Capítulo

1 POR QUE PROGRAMAR?

A programação permite automatizar tarefas que são complexas e massantes para o ser humano, de uma forma extremamente confiável e veloz, a programação pode mudar sua maneira pensar, comunicar e até mesmo o modo como encara os desafios do dia-a-dia, a palavra chave para isso é a objetividade. Isso mesmo, a programação pode te ensinar a ser objetivo e prático.

É fato que a programação potencializa a capacidade de resolver problemas de uma maneira surpreendente, e atualmente não se limita a apenas cientistas da computação... a programção pode ser útil para resolver QUALQUER problema. Os tópicos a seguir apresentam uma série de argumentos para demonstrar a importância da programação computacional.

Automatizar tarefas: Executar sequência, muitas vezes longa e complexa, de instruções via comando único.

Fator velocidade: Computador é capaz de executar cálculos com elevada precisão e rapidez, superando, e muito, a capacidade do ser humano. Atualmente, o fator velocidade é essencial, pois a cada dia são coletados volumes de dados cada vez maiores, o que torna praticamente impossível a derivação de informações sem o auxílio do computador, além disso, as informações devem ser geradas e disseminadas em um intervalo de tempo cada vez menor. Sem a comunicação

em tempo hábil, tal informação pode perder o sentido e ser pouco útil para os processos de tomada de decisão.

Fator erro: Minimiza a quantidade de trabalhos repetitivos e monótonos, diminuindo as chances de ocorrer erros humanos devido ao cansaço e redução gradativa da concentração.

Pensamento criativo vs rotineiro: A programação possibilita ao profissional dedicar mais tempo para o pensamento criativo, que demanda capacidade de imaginação e percepção. Já o pensamento rotineiro requer pouco talento, a não ser aquele de seguir instruções corretamente.

Independência de *softwares* já prontos: Todos os *softwares* são concebidos para resolver um conjunto de problemas que surgem com maior frequência no dia-a-dia dos usuários, pelo menos de acordo com as observações, impressões e pesquisa do desenvolvedor frente ao seu nicho de mercado. É ilusório acreditar que existe *softwares* desenvolvidos para resolver todos os seus problemas. É muito comum se deparar com problemas sem soluções implementadas em *softwares*, em alguns casos o trabalho é interrompido ou então a estratégia de resolução deve ser alterada. Ou altera-se o algoritmo ou então utiliza-se procedimentos manuais.

Documentar metodologias: A sequência de instruções é armazenada em um documento de forma lógica e intuitiva, podendo ser consultada e facilmente entendida, mesmo após anos do desenvolvimento. Especialmente se junto com os

códigos alguns comentários, para descrever as etapas mais críticas, forem adicionados.

Não se sentir um peixe fora d'água: Dentro do seu ambiente de trabalho, ser capaz de identificar problemas que são passíveis de serem resolvidos via programação e resolver por conta própria ou então buscar equipe (internamente ou externamente) capacitada para resolver tal problema.

Potencializa a capacidade de resolver problemas: Essa é uma opinião compartilhada por muitos programadores. A prática de programação desenvolve a capacidade de pensar de forma sistemática e objetiva, o que facilita o processo de resolução de problemas variados, inclusive da vida pessoal. Aprender a programar permite organizar melhor as ideias, focar no que é mais importante e crítico. De maneira geral o conhecimento de programação permite ver o mundo de outra forma. Em muitos países, a prática de programação é incentivada, e faz parte do plano pedagógico de muitas escolas. O incentivo começa com as crianças, mesmo em idades iniciais. A programação favorece a aprendizagem, sobretudo em disciplinas relacionados às ciências exatas. A seguir é apresentada uma frase dita pelo Steve Jobs: "Todas as pessoas deveriam aprender a programar um computador, pois isso ensina você a pensar".

Segurança para resolver problemas: Alguns dos problemas que enfrentamos no dia a dia não possuem solução implementada disponível, diante desse fato, o caminho a ser seguido seria alterar a metodologia de resolução, o que nem

sempre é possível ou então buscar profissional qualificado no mercado, o que pode ser difícil ou então muito oneroso. Ainda existe o cenário mais pessimista, que é simplesmente a constatação de que o problema não pode ser resolvido com a atual infraestrutura em mãos. O fato principal aqui é que com um certo conhecimento de programação dificilmente será intimidado pelos problemas apresentados.

Menos estresse: Com *softwares* de aponte-e-clique momentos de estresse podem ser mais frequente, sobretudo porque esse tipo de *software* é muito mais suscetível a *bugs* do que uma linguagem de programação .Isso porque com esses *softwares*, além de se preocupar com a implementação do algoritmo que efetivamente resolve o problema é necessário desenvolver uma interface gráfica para o usuário, algo muito complexo de ser feito. *Bugs* são coisas realmente estressantes, às vezes a ferramenta não entrega o que promete ou então causa a interrupção do funcionamento do *software*. Aqui cabe um parêntese, de acordo com minha experiência, na maioria das vezes que esses problemas acontecem é culpa do usuário. Isso porque muitas vezes o usuário não sabe “pedir”, isto é, não executa a ferramenta correta ou então não executa a correta como se deveria (como instruído no manual). Ok, mas o desenvolvedor deveria prever isso e simplesmente emitir uma mensagem de erro, com uma instrução para executar a função corretamente. Geralmente os desenvolvedores fazem isso, e *softwares* já maduros fazem isso com maestria, inclusive com planejamento de interfaces que minimizam as chances de o usuário fazer algo errado. Mas imaginar que um *software* irá

controlar (cercar) todas as chances de nós cometermos erros é completamente ilusório. Além disso, geralmente quanto maior é o controle mais engessado é o *software*.

Maior satisfação pessoal: Nada melhor para autoestima do que resolver um problema programação computacional. Você se sente parte da solução, com um sentimento que sua participação foi vital para resolver o problema.

Possibilidade de estar na fronteira do conhecimento: É muito comum que uma pesquisa inovadora necessite de determinado tipo específico análise, mas pode ser que simplesmente não existe metodologia publicada sobre a análise e muito menos implementações disponíveis, nesses casos a programação pode ser útil, pois o atraso na publicação pode significar a perda do caráter inovador e até mesmo de uma eventual patente.

Confere capacidade de identificar e utilizar códigos prontos disponíveis: Existe quantidade muito grande de códigos disponíveis para resolver uma infinidade de problemas. Porções de códigos são muito mais simples de serem desenvolvidos do que softwares com interface gráfica com o usuário. Seja um reciclador de códigos, evite reinventar a roda!

Aprender sobre temas complexos: Alguns temas, especificamente relacionados à matemática, são indigestos para boa parte dos estudantes. Essa dificuldade está associada a diversos fatores, mas a ausência de aplicações práticas com explicação do procedimento de forma detalhada, pode ser

apontando como um dos principais. Com a programação é possível realizar experimentos práticos de forma simples e rápida, além disso, com a programação é possível explorar os conceitos mais básicos da matemática, que apesar de básicos os estudantes têm dificuldades de entender com aulas puramente teóricas, isso devido à baixa capacidade de abstração.

Aprender a utilizar a matemática de forma "correta": A resolução de problema matemático requer 4 etapas básicas: *(1) Identificar as questões corretas / definir o problema:* requer proatividade e conhecimento técnico sobre o tema de estudo; (2) F*ormular o problema:* converter um problema do mundo real em uma formulação matemática, e se possível já na forma de código para ser resolvido por um computador. É aqui que o conhecimento de programação é importante e útil; (3) *Encontrar a solução:* tarefa executada por um computador, seria basicamente fazer contas; (4) *Avaliação da solução:* converter um problema matemático em uma solução passível de ser executada no mundo real. Também é a etapa em que a decisão é tomada.

Se preparar para o futuro: Na verdade se preparar para o agora! A demanda por programadores já é elevada e a tendência é aumentar cada vez mais, atualmente muitas empresas demandam profissional com o domínio de programação além da formação técnica convencional, tal como agronomia, engenharia civil, engenharia florestal e biologia. Há previsões mais extremistas de que a programação será tão

importante e necessária quanto disciplinas básicas, tais como biologia e física, e até mesmo uma atividade tão básica quanto dirigir um carro. Apesar dessas previsões serem um tanto exageradas o que não resta dúvida é que a programação está se tornando cada vez mais relevante e seu domínio pode se tornar a alavanca que você precisa para alcançar colocações melhores no mercado.

Capítulo

2 POR QUE O R?

O R é um software ou linguagem de programação? R é um ambiente completo de desenvolvimento: é um ambiente integrado de funções para manipulação de dados, cálculos e gráficos; além de um conjunto completo de estruturas de controle (condicional e repetição).

É Gratuito: o R pode ser copiado e distribuído entre os usuários, bem como pode ser instalado em diversos computadores livremente, promovendo uma economia para empresas e pessoas físicas, devido ao não pagamento de taxas de licenças que são cobradas por outros softwares pagos, que além de serem altas são bem restritivas.

Facilidade de uso: Apesar do R ser executado a partir de comandos, não é necessário ser um programador para aproveitar dos benefícios oferecidos, pois uma grande quantidade de rotinas já estão implementadas, se o usuário não encontrar determinada função que execute a análise requerida, esta pode ser criada com certa facilidade, pelo menos comparativamente com outras linguagens de programação.

Facilidade de criar novos procedimentos: o R possui uma linguagem de programação bem desenvolvida, simples e efetiva, que inclui condicionais, estruturas de repetição, funções recursivas definidas pelo usuário, e facilidades para entrada e saída de dados. O R ainda suporta a vetorização, o que permite

executar procedimentos repetitivos (loops) sem a necessidade de definição explícita.

Compartilhamento: Facilidade e rapidez de troca de informações e conhecimentos, pois análises complexas podem ser realizadas com poucas linhas de comando, que na verdade são essencialmente blocos de textos. Esses textos poderão ser enviados ou recebidos através de um simples e-mail ou mensagem de WhatsApp, ou então acessadas em salas virtuais de grupos de ajuda por pesquisa em sites de busca, como o Google. Em contrapartida, para compartilhar análise análoga em um *software* com interface de aponte-e-clique o gasto de energia será muito maior, exigindo uma série de capturas de tela (*print screen*) e setinhas indicando o significado de cada elemento da janela do *software*.

Executar análises complexas: Possibilidade de executar análises complexas de forma simples nas mais variadas áreas do conhecimento. Abaixo é apresentado o ajuste de modelos utilizando duas estratégias completamente distintas, com elevada variação em termos de complexidade, porém a forma de declarar esses modelos no R são bem semelhantes. Primeiro o ajuste de uma regressão linear múltipla, utilizando a função **`lm`** (*Fitting Linear Models*) da biblioteca **`stats`**; em seguida o ajuste via redes neurais artificiais, utilizando a função **`nnet`** (*Fit Neural Networks*) da biblioteca de mesmo nome.

```
# Dados: y em função de x1 e x2
y <- c(0.21, 0.25, 0.1, 0.79, 0.55, 0.39, 0.71)
x1 <- c(0.51, 0.66, 0.9, 0.05, 0.42, 0.7, 0.33)
x2 <- c(0.1, 0.23, 0.15, 0.9, 0.65, 0.44, 0.81)

# Regressão Linear Múltipla
linear_multipla <- lm(y ~ x1 + x2)

# Rede Neural Artificial
library(nnet) # Carregar biblioteca nnet
# 2 neurônios na camada oculta
rede_neural <- nnet(y ~ x1 + x2, size = 2)
```

Código fonte aberto: Permite o acesso à rotina utilizada em determinada análise, possibilitando a alteração do código de acordo com necessidades específicas do usuário e possibilita aprender o princípio de funcionamento de determinado procedimento via exame do código. Além disso, as falhas podem ser detectadas com maior facilidade, e as correções e atualizações poderão ser disponibilizadas em questões de dias pelo grupo que gerencia o R (*Core Development Team*).

Quantidade de extensões: O R pode ser estendido via funções, scripts e principalmente via criação de novas bibliotecas (ou pacotes). O R possui uma infinidade de bibliotecas para as mais variadas áreas do conhecimento (ver *CRAN Task Views*).

Capacidade gráfica: É possível construir gráficos variados, robustos e com elevada qualidade tipográfica de forma simples e rápida. O R é recomendado para confecção final de figuras

para livros e materiais didáticos devido à sua qualidade tipográfica.

Multiplataforma: R é disponível para muitas plataformas, incluindo Unix, Linux, Windows, Macintosh.

Disponibilidade de materiais de apoio: existência de inúmeros manuais, tutoriais, cadernos didáticos, apostilas e livros destinados a ensinar o uso do R.

O RStudio: plataforma de desenvolvimento madura, amplamente utilizada pela comunidade e com um excelente editor de texto, que conta com uma série de funcionalidades, tais como identificação de erros de sintaxe; complemento de funções e objetos; coloração diferenciada de objetos e estruturas de controle; atalhos de teclado úteis, como o de executar códigos (ctrl + Enter) e os de alterar do editor para o console (ctrl + 2) e do console para o editor (ctrl + 1); comando para endentação automática; além de outras funcionalidades, como janelas específicas para plotar figuras, acessar arquivos e bases de dados, consultar os documentos de ajuda das funções.

Muitas possibilidades de fazer a mesma coisa: positivo, mas pode ser negativo, sobretudo para iniciantes; dica: identificar os pacotes/autores de confiança, evitar usar um novo pacote para executar determinados procedimentos que podem ser executados com a combinação de poucas funções de um pacote básico.

Curva de aprendizagem íngreme (negativo) vs flexibilidade e capacidade de resolver problemas - mais flexível do que ambientes aponte-e-clique. Mesmo assim, aprender R é muito mais fácil do que uma série de outras linguagens de programação.

Ausência de assistência técnica (negativo)**:** o grupo que gerencia o R não se responsabiliza pelos resultados retornados pela execução das rotinas disponibilizadas, além de não ofertar suporte técnico formalmente vs comunidade ativa de usuários e suporte técnico via contratação de terceiros.