da sua casa.
- **Conversar com estrangeiros na internet:** Uma das formas mais eficientes de aprender uma nova língua é manter contato com quem é falante nativo. Procure então participar de comunidades e conversar com estrangeiros em língua inglesa. Essa é uma excelente forma de aprender na prática. É mais interessante ainda, porém não obrigatório, se essa comunidade estiver relacionada à programação, pois nesse caso você melhora suas habilidades de programação e, ao mesmo tempo, pratica a língua inglesa.
- **Intercâmbio**: De todas as opções, essa é mais rápida e mais eficiente. Realizar um intercâmbio em outro país onde o idioma oficial é a língua inglesa vai fazer com que você fique fluente em questão de meses. Mas tome cuidado: se você tiver essa oportunidade tente diminuir o contato com brasileiros ou evite falar português ao máximo. Fazer isso vai te colocar dentro de uma bolha onde você não vai praticar a língua como deveria.

Essas são algumas dicas que você pode seguir, mas, como reforçamos anteriormente, não se prenda somente a elas. Existe uma quantidade quase que infinita de recursos disponíveis na internet e, na dúvida, procure sempre um especialista ou alguém que vai te guiar melhor.

Não importa a linguagem de programação

Muito provavelmente a pergunta mais frequente que costumamos receber de quem está iniciando no mundo da programação é: “Qual linguagem de programação devo aprender primeiro?”. É uma dúvida super válida, afinal de contas, existem centenas de linguagens de programação disponíveis e sempre vai existir alguém que vai te falar que linguagem de programação A é melhor que B, e vice-versa. Mas então, qual linguagem de programação você deve escolher? A resposta é simples: **qualquer uma!** É uma afirmação impactante para quem está começando, mas essa é a mais pura verdade. Passe a enxergar aprender uma linguagem de programação como aprender a dirigir um carro: uma vez que você aprende a dirigir um Celta, você consegue dirigir praticamente qualquer outro carro sem precisar passar por uma auto-escola novamente.

E a explicação para isso é simples: uma linguagem de programação nada mais é do que um emaranhado de palavras que, se colocadas na forma correta, o computador vai conseguir compreender tudo aquilo e transformar em um *website*, um aplicativo

ou qualquer outro *software*. E geralmente o que muda de uma linguagem para outra são apenas algumas dessas palavras e a ordem em que elas são colocadas. Ainda assim, a essência costuma ser a mesma. Então, depois que você aprende bem uma linguagem e o mecanismo por trás dela, se dar bem nas outras vai ser questão de adaptação.

Além disso, o mundo da programação está em constante movimento e mudança, então não adianta você focar muito em uma linguagem de programação. Existem boas chances de ela não ser mais utilizada daqui a alguns anos.

Então, resumindo, não se preocupe com linguagens de programação. Escolha a que você tem mais facilidade e afinidade com e foque em entender bem como ela funciona. Com o tempo, você vai perceber que as coisas vão ficar mais claras na sua mente e mudar de uma linguagem para outra vai ser trivial. Mas se você insiste em uma recomendação, indicamos a Python por ser comumente considerada a de mais fácil aprendizado. Mas como falamos: se você não achar Python fácil de aprender, não se preocupe, afinal de contas, não importa a linguagem de programação.

Saindo de fato do zero

Agora que esclarecemos todos os aspectos comportamentais de ser um bom programador, vamos colocar a mão na massa e te ensinar o que você de fato pode fazer para sair do zero e se tornar um programador de elite. Mas antes de tudo, vamos deixar uma coisa clara: nosso objetivo com este livro não é te ensinar detalhadamente assunto por assunto, mas sim te dar as ferramentas necessárias para você conseguir fazer isso com as próprias mãos. Afinal de contas, aprender a aprender é uma característica essencial que todo programador de elite deve ter.

E sobre aprender a aprender: vivemos em uma era onde isso nunca foi tão fácil já que existem milhares e mais milhares de cursos, vídeo aulas e artigos na internet que vão te ensinar de tudo, desde aprender o que é um laço até construir um sistema operacional. Se você voltar uns 10 anos no tempo, provavelmente você não teria a mesma sorte. Nos tempos de hoje, as opções são fartas. Você pode fechar este livro e encontrar um curso do que você quiser em menos de 5 minutos na internet.

A seguir, vamos falar sobre alguns dos recursos que recomendamos e como você pode utilizar melhor cada um deles de acordo com a situação. E mais uma vez: não se prenda somente a

esses recursos! Utilize a internet como uma ferramenta de exploração para continuar sempre aprendendo coisas novas.

Cursos online

Essa é uma das sete maravilhas para quem quer aprender a programar. Plataformas onlines competem entre si para decidir quem consegue oferecer o melhor curso de programação para iniciantes e quem se beneficia com isso é você, que pode escolher gratuitamente qual curso começar e, caso não goste, pode sair a qualquer momento e pular para outro.

E com tantas opções disponíveis, seria relativamente fácil preencher este livro inteiro apenas com cursos de programação na internet. Mas como esse não é o nosso objetivo, vamos listar aqui algumas opções e quais os diferenciais entre elas.

- **Code Academy (www.codeacademy.com):** Essa é a plataforma mais conhecida e mais utilizada entre iniciantes do mundo da programação, e é justificado. As aulas disponibilizadas por eles geralmente são bem interativas e fáceis de entender. Você escolhe o que quer aprender, tem acesso a respostas imediatas se o que você fez está certo ou não e ainda tem a possibilidade de desenvolver projetos reais de programação.

- **Codewars (www.codewars.com):** Codewars é uma forma divertida de aprender a programar. O programa de ensino deles é baseado em desafios que você pode completar e progredir de nível, assim como em um jogo de videogame. É uma boa forma de aprender a técnica e ao mesmo tempo se manter motivado.
- **freeCodeCamp (www.freecodecamp.org):** Esse é um dos nosso favoritos. O freeCodeCamp te ensina a programar através de um currículo pré-estabelecido por eles e depois te dá experiência na prática ao possibilitar que você trabalhe em projetos reais para empresas sem fins lucrativos. É uma plataforma perfeita para quem quer aprender a programar e ao mesmo tempo adquirir experiência real com programação.
- **GA Dash (dash.generalassemb.ly):** O GA Dash é uma plataforma de ensino a programação totalmente voltada para projetos. Você escolhe qual projeto trabalhar de acordo com o seu nível e o GA Dash vai te guiar durante todo o caminho.
- **Khan Academy (www.khanacademy.org):** O Khan Academy é uma plataforma de ensino que também oferece cursos de programação gratuitos. Apesar de ser mais voltada para o público mais jovem, os cursos de programação costumam ser

bem completos e auto explicativos, perfeitos para quem está começando.

- **Udacity (www.udacity.com):** O Udacity é também uma plataforma de ensino voltada com bastante foco em cursos de programação. O grande diferencial dela é te permitir fazer os nanocursos, onde basicamente você pode escolher uma carreira específica e eles vão te mostrar cursos que você pode seguir para otimizar o seu caminho.
- **Hackr.io (www.hackr.io):** O Hack.io não é um curso ou uma plataforma. Na verdade, é um site que agrega uma gama de recursos relacionados a programação. Basta você digitar o que deseja pesquisar e eles vão te apresentar uma lista de artigos, cursos onlines e outros livros que são recomendados pela comunidade mundial de programação. É uma excelente ferramenta para servir de guia para quem está começando.
- **Edabit (www.edabit.com):** O Udabit é uma plataforma que vai te ensinar a programar quebrando os assuntos de programação em pequenos desafios ao simular problemas reais. É uma plataforma bem dinâmica e bem fácil de utilizar.

Achou longa essa lista? Olha que isso é apenas uma pequena amostra do que você pode encontrar na internet. Como falamos

anteriormente, a quantidade de recursos disponíveis é praticamente infinita. Mas utilizando algumas dessas plataformas você provavelmente vai estar bem encaminhado seja qual for a sua escolha. E não se preocupe se você não gostar de uma. Pule para a próxima e continue aprendendo. Isso é o que chamamos de liberdade de conhecimento. Você escolhe o que aprender, como aprender e quando aprender.

Programar é um processo de imersão

Aprender a programar é muito mais prática do que teoria. Então, não adianta muito você ler uma centena de livros de programação se você não coloca isso em prática TODOS OS DIAS. Não que eu esteja falando que você deve passar horas e horas programando, mas separe ao menos alguns minutos do seu dia para praticar tudo o que você aprendeu. Essa é a única forma de ficar fluente em programação.

Aprender a programar é basicamente aprender a falar com o computador. Logo, você tem que estar falando constantemente. Pessoas que estão iniciando geralmente costumam praticar longas horas, mas apenas aos fins de semana ou quando tem tempo livre. E isso não funciona, pois tudo que você aprende a exercer com

naturalidade é fruto da constância que você aplica durante o aprendizado dessa habilidade, o que requer prática e estudo diário. E aí recebemos muitas perguntas do tipo: “Eu estou muito ocupado, como vou achar tempo para programar diariamente?” Bom, a resposta é: você realmente quer aprender a programar? Se sim, você tem que entender que vai ser preciso abdicar de algumas coisas para conseguir atingir esse objetivo, e isso vai incluir trocar algumas horas de diversão por horas estudando e praticando programação.

E podemos afirmar isso por experiência própria: principalmente quando você está começando, se você parar nem que seja uma semana, existem boas chances de você esquecer o que aprendeu e ter que estudar tudo novamente. Então, encare programação como uma imersão. É um conceito novo e boa parte do tempo você vai se ver encarando a tela do computador tentando visualizar e entender o que está acontecendo frente aos seus olhos.

Projetos pessoais

Quando você está aprendendo, é essencial que você passe boa parte do tempo construindo projetos pessoais - códigos reais

que você pode utilizar em algum contexto.

Cursos e tutoriais são excelentes formas de aprender o básico, mas o que vai te ensinar a virar um programador profissional é começar a construir programas que possuem utilidade real. Essa é a fase em que você começa a entender melhor os conceitos de programação e como eles podem ser úteis no mundo real. Fazendo um paralelo, é como você aprender o vocabulário Russo em um curso e depois ir em um restaurante e pedir um prato utilizando tudo o que você aprendeu.

Mas é bom que fique claro: quando falamos para você trabalhar em projetos pessoais, não estamos dizendo para você criar o próximo Facebook ou um jogo como o *League of Legends*. Longe disso! Pode ser qualquer coisa simples que você tenha interesse em fazer e tenha alguma forma de utilidade. Por exemplo, você pode decidir fazer uma mini calculadora, um joguinho de linha de comando ou qualquer outra coisa que seja tangível a sua realidade.

A medida que você for melhorando, esses projetos podem começar a se tornar mais complexos e incorporar mais conhecimentos do mundo da programação. Por exemplo, um dos nossos amigos quando estava prestes a entrar na universidade decidiu criar um site na internet onde as pessoas que fizeram a prova

poderiam colocar as suas respectivas notas e o site iria imprimir quem estava aprovado ou não baseado no conjunto de notas de todo mundo. Foi um projeto relativamente simples (apesar de não ser tão iniciante), mas que possibilitou aprendizado prático para ele.

E durante esses projetos pessoais, é completamente normal se você precisar buscar ajuda na internet, e isso também inclui pesquisar e utilizar códigos prontos feito por terceiros. Mas fique atento: no lugar de apenas copiar e colar o código, tente digitá-lo. Isso vai te fazer pensar um pouco mais sobre o que você está digitando e, consequentemente, te ajudar a entender a razão por trás deste código.

Resumindo tudo: comece a trabalhar em projetos pessoais pequenos e tangíveis o quanto antes. Isso vai te permitir aprender a programar muito mais rápido.

Estudando através de códigos de terceiros

Você já parou para ver o quão interessante é o processo de aprendizado de uma criança? Quando elas querem saber como funciona um carrinho de brinquedo, o que elas costumam fazer? Desmontam e tentam montar o carro novamente.

E esse processo funciona para quem quer aprender a programar também. Você não necessariamente precisa aprender tudo do zero. Você pode pesquisar algum código que já faz o que você quer e destrincha-lo até entender os mecanismos que fazem ele funcionar. Por exemplo, se você quer aprender como é feito a programação de uma calculadora simples, você pode pesquisar na internet uma calculadora que faz contas de soma e subtração e depois tentar modificar esse código para adicionar a possibilidade de realizar multiplicação e divisão também. Essa é uma prática bastante comum e também uma das raízes por trás de códigos *open source*, que são feitos com o intuito de educar outros desenvolvedores.

E o lado interessante é que essa habilidade não é só útil para quem é iniciante, mas profissionais da área costumam utilizar isso extensivamente, principalmente porque quando você passa a trabalhar para uma nova empresa ou troca de time internamente, é essencial que você consiga ler e entender códigos que foram escritos por outras pessoas.

Então não tenha medo nem vergonha. Não só é normal fazer isso, como é completamente recomendável para quem quer se dar bem nessa área.

A importância de ter um mentor

A importância de ter um mentor

Aprender a programar pode ser uma jornada um pouco solitária. Geralmente, a maior parte do tempo você vai passar brigando com o computador e tentando entender porque ele não está obedecendo os seus comandos.

É por isso que ressaltamos a importância de se encontrar um mentor ou um grupo de amigos que também gostem de programação. Participar de *hackathons*, palestras e eventos relacionados a programação são boas formas de se construir amizades e procurar pessoas que estejam dispostas a te ajudar. Não caia no conto de que todo programador é anti-social e não possui amigos. Muito pelo contrário! Os bons programadores que atingem cargos altos nas grandes empresas são geralmente as pessoas mais bem relacionadas.

Existe um limite até onde você consegue chegar sozinho. Sem a ajuda de outras pessoas, você vai estar limitando o seu alcance. Então, tenha sempre em mente que fazer novas amizades nesse ramo é muito positivo e isso inclui encontrar uma pessoa que pode servir de mentor para você nessa jornada.

Afiando as habilidades e se destacando dos demais

O inconsciente competente

Essa é a meca dos programadores, ou seja, onde vive a alta sociedade do mundo da programação e aqueles que intitulamos de Programadores de Elite. Como afirmamos anteriormente, são essas pessoas que podem escolher onde trabalhar e com o que vão trabalhar, pois chegaram em um nível de conhecimento e experiência tão alto que programar se tornou um ato tão mecânico quanto beber um copo de água.

E por mais estranho que pareça, não existe nenhum segredo ou fórmula mágica para te fazer chegar até aqui. O que separa um programador mediano de um programador de elite é a quantidade de horas colocadas em prática. Conhecimento é criado basicamente através da repetição. Você repete tanto uma certa atividade que, em um certo ponto, tudo vai começar a acontecer no automático. Quantas repetições vão ser necessárias ou quanto tempo todo esse processo vai levar varia muito de pessoa para pessoa. O que

podemos afirmar com certeza é que um programador de elite possui milhares de horas de prática no currículo.

Então, não adianta tentar pular etapas ou se enganar nesse processo. A única coisa que vai te salvar de verdade é ter consistência e disciplina para estudar e praticar todos os dias, religiosamente. E como o seu objetivo é programar a nível profissional, não existe outra forma que não seja colocando a mão na massa em trabalhos reais. Isso significa que necessariamente você vai ter que ir atrás de um estágio, trabalhar em *open source*, remoto ou de forma autônoma, para ganhar a experiência que precisa para atingir o templo dos programadores. Nos próximos capítulos, vamos falar exclusivamente sobre o que você pode fazer para conseguir isso.

Conseguindo a entrevista

Pode parecer óbvio, mas o currículo é a primeira parte do processo, infelizmente muitas pessoas não dão a importância necessária e acabam não conseguindo a oportunidade de ser entrevistado.

Grandes empresas recebem milhares de currículos diariamente e tem a árdua tarefa de selecionar os melhores. Muitas vezes, o seu currículo é inicialmente analisado por pessoas que entendem um pouco da área de tecnologia e não são tão técnicas quanto você, por isso é importante usar uma linguagem clara e concisa para ser um destaque entre milhares de outros candidatos.

Abaixo sugerimos alguns pontos que te levarão a um currículo mais completo.

Mantenha curto e seja objetivo

Currículos com informação excessiva distraem o recrutador das partes que realmente importam, com tanta informação disponível é possível que as partes não essenciais ou que não demonstrem o seu melhor sejam as únicas que o recrutador leia. O currículo é

apenas um primeiro filtro nos candidatos durante as entrevistas, você terá tempo para demonstrar o que você sabe em outras etapas do processo.

Antes da entrevista, é comum engenheiros analisarem seu currículo para falar sobre alguns pontos, se o seu currículo tem muita informação, pode ser que a parte mais importante não seja levada em consideração e vocês fiquem boa parte da entrevista falando de assuntos que você talvez não ache mais tão relevante.

Não enrole. Currículos com textos longos acabam distraindo os recrutadores e escondendo informações importantes sobre você. Escreva de forma clara e objetiva seus maiores e mais relevantes feitos.

Mostre o melhor de si

Assim como o ponto anterior, lembre-se que o tempo de todo mundo é limitado, entre milhares de candidato, porque deverão te escolher? Por exemplo, imagine que você é engenheiro do YouTube, ao mencionar isso no seu currículo nem todos sabem quantas pessoas usam diariamente o serviço e nem terão tempo suficiente para pesquisar, uma dica é sempre acrescentar números expressivos

que impressionam o leitor, como por um exemplo, um engenheiro da Google que trabalha na parte de fotos pode mencionar que o seu sistema é responsável por bilhões de operações diariamente.

Não minta

Pode parecer óbvio, mas não importa o quão bom você seja, nenhuma empresa quer trabalhar com pessoas desonestas, grandes empresas estão interessadas se você aprende com seus erros e nao nos seus erros em si.

Se preparando para a entrevista

A entrevista é o momento no qual você tem a oportunidade de demonstrar suas habilidades e convencer os entrevistadores de que é a pessoa certa para aquela vaga. Mesmo com o currículo perfeito e conhecimento em diversas áreas, o seu desempenho é o principal ponto levado em consideração para ser contratado. Portanto, dominar os diferentes tipos de entrevista é fundamental para se diferenciar dos outros candidatos. São inúmeros os candidatos com currículo e experiência exemplares que acabam não se preparando corretamente para a mesma e, por isso, não sucedem.

A principal entrevista, especialmente para engenheiros recém formados ou com pouca experiência, é de programação. No entanto, mesmo que você seja especialista nesse assunto, é importante lembrar que cada entrevista possui um formato diferente. Certos detalhes devem ser estudados a fim de evitar surpresas e existem diversos outros aspectos que são levados em consideração durante a avaliação do candidato. Por isso, é importante que mesmos os engenheiros mais experientes não deixem de se preparar para a

entrevista; revisar algoritmos e praticar sua comunicação pode ser fundamental para garantir uma boa performance. Mesmo que programar seja parte do seu dia-a-dia, é importante saber que uma entrevista possui aspectos diferentes das tarefas que você executa normalmente.

Antes de explicar como se preparar para as entrevistas, iremos classificar os engenheiros em dois grupos, baseado-nos em suas experiências:

Candidatos recém formados e com pouca experiência

Anualmente, milhares de pessoas terminam a faculdade e são elas que estão ativamente em busca do seu primeiro emprego. Esse, portanto, é o tipo de candidato mais comum para as empresas.

Não se espera que um recém-formado tenha um conhecimento profundo de como criar sistemas ou liderar times de vários engenheiros. Portanto, as principais entrevistas para eles são a de comportamento e de algoritmos e programação. As empresas esperam que um candidato júnior seja capaz de programar e executar de forma rápida as suas tarefas. Espera-se que esse candidato tenha um relacionamento bom com seus colegas e

gerente e que seja capaz de evoluir de maneira rápida. Afinal, em todo emprego novo existe um longo aprendizado e o quanto você for capaz de aprender definirá sua evolução na empresa a longo prazo.

Candidatos com experiência

Engenheiros mais experientes são capazes de executar as mesmas tarefas que um engenheiro júnior. Ainda assim, além dessas funções, eles devem ser capazes de direcionar o seu time e liderar outros engenheiros com menos experiência. Esse tipo de candidato possui experiência para fazer o *design* de sistemas, conectando várias partes que poderiam ser executadas por múltiplos engenheiros. Portanto, para avaliar esse tipo de candidato, as empresas aplicam outros tipos de entrevistas, tais como a entrevista de *design* de sistemas e a entrevista comportamental e de liderança.

As entrevistas

Apesar do processo de entrevista ser específico para cada empresa, formatos mais estabelecidos são compartilhados por diversas organizações. Contudo, é fundamental lembrar que o conjunto específico de entrevistas do candidato depende de muitos

fatores, tais como o cargo, o nível de experiência e a empresa. Podemos citar como algumas das entrevistas mais comuns:

- Algoritmos e programação
- Comportamento e liderança
- Design de sistemas
- Mini projeto

Perceba que nenhum processo é perfeito e entrevistas podem eliminar candidatos que teriam um excelente desempenho na empresa. No entanto, as entrevistas buscam seguir um formato padronizado para facilitar a comparação entre os candidatos de forma mais justa. Por isso, mais uma vez, é essencial se preparar para o formato específico de cada uma delas.

Nas seções abaixo, detalhamos os tipos mais importante de entrevistas. Além disso, explicamos como você pode melhor se preparar para cada um deles.

Algoritmos e programação

Quando se fala em entrevistas de empresas de tecnologia, a primeira coisa que vem em mente são as entrevistas de programação, algoritmos e estruturas de dados. Essa é, muitas vezes, uma habilidade essencial em qualquer empresa, e por isso bastante exigida durante entrevistas. Foque em ficar proficiente nesses assuntos. Você deve respirar os fundamentos dessa área e raciocinar com eles de forma bem natural.

No entanto, uma boa comunicação é essencial. Pratique bastante sua comunicação! Experimente ser entrevistado por algum amigo e vice-versa. Hoje em dia, são inúmeros os recursos disponíveis *online* e *offline* para se preparar para tal tipo de entrevista. Abaixo, selecionamos alguns desses principais recursos, dividindo-os por categoria.

Resolvendo problemas de entrevistas

Independente de quanta experiência você já tenha, resolver problemas semelhantes ou que já foram utilizados em entrevistas é um passo OBRIGATÓRIO para todos que querem concorrer a essas

vagas. A memória humana funciona de forma muito semelhante a dos computadores. Nós temos uma memória de curto e longo prazo (assim com memória RAM e SSDs em computadores). Quando você está constantemente estudando um assunto, você mantém esse assunto na memória de curto prazo e, portanto, é capaz de raciocinar e usá-lo de forma muito mais rápida e fluente. Alguns dos melhores sites que poderão ser utilizados:

- Leetcode (https://leetcode.com/) – Site com centenas de problemas no mesmo formato das entrevistas, dos quais muitos já foram utilizados em entrevistas reais.
- HackerRank (https://www.hackerrank.com/) – Site com vários desafios cobrindo todos os tópicos que podem ser abordados em uma entrevista.
- InterviewBit (https://www.interviewbit.com/) – Site que contém diversos problemas e tutoriais de assuntos que são explorados em entrevistas.

Se você perguntar para todas as pessoas que passaram em alguma entrevista de programação, especialmente para grandes empresas, eles muito provavelmente utilizaram algum dos *websites* acima.

Sites com competições online de programação

Competições *online* de programação são uma excelente forma de se manter afiado e aprender novos assuntos. Embora não seja essencial para uma entrevista de programação, é mais uma ferramenta para continuar praticando e motivado nos estudos. Estes websites são muito utilizados, principalmente por estudantes que participam de olimpíadas de informática tais como: OBI (Olimpíada Brasileira de Informática), IOI (International Olympiad in Informatics) e Maratona de Programação (http://maratona.ime.usp.br/).

- URI Online Judge (https://www.urionlinejudge.com.br) – Site com centenas de problemas de algoritmos e estruturas de dados. A melhor forma de evoluir suas habilidades nessa área é praticando com muitos problemas. Os problemas possuem discussões com outros usuários, contribuindo para um melhor aprendizado. Por ser uma plataforma muito utilizada no Brasil, em geral as pessoas são muito receptivas a ajudar uns aos outros.
- Codeforces (https://codeforces.com/) - Outro site internacional com milhares de problemas e comunidade ativa. Em geral os problemas possuem um nível de dificuldade ainda maior do que

de entrevistas. Contudo, essa pode ser mais uma fonte de treino para entrevistas de programação.

No entanto, não se assuste com o nível de dificuldade dos problemas nesses sites! Como já dito, eles possuem um espectro de assuntos e dificuldade muito além do exigido em uma entrevista. Essencialmente, funcionam mais como uma ferramenta de aprendizado - para se manter afiado nesses assuntos, do que um padrão do que seria a entrevista em si.

Simulando entrevistas

Uma das melhores formas de realmente se preparar é realizando entrevistas testes com amigos ou com outras pessoas que estejam com o mesmo objetivo. Essa é a cereja do bolo. Isso vai te dar a confiança necessária para mostrar, durante a entrevista, o que você tem de melhor.

Existem alguns recursos disponíveis *online*. Um deles é o Pramp (https://www.pramp.com/). Neste site, é possível praticar entrevistas tanto na posição de entrevistador, como na posição de entrevistado. Nele, você é conectado com outras pessoas - sejam amigos ou desconhecidos, com o objetivo de simular o ambiente real de uma entrevista.

Simular entrevistas com amigos é uma excelente opção, mesmo que você esteja na posição de entrevistador. Nessa colocação, é possível enxergar, mais nitidamente, o processo de raciocínio que uma outra pessoa tem e a forma como ela lida para resolver um problema. Isso, com certeza, vai contribuir para te deixar mais consciente quando na posição de entrevistado, dando-lhe um norte do que fazer e não fazer durante a entrevista.

Portanto, possui algum amigo que esteja se preparando para a entrevista? Tem algum amigo mais experiente? Então peça para que eles te entrevistem e, em troca, você os entrevista de volta.

Livros

- **Estruturas de Dados e Seus Algoritmos**. Jayme Luiz Szwarcfiter e Lilian Markenzon.
- ***Introduction to algorithms***. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein.
- ***Introduction to Algorithms****. Udi* Manber.
- ***The Algorithm Design Manual***. Steven S S. Skiena.

Todos esses livros te ensinam os fundamentos de algoritmos e estruturas de dados. Eles possuem conteúdo mais do que suficientes para passar em qualquer entrevista de programação. No

entanto, esteja ciente que você não precisa saber tudo! As entrevistas, em geral, buscam testar se o candidato sabe bem os fundamentos de algoritmos e estruturas de dados ao invés de saber algo muito mais avançado. Por exemplo, saber bem conceitos como árvore binária, árvore balanceada e busca binária são mais importantes do que saber como uma árvore de segmentos multidimensional ou uma *link-cut tree* funcionam.

No entanto, não se esqueça, a Google é a principal ferramenta para aprender algoritmos e estruturas de dados. Ouviu falar de algum conceito do qual não faz ideia do que significa? *Google it*! No final, uma fonte única de informação para aprender tudo não existe, e é sempre bom ler uma mesma informação de diferentes perspectivas até que você tenha um entendimento completo do assunto.

Design de sistemas

No dia a dia de trabalho, o engenheiro de *software* lida com sistemas complexos, os quais devem ser ao mesmo tempo escaláveis e resilientes. Sempre que confrontado com um novo desafio, criatividade e experiência são necessárias para resolver

esse problema. Por exemplo, imagine que você criou um aplicativo de troca de mensagens no estilo do Whatsapp, mas que, no início, só permite trocas de texto. Agora, você quer adicionar funcionalidades nesse sistema, tais como: distribuição de fotos e vídeos, criação de grupos e a possibilidade de acessar o aplicativo pelo desktop. Todas essas funcionalidades, por mais que pareçam simples, exigem muito conhecimento. Criar sistemas complexos como o Instagram, Snapchat, Google Search, Google Photos, entre outros, exige conhecimento de múltiplas áreas, as quais, em geral, não são 100% dominadas por uma única pessoa.

Em posições que exigem mais experiência, parte das expectativas de trabalho de um engenheiro é criar ou melhorar esses sistemas, que podem atender milhares ou até mesmo bilhões de pessoas. Eles possuem a responsabilidade de identificar oportunidades de crescimento para esses produtos, sendo eles novos ou já estabelecidos.

Uma entrevista em *design* de sistemas é focada em testar o conhecimento geral do candidato e sua habilidade de identificar oportunidades no sistema ou produto. Muitas vezes, esse tipo de entrevista é utilizado para selecionar candidatos com um maior nível de experiência.

Algumas das questões clássicas que são perguntadas em entrevistas desse tipo são:

- Faça o *design* de um encurtador de URLs
- Faça o *design* do Google maps
- Faça o *design* do Tinder
- Faça o *design* do Instagram
- Faça o *design* do Google photos

Esse tipo de questão é capaz de avaliar a habilidade do candidato em múltiplas áreas, tais como:

- Sistemas distribuídos
- Sistemas operacionais
- Redes de computadores
- Modelagem de sistemas
- Bancos de dados

Em geral, as questões desse tipo possuem múltiplas respostas corretas, às quais incluem diferente *trade offs*. Por exemplo, você sacrifica latência por resiliência do sistema. O objetivo final é entender a extensão do conhecimento do candidato e o quão profundamente ele é capaz de ir em cada assunto. Em geral, um

candidato mais qualificado precisa de quase nenhuma orientação e faz muitas perguntas para clarificar partes ambíguas do problema.

As questões que envolvem *design* de sistemas, além trazer problemas abertos com múltiplas soluções, possuem enunciados ambíguos e mal especificados. Portanto, é extremamente importante fazer perguntas para clarificar o que não está bem especificado. Lembre: muitas vezes o mesmo problema, quando resolvido para suportar 100 usuários, exige uma solução completamente diferente da pensada para suportar 1 milhão de usuários.

Por esse tipo de questão ser bastante ampla e aberta, a preparação não é exatamente linear e trivial. Detalharemos abaixo nossas principais recomendações para a preparação para esse tipo de entrevista.

A melhor forma de se preparar para uma entrevista de design de sistemas é com experiência real de múltiplos projetos. A vivência desenvolvendo projetos te dá a perspectiva grande e em profundidade naquele assunto. No entanto, experiência de muitos projetos é algo que demora anos para ser construída. Buscamos apresentar algumas forma de acelerar a absorção desse conhecimento sem a necessidade desses muitos anos de experiência. Hoje em dia na internet existem recursos extremamente

valiosos na preparação para esse tipo de entrevista. O mais famoso deles é o http://www.hiredintech.com/system-design. Esse site mostra todos os fundamentos em formas de vídeos e texto necessários para a preparação de forma bem didática e intuitiva. Outro site com dezenas de questao resolvidas é o https://leetcode.com/. Resolver múltiplas questões desse tipo é uma das melhores formas de se preparar.

Um recurso disponível que não podemos esquecer são os livros. Livros podem ser um complemento muito bom para os estudos para esse tipo de entrevista. No entanto, não se prenda totalmente a eles. Mesmo que eles tragam um conhecimento valioso na preparação para a entrevista, a preparação pode se tornar muito longa visto que eles possuem muito mais informação do que a necessária para passar na entrevista. Abaixo citamos alguns livros que são clássicos em alguns dos principais assuntos em uma entrevista de design.

- **Redes de computadores** - Redes de Computadores e a Internet: Uma Abordagem Top-Down. Jim Kurose, Keith Ross.
- **Sistemas operacionais** - Sistemas Operacionais Modernos. Andrew S. Tanenbaum

- **Bancos de dados** - Sistemas de Banco de Dados. Ramez Elmasri e Shamkant B. Navathe.
- **Sistemas distribuidos** - Designing Data-Intensive Applications: The Big Ideas Behind Reliable, Scalable, and Maintainable Systems. Martin Kleppmann.

Lembre-se: da mesma forma que na entrevista de algoritmos e programação é essencial praticar fazendo simulações *online* e com amigos, praticar para uma entrevista com uma quantidade de tempo limitada e sem nenhum recurso disponível para pesquisa te dá uma sensação única do cenário da entrevista. Esse pode ser um grande diferencial entre você e outros candidatos.

Comportamento e liderança

No mundo de hoje, apenas praticar para entrevistas tradicionais não é o bastante. Grande parte do seu trabalho envolverá interação com outras pessoas. Por isso, a entrevista comportamental vem ganhando bastante popularidade. Ela também é uma das formas de identificar líderes e pessoas que não possuem a cultura da empresa.

Não há segredo para uma boa preparação. É fundamental estudar a descrição da vaga e da empresa de seu interesse. Isso vai te ajudar a entender a cultura da empresa e, para eles, esse é um forte sinal que você tem grande interesse na vaga.

Não se surpreenda com questões que perguntam a suas maiores habilidades, defeitos e conquistas. Logo, não fique parado! Faça um lista de todos esses pontos antes mesmo da entrevista. Nessa lista, não foque apenas nos seus casos de sucesso. Procure também mostrar onde você já falhou e o que você aprendeu com isso. Essas empresas não estão em busca de alguém que nunca errou, mas sim de alguém que erra e aprende com os seus erros.

A entrevista

Entrevistas de emprego são importantes tanto para a empresa quanto para o candidato. É por meio delas que se pode analisar o nível técnico do entrevistado, suas habilidades de comunicação e se ele possui a cultura da empresa.

Esse processo é muito rígido, pois tem o intuito de garantir que a organização empresarial não contrate um mau funcionário. Por ser um procedimento complexo, rigoroso e formado por várias etapas, é óbvio que você, o entrevistado, deva se sentir muitíssimo estressado.

Embora nossa intenção seja a de te orientar e explicar como se preparar para a entrevista e evitar que você seja pego de surpresa com algum questionamento lançado, é importante lembrar que cada empresa de *software* avalia os seus candidatos diferentemente.

De agora em diante, iremos te passar as dicas essenciais para as entrevistas em geral, bem como o modo de se portar perante o entrevistador. Considerando que cada entrevista é um momento único, reconhecemos que questões específicas podem acontecer no desenrolar do processo. Contudo, ao absorver as dicas gerais, independentemente do formato que sua entrevista tiver, temos

certeza que você será bem sucedido, obtendo a aprovação almejada.

É preciso lembrar que ninguém é perfeito. Não se assuste se não souber algo ou se cometer algum erro. Isso é completamente normal! O resultado final depende da forma com a qual você lida com esses erros.

Nós temos anos de experiência, já tendo entrevistado milhares de candidatos. Por isso, juntos decidimos te mostrar algumas dicas de como ter uma ótima performance em uma entrevista!

O primeiro contato com o recrutador

Recrutadores analisam milhares de currículos diariamente. Dentre eles, os melhores são selecionados para uma conversa inicial. Nesta etapa, recrutador e candidato conversam por videoconferência.

Esta conversa pode levar de alguns minutos até uma hora. Inicialmente, será explicado o momento atual da empresa e quais os planos futuros. Depois disso, irão abordar os diferentes cargos que estão disponíveis e responder quaisquer dúvidas sobre a empresa.

A empresa quer ter contratar

Antes de mais nada, saiba que milhares de pessoas são rejeitadas diariamente e sequer conseguem uma entrevista. Assim, ao conseguir a oportunidade de ser entrevistado e, consequentemente, de aplicar as dicas dadas neste livro, você já é considerado um vencedor que está na frente de milhares de outros candidatos.

Entenda que o recrutador, apenas por ter te concedido a chance em questão, já tem o interesse em lhe contratar.

Empresas de tecnologia estão desesperadas por novos talentos. Portanto, enquanto candidato, você já é considerado um ponto fora da curva para elas. Então, não se acanhe e sinta-se à vontade para perguntar qual será o formato das entrevistas, quais as expectativas e até materiais de preparação. A área de computação é muito vasta e isso te ajudará a manter sua atenção no assunto correto.

A primeira entrevista (*screening*)

A primeira entrevista técnica, também conhecida como *screening interview*, é o seu primeiro contato com um engenheiro da empresa.

Nessa entrevista, você terá uma nova oportunidade de mostrar suas ambições e motivações. O entrevistador pode perguntar a razão de você aplicar para a empresa, por que você escolheu computação e por que está mudando de empresa.

Boas respostas mostram que o candidato não está só interessado em dinheiro, mas que acredita na visão da empresa e que um dos seus maiores motivadores é o interesse em aprender.

A maior parte dessa entrevista vai ser técnica. Nela, você terá que resolver um ou mais problemas de computação similares aos da sua preparação. Muitas vezes, essa entrevista é por videoconferência, utilizando um editor de texto online, como o http://collabedit.com/.

Lembre-se de explicar sua ideia antes de começar a programar. Comunicação é um ponto chave. Ela não se resume a apenas escrever o seu código. É importante mostrar o seu raciocínio de como testar seu código. Esse é um dos principais e mais comuns erros que muitos candidatos tendem a cometer já que apenas programam a sua solução e não mostram como o seu código pode ser testado. Grandes empresas prezam por códigos bem testados!

A intenção do entrevistador não é somente a de obter as respostas codificadas, mas também de traçar seu perfil pessoal e de

como você se relaciona com as pessoas no desenvolvimento do seu trabalho.

Mini-projeto

Nesse tipo de entrevista, você deve desenvolver um pequeno projeto, o qual em geral leva alguns dias de trabalho. Esse tipo de entrevista é exigido por muitas empresas, especialmente por *startups*. Em grandes empresas de tecnologia, esse tipo de entrevista costuma não existir. Essa fase é resolvida de casa, com o intuito de verificar sua organização e responsabilidade na resolução do que foi posto. Desta forma, podem ser analisados diversos fundamentos básicos de computação (e.g. programação orientada a objetos) e se você segue boas práticas de programação. Confira alguns exemplos de questões desse tipo:

- Crie um aplicativo de calculadora que possua as operações aritméticas básicas, tais como somar, subtrair, multiplicar e dividir.

  Com esse tipo de problema, a empresa é capaz de identificar algumas características do candidato, tais como:

- Motivação - o problema levará algumas horas para ser resolvido e portanto exigirá um certo esforço. O quão rápido o candidato consegue entregá-lo?
- Capacidade de pesquisa - é impossível dominar a computação por inteiro. Muitas vezes, surgem problemas novos, portanto, a capacidade do candidato para pesquisar informações é uma habilidade indispensável.

Normalmente, essa é a primeira ou segunda fase do processo de entrevistas, já que é capaz de eliminar a maior parte dos candidatos. Dentre eles, os que possuem experiência baixa com programação e/ou não estão motivados o suficiente para trabalhar naquela empresa.

Empresas grandes de tecnologia costumam não incluir essa etapa no processo seletivo devido a enorme quantidade de candidatos aplicando para as mesmas. Seria inviável avaliar o código de milhares de candidatos e tal avaliação poderia introduzir uma variedade muito grande ao processo. O processo nessas empresas precisa ser extremamente escalável e unificado.

A entrevista presencial (on-site interview)

Se você foi bem na primeira entrevista ou no mini-projeto, será convidado para a entrevista presencial.

As grandes empresas costumam realizar essa entrevista em seu *headquarters* e, conversando pessoalmente, conseguem identificar se você faz parte da cultura da empresa, pois o que menos almejam é firmar contrato com quem não se dará bem com os outros funcionários. Dessa forma, se o entrevistado não morar na mesma cidade da sede, elas irão pagar a passagem e hospedagem, mesmo que seja uma viagem internacional.

Feita essas considerações, vamos mostrar como um programador de elite se comporta durante a série de entrevistas presenciais que, somadas, podem durar mais de quatro horas! Mas não se assuste. Em muitas dessas entrevistas, vocês será tratado super bem, com direito a um tour pelo escritório, almoço e café!

Algoritmos

Ao longo da carreira em uma empresa, você vai precisar, no seu dia a dia de trabalho, criar novos sistemas e mudar códigos.

Neste tipo de entrevista, você irá resolver uma ou mais questões relacionadas a algoritmos. O objetivo é analisar o seu conhecimento de computação, o quão rápido é o seu desempenho e como você comunica suas ideias para os outros membros da equipe.

Seguem algumas dicas para utilizar durante a entrevista:

Resolva o problema certo

Durante toda essa seção de dicas, você vai ver entender que a palavra chave para uma boa entrevista é a comunicação. Imagine o seguinte exemplo: **Encontre o menor número em uma árvore de inteiros.**

Você pode resolver o problema percorrendo todos os nós da árvore e verificando qual o menor número, por exemplo:

```
def find_min_number(node):
    if node is None:
       return None
   min_number = node.number
   for child in node.children:
      result = find_min_number(child):
      if result is None and result < min_number:
          min_number = result
    return min_number
```

Quando, na verdade, a árvore poderia ser uma árvore binária de busca, isto é, todos os filhos à esquerda de um nó são menores do que o pai, e todos os filhos à direita são maiores! Dessa forma, o problema poderia ser reduzido a apenas encontrar o nó mais à esquerda da árvore.

```
def find_min_number(node):
    min_number = None
   while node is not None:
      min_number = node.number
      node = node.left_child
    return min_number
```

```
def find_min_number(node):
    min_number = None
   while node is not None:
      min_number = node.number
      node = node.left_child
    return min_number
```

Fica a dica: **busque entender o problema real antes de pensar na sua solução!**

Pense antes de resolver o problema

O problema, em algumas empresas, pode ser vago. Isto é, o entrevistador pode omitir alguns fatos com o intuito de testar se o candidato busca entender a fundo o problema antes de começar a

programar. Por exemplo, dado um problema para somar números, você pode escrever um código, como este:

```
int sum(int a, int b) {
  return a + b;
}
```

O código em si está certo, mas em nenhum momento você perguntou se **a** e **b** são inteiros ou se a soma deles cabe em um inteiro de 32 bits.

É importante se comunicar bem e esclarecer dúvidas como estas antes de iniciar a sua solução!

Comunique-se bastante

Mesmo que você pense bastante na ideia antes de iniciar a resolução do problema, existem decisões que precisam ser tomadas enquanto escreve seu código. Por isso, é super importante comunicá-las ao entrevistador, sempre mostrando que você sabe o porquê da decisão que está tomando.

Para o entrevistador, essa atitude vai mostrar que você é uma pessoa que sabe se comunicar bem em equipe!

Sempre teste o seu código

Você não será analisado apenas pelo seu conhecimento de algoritmos ou de escrever códigos. Outro comportamento que também é julgado é como você age após terminar o seu código. Algumas pessoas apenas terminam e passam a bola da vez para o entrevistado.

O programador de elite sugere diversas formas de testar o código e faz uma pequena simulação para um caso de teste simples. Lembre-se: **não se é esperado escrever testes unitários durante uma entrevista!** O importante é mostrar que você entende quais casos podem ser testados e como o seu programa pode falhar!

Design de sistemas

Essa entrevista é uma das formas utilizadas para descobrir qual cargo o candidato deve ocupar na empresa. Para estagiários ou recém-formados, ela é inexistente ou possui um peso muito pequeno.

Nela, você será desafiado a desenvolver o *design* de um sistema muitas vezes na escala de Google ou Microsoft, isto é, sistemas com bilhões de acessos diários.

É importante lembrar que poucas pessoas conseguem criar um sistema perfeito em apenas uma hora. Então, não se assuste se

depois da entrevista você pensar em formas melhores de resolver o problema.

O fundamental é mostrar como você pensa, por exemplo, que diferentes partes dos sistemas podem possuir diversas soluções. Portanto, sempre liste os prós e contras dessas possibilidades e explique porque acha que a sua escolha é correta.

Essa questão vai ser aberta. Um exemplo: **faça o design do Bing**! É um problema extremamente amplo que não possui resposta correta, sendo importante lembrar que poucas pessoas são capazes de construir um sistema desses em menos de uma hora. Por isso, aqui seguem mais algumas dicas:

Procure entender os requisitos do sistema

Pedir para fazer o *design* do Bing é muito vago, mas em nenhum momento comece a fazer o *design* de um sistema sem entender os requisitos de uma empresa. O Facebook não abriu as portas com 2 bilhões de usuários.

Listamos algumas perguntas que você pode fazer e qual o sentido delas:

- **Qual a quantidade de usuários?** A ideia é saber a escala do seu sistema; a parte interna de um produto com 10 usuários será diferente quando o produto tiver 10 bilhões

de usuários. Geralmente, sistemas com vários usuários precisam ser distribuídos em várias máquinas diferentes.

- **Quais os idiomas disponíveis?** Você pode achar que é preciso criar um sistema que funcione para o português, inglês ou japonês, mas o entrevistador pode esclarecer que eles só se importam com o espanhol.
- **Nos preocupamos com erros ortográficos?** É comum cometer erros quando pesquisamos em um site de buscas. O sistema que precisa ser desenvolvido para detectar esses tipos de erro é mais complexo. Por essa razão, antes de começar a desenvolvê-lo, pergunte ao entrevistador.

Embora você deva fazer várias perguntas, entrevistadores também dão valor a proatividade. Ao perguntar algo, você pode rapidamente sugerir uma resposta. Por exemplo: ao perguntar sobre erros ortográficos, você pode falar que acha melhor incluir essa *feature* agora, pois vai fugir um pouco do real objetivo do sistema.

Fica a dica: **quando possível, sempre pergunte e proponha soluções para a sua pergunta!**

Sempre explique os prós e contras

Uma grande empresa não está preocupada em encontrar um candidato que consiga fazer tudo sozinho. Caso seja o melhor engenheiro do mundo, a empresa vai lhe incentivar a ajudar outros engenheiros. Dessa forma, é de extrema importância mostrar que você se comunica bem.

Em diversas partes da resolução do problema, você vai se deparar com diversas ideias para resolver uma parte do sistema. Enumere-as no quadro negro e explique as diferenças. Faça sua escolha lembrando que o importante é sempre mostrar assertividade.

Comportamental

A entrevista comportamental é sobre você, sua motivação, seu currículo e o caminho trilhado para chegar onde chegou. O propósito desta entrevista é analisar se o candidato se adapta à cultura da empresa e está aplicando para o cargo correto.

Assim, esteja preparado para responder perguntas sobre o seu currículo, suas motivações e suas falhas. Para engenheiros mais experientes, as grandes empresas também procuram saber se você já teve conflitos no seu trabalho e como você os resolveu.

Exemplos típicos de perguntas incluem:

- Por que você escolheu ser engenheiro de *software*?

- Qual foi o seu maior erro na computação e o que você aprendeu com ele?
- Qual foi o maior conflito que você enfrentou no trabalho? Como você lidou com isso?
- Qual foi o projeto mais complexo com o qual você já trabalhou?
- Como você define os próximos projetos a serem desenvolvidos?
- Por que você deseja sair da sua empresa atual?
- Por que você escolheu a nossa empresa?

Não existe resposta correta para essas perguntas. O entrevistador está interessado em descobrir o que te motiva. O objetivo é compreender melhor a sua experiência e averiguar a veracidade do que está escrito no seu currículo.

Essa entrevista é a melhor oportunidade que você tem para descrever o que você já fez no passado. Lembre-se: tendo muita ou nenhuma experiência de emprego, todo mundo já participou de atividades semelhantes às que ocorrem em um ambiente de trabalho. Trabalhos escolares em grupo (e individual), competições esportivas, apresentações, trabalho voluntário são exemplos de situações extremamente semelhantes às que ocorrem em um ambiente de trabalho.

Não esqueça que o termo liderança não se refere apenas a liderar outras pessoas, mas também liderar a si mesmo, especialmente no que diz respeito à forma de ajustar a direção da sua carreira; o quão independente você é para realizar as suas tarefas; e como você define as suas prioridades.

Como exemplo, dispomos o trecho de uma entrevista desse tipo:

- **Entrevistador**: “Bom dia João! Como está? Deseja beber alguma coisa ou ir ao banheiro?”
- **Candidato**: “Muito obrigado, mas estou bem.”
- **Entrevistador**: “Então vamos começar a entrevista, ok? Qual o motivo de você querer sair da sua empresa atual?”
- **Candidato**: “Os projetos da minha empresa atual são muito relacionados a *frontend*, enquanto que eu quero direcionar minha carreira mais para o lado do *backend*. Além disso, às vezes eu sinto que as oportunidades de crescimento são muito limitadas.”
- **Entrevistador**: “Entendi. E qual seria o motivo de mudar para a nossa empresa? Como podemos ajudar você a alcançar esses objetivos?”

- **Candidato**: “Pelos meus amigos que trabalham com vocês e pelas notícias que confiro, percebo que vocês possuem uma estrutura bem avançada de sistemas distribuídos”
- **Entrevistador**: “Qual foi o projeto mais desafiador com o qual você trabalhou?”
- **Candidato**: “O maior projeto no qual trabalhei foi para criar todo o *frontend* de um *e-commerce* com mais de 10 mil usuários. Nós tivemos que usar múltiplas tecnologias, tais como React e Node e coordenar constantemente com o time de *backend* nos requerimentos para fazer o sistema funcionar como um todo. Além disso, inúmeras interações foram feitas com gerente de produto, *designers* e *UX researchers* para descobrir as melhores *features*, facilitar a usabilidade e descobrir o que os usuários eventualmente queriam.”
- **Entrevistador**: “Interessante! E quantas pessoas trabalharam nesse projeto no total? Você era a única pessoa trabalhando no *frontend*?”
- **Candidato**: ”Nós tínhamos 1 *designer*, 1 *UX researcher*, 1 gerente de produto, 2 pessoas no *backend* e 2 engenheiros no *frontend* (contando comigo)”

- **Entrevistador**: “Como você dividia as tarefas com o outro engenheiro no *frontend*?”
- **Candidato**: “Eu era o engenheiro mais sênior na parte do *frontend*, então eu organizava os requerimentos de todas as outras partes (*designers*, produto, ….) e separava isso em múltiplas tarefas e projetos menores a serem feitos. Eu delegava alguns desses projetos menores para o outro engenheiro no *frontend* e com isso nós conseguimos obter o máximo de produtividade no time com um todo”
- **Entrevistador**: “Qual o maior conflito com o qual você teve que lidar no trabalho?”
- **Candidato**: “O outro engenheiro, com quem eu trabalhava no *frontend*, estava constantemente discordando da forma que eu propunha para resolvermos uma certa parte do projeto.”
- **Entrevistador**: “E como você lidou com isso?”
- **Candidato**: “Marquei múltiplas reuniões com ele para que pudéssemos chegar na melhor solução possível. Afinal, eu poderia estar errado também. Após múltiplas reuniões, nós conseguimos, de fato, chegar em uma evolução da minha solução original. Então, eu convoquei uma reunião

com todo o time para que pudéssemos discutir a melhor abordagem possível e, assim, nós conseguimos eventualmente chegar a um consenso e resolver de vez o problema.”

- **Entrevistador**: “O que você mais aprendeu com esse conflito?”
- **Candidato**: “Aprendi que mesmo que você esteja certo, ouvir diferentes perspectivas pode ser muito útil e acabar levando a uma solução final de maior qualidade”
- **Entrevistador**: “Se eu perguntasse para o seu gerente sobre você, o que ele diria?”
- …
- **Entrevistador**: “Você tem alguma pergunta para mim?”
- **Candidato**: “Com o que você mais gosta de trabalhar aqui? Você se vê trabalhando aqui por muito mais tempo? Qual um ponto negativo de trabalhar nessa empresa?”
- …

Novamente, perceba que todas essas questões são muito boas para ver a extensão das atividades com as quais o candidato já atuou. O entrevistador também avalia o nível de liderança do

candidato, como ele lida com múltiplas situações e se ele eventualmente seria um bom funcionário para a empresa.

Por ser uma entrevista com formato bastante aberto e informal, muitos candidatos acabam por subestimá-la e não se preparar adequadamente, o que pode se tornar um grande erro.

Como já dito ao longo deste livro, é de extrema importância se preparar corretamente para TODAS as entrevistas.

Encontrando *bugs*

Ao entrar em uma nova empresa, muito provavelmente você irá trabalhar com um sistema que nunca utilizou. No mundo da computação, é normal acontecerem problemas que não foram previstos anteriormente.

Parte do dia-a-dia de um engenheiro de *software* é arrumar esses problemas em tempo hábil. Imagine que a empresa pode perder milhões de reais por hora se seu sistema ficar fora do ar. É exatamente por isso que algumas empresas adotam uma entrevista específica para testar o raciocínio do candidato. O objetivo é que o candidato consiga encontrar problemas, mesmo em um código que não lhe é familiar.

Uma dica essencial para essa situação é: pensar que o entrevistador é um membro do seu time e que entende o sistema. Dessa forma, não hesite em formular perguntas características que envolvam o próprio sistema. Ou seja, não fique calado!

Não existe nenhum segredo. A dica é sempre se comunicar bastante! Na vida real, você não será responsável por resolver esses problemas sozinho. Então, use o entrevistador a seu favor.

Os principais erros

Não fazer perguntas

Repetimos essa dica porque ela é importante. Muitos candidatos, durante a entrevista, começam a tentar resolver o problema muito antes de entendê-lo completamente. Uma das principais características avaliadas durante a entrevista é a capacidade do candidato suceder em problemas ambíguos. Muitas vezes, o entrevistador, de forma proposital, não especifica de forma completa o problema, esperando que o candidato faça perguntas, clarificando-o.

Perguntas básicas como: “os dados são apenas números inteiros?”; “esse caso é possível?”; “qual o número de pontos de

entrada?” são essenciais para mostrar que você está buscando uma solução correta para o problema específico que o entrevistador deseja.

Não explicar as suas ideias

Muito candidatos simplesmente têm uma ideia de como resolver um problema e começam a programá-lo sem ao menos explicá-la ao entrevistador. Tal conduta possui inúmeros problemas e sugere que o candidato não trabalha bem em equipe.

Primeiro: a ideia que você pensou pode estar errada e, se isso for verdade, o entrevistador fará perguntas para tentar mostrar o erro ou o que você precisa melhorar. Segundo: um dos principais pontos avaliados durante a entrevista é a capacidade de raciocínio e o quão bem você consegue sintetizar as suas ideias.

No dia a dia de trabalho, com outras pessoas, você está sendo constantemente exposto a problemas e deve discutir e apresentar suas ideias para elas. A entrevista busca simular esse tipo de interação e esse é um dos principais pontos avaliados.

Não pesquisar sobre a empresa

Em algum momento durante a entrevista, surgirá a oportunidade de perguntar aos entrevistadores sobre a empresa e o trabalho como um todo. Por isso, é essencial que você tenha feito o seu dever de casa e pesquisado pelo menos por algumas horas sobre a empresa. Demonstrar, durante a entrevista, que você efetuou pesquisas sobre a empresa e preparou boa perguntas o ambiente e rotina de trabalho demonstra um grau de interesse no cargo intencionado. Seguem alguns exemplos de perguntas:

- Quais são os maiores competidores dessa empresa? Como ele se posiciona com relação a eles?
- Quais oportunidades de crescimento de carreira que a empresa oferece a seus funcionários?
- Os objetivos da empresa se alinham com os objetivos do funcionário?

Portanto, faça uma pesquisa, mesmo que básica, sobre a empresa e pense em algumas perguntas para fazer na hora da entrevista.

Cometer alguma *red flag*

O que é uma *red flag*? É quando você faz algo que torna todos os pontos positivos durante a entrevista irrelevantes. Lembre-se: por mais que você esteja aplicando para uma empresa com ambiente flexível, você ainda deve ser profissional. Exemplos de *red flag* são:

- Assédio sexual
- Racismo
- Ser rude
- Não respeitar diferentes pessoas e culturas - sejam elas de diferentes religiões, opiniões políticas, classes sociais, etc
- Se atrasar para a entrevista
- Palavrões

No entanto, não se preocupe muito com isso. Esses tipos de assuntos não serão abordados em uma entrevista normal e são extremamente raros os casos em que candidatos são eliminados por esses motivos. Entretanto, esse é mais um ponto a ser levado em conta na hora de fazer sua entrevista.

Pós entrevista

Como lidar com a falha

Falhas apenas plantam sementes para o sucesso do futuro. É importante lembrar que a falha só realmente existe no momento em que paramos de tentar. Steve Jobs foi demitido da própria empresa. Isso o fez fundar a NeXT e a Pixar, o levando, eventualmente, a ser novamente o CEO da Apple. Abraham Lincoln, um dos maiores presidentes dos Estados Unidos, uma vez disse: “A minha maior preocupação não é saber se você falhou, mas sim se você se contenta com a falha”.

Embora você tenha se preparado bastante, diversos fatores, incluindo alguns que estão fora do seu controle, podem influenciar na sua contratação. Lembre-se também que nenhum processo de seleção é perfeito. A grande maioria das empresas busca minimizar o número de falso positivos. Então, é possível que mesmo que você seja competente o suficiente, o processo de seleção não conseguiu extrair o máximo na sua entrevista.

Lidar com a falha é a parte mais importante enquanto estiver prestando entrevistas para múltiplas empresas. Em seu processo de seleção, todas elas buscam minimizar o número de falsos positivos

(quando um candidato com um nível abaixo do esperado é contratado), portanto, mesmo que você já esteja preparado para passar, você pode não estar em um dia bom ou cometer algum erro por falta de atenção e isso pode acabar fazendo com que você não performe bem naquele dia. Ainda assim, isso não quer dizer que você não está preparado.

Uma das características mais marcantes de pessoas altamente bem sucedidas é a persistência. O sucesso não vem da noite para o dia. Inúmeros são os exemplos de pessoas bem sucedidas nas quais o sucesso demorou a aparecer. Como dissemos antes, Steve Jobs foi demitido da própria empresa até conseguir demonstrar de vez o seu valor. Jack Ma foi rejeitado por inúmeras empresas até finalmente conseguir suceder com o Alibaba. Os próprios autores deste livro já falharam algumas vezes em entrevistas. Entretanto, a falha nada mais é do que apenas um aprendizado para o próximo passo. Você deve sempre tirar o máximo das suas supostas ‘falhas’ e encará-las de forma positiva. Perguntar-se por que você não foi aceito é muito importante. O que eu fiz de errado nessa entrevista? Preciso melhorar minha capacidade de resolver problemas? Preciso melhorar meu conhecimentos de

algoritmos? Devo estudar mais teorias dos grafos? Mais estruturas de dados? Preciso melhorar minhas habilidades de comunicação? Não desistir é fundamental para se alcançar seu objetivo. Afinal, se fosse algo fácil, não teria tanto valor e muitas pessoas não estariam buscando isso, certo?

Uma coisa é certa: quanto mais você se prepara, maiores são as chances de você passar e o seu ‘dia ruim’ ser mais do que o suficiente para passar. A evolução não acontece da noite para o dia, mas com a direção correta, o sucesso é garantido.

Escolhendo a empresa certa

Uma vez que você tenha oferta de múltiplas empresas, escolher a empresa certa definirá seu sucesso nos próximos anos. Muitos fatores devem ser levados em conta para escolher a empresa na qual você pretende trabalhar. Alguns deles são:

- **Salário -** Embora salários maiores sejam mais atraentes, é importante levar em consideração o aprendizado. Afinal, nada melhor do que ser pago para aprender! É importante também lembrar que grande parte da sua renda em empresas maiores será em ações.

- **Perspectiva de crescimento da empresa** - Quanto maior for a perspectiva de crescimento da empresa, melhor. Pense só: se você entrar em uma empresa que aumenta 10 vezes de tamanho em alguns anos, você irá se tornar rapidamente um dos funcionários mais antigos na empresa. Isso fará com que você tenha mais chances de ocupar uma posição de destaque. Em empresas com baixa taxa de crescimento, costuma ser mais difícil de se obter tais posições.
- **Cultura da empresa** - A grande maioria das empresas de tecnologia possuem horários de trabalho super flexíveis, incentivos para trabalhar de casa e não exigem vestimenta formal. Lembre-se também de analisar a cultura relacionada aos projetos. Algumas empresas prezam por um desenvolvimento mais sólido e lento, enquanto outras preferem iteração rápida, mesmo que o código não seja perfeito.
- **Produto e missão** - O seu alinhamento com o produto e a missão da empresa será fundamental para o seu desempenho dentro da empresa. Você estará muito mais motivado para trabalhar com algo que segue a sua visão do que o contrário. Isso te ajudará a definir a sua progressão dentro da empresa.

Portanto, é um aspecto fundamental no longo prazo em sua carreira.

Você deve botar na balança cada um dos aspectos citados acima e ver o que te faz mais feliz. Diferente pessoas possuem diferentes pesos. Mas uma coisa é certa: especialmente no início da sua carreira, você deve sempre ter como foco a maximização do seu aprendizado. No longo prazo isso que será o maior fator responsável pelo seu sucesso. Imagine que você seja capaz de melhorar todos os dias seu conhecimento em 0.5%. Em 1 ano você terá mais de 6 vezes o conhecimento inicial, em 5 anos estará quase 9 mil vezes mais sábio e em 10 anos estará 80 milhões de vezes mais sábio! Por isso é essencial focar em você acima de tudo.

Conclusão

Parabéns! Ao ler este livro, você já deu um grande passo para se tornar um programador de elite. Lembre-se que não existe um único caminho para se conseguir o emprego dos sonhos, independente do qual você escolha, não vai ser fácil, mas você não pode desistir!

Existem inúmeros casos de pessoas que tentaram um emprego em empresas como Microsoft e Google, falharam na primeira entrevista mas não desistiram e posteriormente foram contratadas.

O caminho que mostramos torna muito mais fácil e rápido alcançar o seu ápice na programação. Apesar de já termos colocado aqui vários conteúdos que te ajudarão a aprimorar suas habilidades, existem muito mais materiais disponíveis de forma gratuita online. Uma característica forte de grandes engenheiros é ser autodidata, então não se limite a este livro e continue buscando conhecimento em outras plataformas.

É importante lembrar que apenas saber programar não é o bastante. Pratique muito para a entrevista. Não se esqueça de aprimorar sua comunicação e treinar com amigos, como

entrevistador e entrevistado. Esses dois papéis te darão perspectivas diferentes de como o processo funciona. Nunca se esqueça que você sempre irá trabalhar com outras pessoas e saber se comunicar é essencial para isso. Lembre-se de sempre explicar as suas ideias.

Não importa o resultado da entrevista, procure sempre aprimorar seus conhecimentos. Com isso, você irá progredir mais rapidamente na empresa e, caso, não tenha conseguido a sua oferta estará pronto para a próxima tentativa.

Com este livro, te demos todas as dicas que gostaríamos de ter ouvido antes de iniciarmos a nossa carreira. E não paramos por aqui! Também criamos uma conta no instagram, @programadordeelite, com o intuito de responder às dúvidas mais frequentes e te ajudar cada vez mais a conseguir se tornar um programador de elite!