

*2^a EDIÇÃO
Revisada e ampliada*

INTRODUÇÃO À PROGRAMAÇÃO COM
PYTHON

Algoritmos
e lógica de programação
para iniciantes

novatec

Nilo Ney Coutinho Menezes

Introdução à Programação com Python

Algoritmos e lógica de programação para iniciantes

Nilo Ney Coutinho Menezes

Copyright © 2010, 2014 da Novatec Editora Ltda.

Todos os direitos reservados e protegidos pela Lei 9610 de 19/02/1998.

É proibida a reprodução desta obra, mesmo parcial, por qualquer processo, sem prévia autorização, por escrito, do autor e da Editora.

Editor: Rubens Prates

Revisão gramatical: Adriana Bernardino

Editoração eletrônica: Camila Kuwabata

Capa: Victor Bittow

ISBN: 978-85-7522-408-3

Histórico de impressões:

Junho/2014	Segunda edição
Agosto/2013	Terceira reimpressão
Novembro/2012	Segunda reimpressão
Outubro/2011	Primeira reimpressão
Novembro/2010	Primeira edição (ISBN: 978-85-7522-250-8)

Novatec Editora Ltda.
Rua Luís Antônio dos Santos 110
02460-000 – São Paulo, SP – Brasil
Tel.: +55 11 2959-6529
E-mail: novatec@novatec.com.br
Site: novatec.com.br
Twitter: twitter.com/novateceditora
Facebook: facebook.com/novatec
LinkedIn: linkedin.com/in/novatec
MP20140623

Impressão e acabamento: Mark Press Brasil

A minha esposa, Chris; e aos meus filhos, Igor, Hanna e Iris.

Sumário

Agradecimentos.....	13
Prefácio da segunda edição	14
Prefácio da primeira edição	15
Introdução.....	16
Capítulo 1 ■ Motivação	19
1.1 Você quer aprender a programar?	19
1.2 Como está seu nível de paciência?	20
1.3 Quanto tempo você pretende estudar?	21
1.4 Programar para quê?	21
14.1 Escrever páginas web.....	21
14.2 Acertar seu relógio.....	22
14.3 Aprender a usar mapas.....	22
14.4 Mostrar para seus amigos que você sabe programar	22
14.5 Parecer estranho	22
14.6 Entender melhor como seu computador funciona.....	23
14.7 Cozinhar	23
14.8 Salvar o mundo	24
14.9 Software livre	24
1.5 Por que Python?	24
Capítulo 2 ■ Preparando o ambiente	27
2.1 Instalação do Python	27
2.1.1 Windows.....	28
2.1.2 Linux	34
2.1.3 Mac OS X.....	34
2.2 Usando o interpretador	34

2.3 Editando arquivos	36	6.6 Usando listas como filas	109
2.4 Cuidados ao digitar seus programas.....	40	6.7 Uso de listas como pilhas.....	110
2.5 Os primeiros programas	41	6.8 Pesquisa	112
2.6 Conceitos de variáveis e atribuição	44	6.9 Usando for	114
Capítulo 3 ■ Variáveis e entrada de dados	49	6.10 Range	115
3.1 Nomes de variáveis.....	49	6.11 Enumerate.....	117
3.2 Variáveis numéricas	50	6.12 Operações com listas	117
3.2.1 Representação de valores numéricos	51	6.13 Aplicações	118
3.3 Variáveis do tipo Lógico	53	6.14 Listas com strings.....	120
3.3.1 Operadores relacionais	53	6.15 Listas dentro de listas	120
3.3.2 Operadores lógicos	55	6.16 Ordenação	123
3.4 Variáveis string	60	6.17 Dicionários	127
3.4.1 Operações com strings	62	6.18 Dicionários com listas	131
3.5 Sequências e tempo	67	6.19 Tuplas	133
3.6 Rastreamento	68		
3.7 Entrada de dados.....	69		
3.7.1 Conversão da entrada de dados.....	70		
3.7.2 Erros comuns	72		
Capítulo 4 ■ Condições	75		
4.1 if	75		
4.2 else	79		
4.3 Estruturas aninhadas	80		
4.4 elif	83		
Capítulo 5 ■ Repetições.....	85		
5.1 Contadores	87		
5.2 Acumuladores	91		
5.3 Interrompendo a repetição	93		
5.4 Repetições aninhadas	95		
Capítulo 6 ■ Listas.....	98		
6.1 Trabalhando com índices.....	101		
6.2 Cópia e fatiamento de listas	101		
6.3 Tamanho de listas.....	104		
6.4 Adição de elementos	105		
6.5 Remoção de elementos da lista.....	108		
		Capítulo 7 ■ Trabalhando com strings.....	137
		7.1 Verificação parcial de strings	138
		7.2 Contagem	140
		7.3 Pesquisa de strings.....	140
		7.4 Posicionamento de strings	144
		7.5 Quebra ou separação de strings.....	145
		7.6 Substituição de strings	145
		7.7 Remoção de espaços em branco.....	146
		7.8 Validação por tipo de conteúdo	147
		7.9 Formatação de strings	150
		7.9.1 Formatação de números.....	152
		7.10 Jogo da força	156
		Capítulo 8 ■ Funções.....	161
		8.1 Variáveis locais e globais.....	168
		8.2 Funções recursivas	170
		8.3 Validação	173
		8.4 Parâmetros opcionais	174
		8.5 Nomeando parâmetros	176
		8.6 Funções como parâmetro	178
		8.7 Empacotamento e desempacotamento de parâmetros	179
		8.8 Desempacotamento de parâmetros	180
		8.9 Funções Lambda	181

8.10 Módulos	181
8.11 Números aleatórios	183
8.12 A função type	185
Capítulo 9 ■ Arquivos	188
9.1 Parâmetros da linha de comando	191
9.2 Geração de arquivos	192
9.3 Leitura e escrita	192
9.4 Processamento de um arquivo	193
9.5 Geração de HTML	200
9.6 Arquivos e diretórios	205
9.7 Um pouco sobre o tempo	210
9.8 Uso de caminhos	213
9.9 Visita a todos os subdiretórios recursivamente	215
Capítulo 10 ■ Classes e objetos	216
10.1 Objetos como representação do mundo real	216
10.2 Passagem de parâmetros	220
10.3 Exemplo de um banco	222
10.4 Herança	227
10.5 Desenvolvendo uma classe para controlar listas	230
10.6 Revisitando a agenda	243
Capítulo 11 ■ Banco de dados	260
11.1 Conceitos básicos	260
11.2 SQL	262
11.3 Python & SQLite	263
11.4 Consultando registros	270
11.5 Atualizando registros	273
11.6 Apagando registros	276
11.7 Simplificando o acesso sem cursores	276
11.8 Acessando os campos como em um dicionário	277
11.9 Gerando uma chave primária	278
11.10 Alterando a tabela	280
11.11 Agrupando dados	281
11.12 Trabalhando com datas	286
11.13 Chaves e relações	289
11.14 Convertendo a agenda para utilizar um banco de dados	292

Capítulo 12 ■ Próximos passos	314
12.1 Programação funcional	314
12.2 Algoritmos	315
12.3 Jogos	315
12.4 Orientação a objetos	316
12.5 Banco de dados	316
12.6 Sistemas web	316
12.7 Outras bibliotecas Python	317
12.8 Listas de discussão	317
Apêndice A ■ Mensagens de erro	318
A.1 SyntaxError	318
A.2 IndentationError	319
A.3 KeyError	320
A.4 NameError	320
A.5 ValueError	321
A.6 TypeError	322
A.7 IndexError	322
Referências	323
Índice remissivo	324

Agradecimentos

Este livro não seria possível sem o incentivo de minha esposa, Emilia Christiane, e a compreensão de meus filhos, Igor, Hanna e Iris. Para quem mora em um país frio como a Bélgica, escrever no verão nem sempre é fácil.

Também agradeço o suporte oferecido por meus pais e avós durante meus estudos, e os conselhos, compreensão e orientação que sempre recebi.

Não poderia me esquecer de meu irmão, Luís Victor, pela ajuda com as imagens em português.

A Luciano Ramalho e aos colegas da lista python-brasil. Luciano, obrigado pelo incentivo para publicar este livro e pelos comentários mais que pertinentes. À comunidade python-brasil pelo esforço e prova de civilidade ao manter as discussões em altíssimo nível, agradando iniciantes, curiosos e profissionais de computação.

Agradecimentos também aos colegas, amigos e alunos da Fundação Matias Machline, hoje Fundação Nokia de Ensino, onde tive oportunidade de estudar e trabalhar como professor de lógica de programação. Aos amigos e colegas do Centro Educacional La Salle e da Fundação Paulo Feitoza, onde ministrei cursos de lógica de programação e de Python.

Revisar este livro foi um trabalho muito gratificante, especialmente pelas mensagens de apoio recebidas na lista Python-Brasil e por e-mail. Esta nova edição traz um capítulo extra, cobrindo o básico de banco de dados, onde introduzimos o uso do SQLite. O capítulo 10, que trata da orientação a objetos, também foi expandido, e vários conceitos específicos ao Python foram adicionados. Muitas pequenas revisões foram feitas por todo o livro, desde a atualização para Python 3.4 até a migração para Windows 8.1.

Apesar da tentação de cobrir tudo que se possa imaginar sobre Python, este livro continua uma obra dedicada àqueles que dão seus primeiros passos na programação de computadores. Nesta nova edição, conceitos mais específicos do Python também foram abordados para tornar o texto e nossos programas mais "Pythonicos".

Aprendi a programar usando a linguagem BASIC ainda em meados dos anos 1980. Lembro-me de construir pequenos programas de desenho, agendas telefônicas e jogos. Armazenamento só em fitas K-7. Antes da internet, e morando no norte do Brasil, a forma de aprender a programar era lendo livros e, claro, programando. A forma mais comum de obter novos programas era por meio de revistas de programação. Essas revistas eram verdadeiramente dedicadas à nova comunidade de usuários de microcomputadores, termo usado na época para diferenciar os computadores domésticos. Elas traziam listagens completas de programas, escritos em BASIC ou Assembly. Em uma época em que o download mal era um sonho distante, digitar esses programas era a única solução para poder executá-los. A experiência de ler e digitar os programas foi muito importante para aprender a programar, mas, infelizmente, poucas revistas técnicas hoje são acessíveis a iniciantes. A complexidade dos programas de hoje também é muito maior, exigindo mais tempo de estudo que antigamente. Embora a internet ajude muito, seguir uma ordem planejada de aprendizado é muito importante.

Ao começar a ensinar programação, o desafio sempre foi encontrar um livro que pudesse ser lido por alunos do ensino médio ou no início do ensino superior. Embora várias obras tenham suprido essa necessidade, o uso de apostilas sempre foi necessário, pois a ordem em que os novos conceitos eram apresentados era quase sempre mais interessante para um dicionário que para o ensino de programação em si. Conceitos importantes para o iniciante eram completamente esquecidos, e o foco maior era dado a assuntos mais complexos. Seguindo um conceito apresentado e utilizado por meus professores do ensino médio, a lógica de programação é mais importante que qualquer linguagem. Quem aprende a programar uma vez fica mais apto a aprender outras linguagens de programação. É nessa lógica de programação para iniciantes que este livro se concentra, apresentando recursos do Python sempre que possível. O intuito é de iniciar o leitor no mundo da programação e prepará-lo para cursos e conceitos mais avançados. Acredito que depois de ler e estudar este livro, você estará apto a ler outras obras de programação e a aprender novas linguagens por conta própria.

Introdução

Este livro foi escrito com o iniciante em programação em mente. Embora a linguagem Python seja muito poderosa e rica em recursos modernos de programação, este livro não pretende ensinar apenas a linguagem em si, mas ensinar a programar em qualquer linguagem. Alguns recursos da linguagem Python não foram utilizados. O objetivo foi privilegiar os exercícios de lógica de programação e melhor preparar o leitor para outras linguagens. Essa escolha não impediu que recursos poderosos da linguagem fossem apresentados, mas este livro não é uma referência da linguagem Python.

Os capítulos foram organizados de forma a apresentar progressivamente os conceitos básicos de programação. A leitura do livro, feita perto de um computador com o interpretador Python aberto, é extremamente recomendada para facilitar a experimentação dos exemplos propostos.

Cada capítulo traz exercícios organizados de forma a explorar o conteúdo apresentado. Alguns exercícios apenas modificam os exemplos do livro, enquanto outros requerem a aplicação dos conceitos apresentados na criação de novos programas. Tente resolver os exercícios na medida em que são apresentados. Embora seja impossível não falar de matemática em um livro de programação, os exercícios foram elaborados para o nível de conhecimento de um aluno do ensino médio, e utiliza problemas comerciais ou do dia a dia. Essa escolha não foi feita para evitar o estudo de matemática, mas para não misturar a introdução de conceitos de programação com novos conceitos matemáticos.

É recomendado que você organize os programas gerados, com uma pasta (diretório) por capítulo, de preferência adicionando o número do exemplo ou exercício aos nomes dos arquivos. Alguns exercícios alteram outros exercícios, mesmo em capítulos diferentes. Uma boa organização desses arquivos vai facilitar seu trabalho de estudo.

Um apêndice foi preparado para ajudar a entender as mensagens de erro que podem ser geradas pelo interpretador Python.

O uso de Python também libera alunos e professores para utilizar o sistema operacional de sua escolha, seja Windows, Linux ou Mac OS X. Todos os exemplos do livro requerem apenas a distribuição padrão da linguagem, que é disponibilizada gratuitamente.

Embora todo esforço tenha sido realizado para evitar erros e omissões, não há garantias de que o livro esteja isento de erros. Se você encontrar falhas no conteúdo, envie um e-mail para erros@nilo.pro.br. Em caso de dúvidas, embora eu não possa garantir resposta a todos os e-mails, envie sua mensagem para duvidas@nilo.pro.br. Dicas, críticas e sugestões podem ser enviadas para professores@nilo.pro.br. O código-fonte e eventuais correções deste livro podem ser encontrados no site <http://python.nilo.pro.br>.

Um resumo do conteúdo de cada capítulo é apresentado a seguir:

Capítulo 1 – Motivação: visa a apresentar o desafio de aprender e estimular o estudo da programação de computadores, apresentando problemas e aplicações do dia a dia.

Capítulo 2 – Preparação do ambiente: instalação do interpretador Python, introdução ao editor de textos, apresentação do IDLE, ambiente de execução, como digitar programas e fazer os primeiros testes com operações aritméticas no interpretador.

Capítulo – 3 Variáveis e entrada de dados: tipos de variáveis, propriedades de cada tipo, operações e operadores. Apresenta o conceito de programa no tempo e uma técnica simples de rastreamento. Entrada de dados pelo teclado, conversão de tipos de dados e erros comuns.

Capítulo 4 – Condições: estruturas condicionais, conceitos de blocos e seleção de linhas a executar com base na avaliação de expressões lógicas.

Capítulo 5 – Repetições: estrutura de repetição while, contadores, acumuladores. Apresenta o conceito de repetição da execução de um bloco e de repetições aninhadas.

Capítulo 6 – Listas: operações com listas, ordenação pelo método de bolhas, pesquisa, utilização de listas como pilhas e filas.

Capítulo 7 – Trabalhando com strings: apresenta operações avançadas com strings. Explora a classe string do Python. O capítulo traz também um jogo simples para fixar os conceitos de manipulação de strings.

Capítulo 8 – Funções: noção de função e transferência de fluxo, funções recursivas, funções lambda, parâmetros, módulos. Apresenta números aleatórios.

Capítulo 9 – Arquivos: criação e leitura de arquivos em disco. Geração de arquivos HTML em Python, operações com arquivos e diretórios, parâmetros pela linha de comando, caminhos.

Capítulo 10 – Classes e objetos: introdução à orientação a objetos. Explica os conceitos de classe, objetos, métodos e herança. Prepara o aluno para continuar estudando o tópico e melhor compreender o assunto.

Capítulo 11 – Banco de dados: introdução à linguagem SQL e ao banco de dados SQLite.

Capítulo 12 – Próximos passos: capítulo final, que lista os próximos passos em diversos tópicos, como jogos, sistemas web, programação funcional, interfaces gráficas e bancos de dados. Visa a apresentar livros e projetos open source pelos quais o aluno pode continuar estudando, dependendo de sua área de interesse.

Apêndice A – Mensagens de erro: explica as mensagens de erro mais frequentes em Python, mostrando suas causas e como resolvê-las.

CAPÍTULO 1

Motivação

Então, você quer aprender a programar?

Programar computadores é uma tarefa que exige tempo e dedicação para ser corretamente aprendida. Muitas vezes não basta só estudar e fazer os exemplos, mas também deixar a mente se acostumar com a nova forma de pensar. Para muitas pessoas, o mais difícil é continuar gostando de programar. Elas desistem nas primeiras dificuldades e não voltam mais a estudar. Outras são mais pacientes, aprendem a não se irritar com a máquina e a assumir seus erros.

Para não sofrer dos males de quem não aprendeu a programar, você precisa responder a algumas perguntas antes de começar:

1. Você quer aprender a programar?
2. Como está seu nível de paciência?
3. Quanto tempo você pretende estudar?
4. Qual o seu objetivo ao programar?

1.1 Você quer aprender a programar?

Responda a essa questão, mas pense um pouco antes de chegar à resposta final. A maneira mais difícil de aprender a programar é não querer programar. A vontade deve vir de você, e não de um professor ou de um amigo. Programar é uma arte e precisa de dedicação para ser dominada. Como tudo o que é desconhecido, é muito difícil quando não a entendemos, mas se torna mais simples à medida que a aprendemos.

Sé você já decidiu aprender a programar, siga para a próxima parte. Se ainda não se convenceu, continue lendo. Programar pode tornar-se um novo *hobby* e até mesmo uma profissão. Se você estuda computação, precisa saber programar. Isso não significa que você será um programador por toda a vida, ou que a programação limitará seu crescimento dentro da área de informática. Uma desculpa que já ouvi muitas vezes é “*eu sei programar, mas não gosto*”. Vários alunos de computação terminam seus cursos sem saber programar; isto é, sem realmente saber programar. Programar é como andar de bicicleta, você não se esquece, mas só aprende fazendo. Ao trocar de uma linguagem de programação para outra, se você realmente aprendeu a programar, terá pouca dificuldade para aprender a nova linguagem. Diferentemente de saber programar, a sintaxe de uma linguagem de programação é esquecida muito facilmente. Não pense que saber programar é decorar todos aqueles comandos, parâmetros e nomes estranhos. Programar é saber utilizar uma linguagem de programação para resolver problemas, ou seja, saber expressar uma solução por meio de uma linguagem de programação.

1.2 Como está seu nível de paciência?

Seja paciente.

Outro erro de quem estuda programação é querer fazer coisas difíceis logo de início.

Qual será seu primeiro programa? Um editor de textos? Uma planilha eletrônica? Uma calculadora?

Não! Será algo bem mais simples... Como somar dois números.

É isso mesmo: somar dois números!

Com o tempo, a complexidade e o tamanho dos programas aumentarão.

Seja paciente.

Programar exige muita paciência e, principalmente, atenção a detalhes. Uma simples vírgula no lugar de um ponto ou aspas esquecidas podem arruinar seu programa. No início, é comum perder a calma ou mesmo se desesperar até aprender a ler o que realmente escrevemos em nossos programas. Nessas horas, paciência nunca é demais. Leia novamente a mensagem de erro ou pare para entender o que não está funcionando corretamente. Nunca pense que o computador está contra você, nem culpe o dia ou o destino.

Seja paciente.

1.3 Quanto tempo você pretende estudar?

Pode-se aprender a programar em poucas horas. Se você é o sujeito que programa o micro-ondas da sua tia, que abre vidros de remédio lendo as instruções da tampa ou que brincou de Lego, programar é seu segundo nome.

Todos nós já programamos algo na vida, nem que seja ir ao cinema no sábado. A questão é: quanto tempo você dedicará para aprender a programar computadores? Como tudo na vida, nada de exageros. Na realidade, tanto o tempo como a forma de estudar variam muito de pessoa para pessoa. Algumas rendem mais estudando em grupo. Outras gostam de ter aulas.

O importante é incluir o estudo da programação em seu estilo preferido. Não tente aprender tudo ou entender tudo rapidamente. Se isso ocorrer, parabéns, há muito pela frente. Caso contrário, relaxe. Se não entender na segunda tentativa, pare e volte a tentar amanhã.

Quando encontrar um problema, tenha calma. Veja o que você escreveu. Verifique se você entende o que está escrito. Um erro comum é querer programar sem saber escrever as instruções. É como querer escrever sem saber falar.

Inicie o estudo com sessões de uma ou no máximo duas horas por dia. Depois ajuste esse tempo a seu ritmo.

1.4 Programar para quê?

Se você não precisa programar para seu trabalho ou estudo, vejamos algumas outras razões:

1.4.1 Escrever páginas web

Hoje, todos estão expostos à web, à Internet e a seus milhares de programas. A web só funciona porque permite a publicação de páginas e mais páginas de textos e imagens com apenas um editor de textos. A mais complexa página que você já acessou é um conjunto de linhas de texto reunidas para instruir um programa, o navegador (*browser*), sobre como apresentar seu conteúdo.

1.4.2 Acertar seu relógio

Você conhece aquelas pessoas que nunca aprenderam a acertar seus relógios? Consigo me lembrar de várias delas...

Seguir instruções é muito importante para tarefas tão simples quanto essas. A sequência de passos para ajustar as horas, minutos e até mesmo a data de seu relógio podem ser encaradas como um programa. Normalmente, aperta-se o botão de ajuste até que um número comece a piscar. Depois, você pode usar um botão para mudar a hora ou ir direto para o ajuste dos minutos. Isso se repete até que você tenha ajustado todos os valores como segundos, dia, mês e, às vezes, o ano.

1.4.3 Aprender a usar mapas

Já se perdeu em uma cidade estranha? Já fez uma lista de passos para chegar a algum lugar? Então você já programou antes. Só de procurar um mapa você já mereceria um prêmio. Ao traçar um caminho de onde você está até onde deseja chegar, você normalmente relaciona uma lista de ruas ou marcos desse caminho. Normalmente é algo como passar três ruas à esquerda, dobrar à direita, dobrar à esquerda... Ou algo como seguir reto até encontrar um sinal de trânsito ou um rio. Isso é programar. Seguir seu programa é a melhor forma de saber se o que você escreveu está correto ou se você agora está realmente perdido.

1.4.4 Mostrar para seus amigos que você sabe programar

Esta pode ser a razão mais complicada. Vamos ver isso como um subproduto do aprendizado, e não seu maior objetivo. Se essa for a razão de aprender a programar, é melhor continuar lendo e arranjar outra.

Programar é um esforço para você realizar algo. É uma tarefa que exige dedicação e que traz muita satisfação pessoal. Seus programas podem ser legais, mas afinal são seus programas e não somente você.

1.4.5 Parecer estranho

Agora estamos conversando. Entender o que milhares de linhas de programa significam pode realmente gerar uma fama como essa entre os leigos. Se esse é seu objetivo, saiba que há maneiras mais fáceis, como parar de tomar banho, deixar

as unhas crescerem, ter um cabelo de cor laranja ou roxa, parecer um roqueiro sem nunca ter tocado em uma banda etc.

Embora boa parte dos programadores que conheço não seja exatamente o que eu classifico de 100% normal, ninguém o é.

Saber programar não significa que você seja louco ou muito inteligente. Saber programar também não significa que você não seja louco ou que não seja muito inteligente. Imagine aprender a programar como qualquer outra coisa que você já aprendeu.

Um dia, sua mente poderá ter pensamentos estranhos, mas, quando esse dia chegar, pode apostar que você saberá.

1.4.6 Entender melhor como seu computador funciona

Programando você pode começar a entender por que aquela operação falhou ou por que o programa simplesmente fechou sem nem mesmo avisar o motivo.

Programar também pode ajudá-lo a utilizar melhor sua planilha ou editor de textos. O tipo de pensamento que se aprende programando servirá não só para fazer programas, mas também para usar programas.

1.4.7 Cozinhar

Uma vez precisei cozinhar um prato, mas as instruções estavam escritas em alemão. Não sei nada de alemão. Peguei o primeiro dicionário e comecei a traduzir as principais palavras. Com as palavras traduzidas, tentei entender o que deveria realmente fazer.

Naquela noite, o jantar foi apenas uma sopa instantânea. Receitas podem ser vistas como programas. E como programas, só é possível segui-las se você entender aquilo que foi escrito.

A simples sequência de instruções não ajuda uma pessoa que não entenda seus efeitos.

Para algumas pessoas, programar é mais fácil que aprender alemão (ou qualquer outro idioma estrangeiro). E como qualquer outra língua, não se aprende apenas com um dicionário.

Idiomas humanos são ricos em contextos, e cada palavra costuma ter múltiplos significados. A boa notícia: linguagens de programação são feitas para que máquinas possam entender o que ali foi representado. Isso significa que entender um programa é muito fácil, quase como consultar um dicionário. Outra boa notícia é que a maioria das linguagens contém conjuntos pequenos de “palavras”.

1.4.8 Salvar o mundo

Uma boa razão para aprender a programar é salvar o mundo. Isso mesmo!

Todos os dias, milhares de quilos de alimento são desperdiçados ou não chegam onde deveriam por falta de organização. Programando você pode ajudar a criar sistemas e até mesmo programas que ajudem outras pessoas a se organizar.

Outra boa ação é ajudar um projeto de software livre. Isso permitirá que muitas pessoas que não podem pagar por programas de computador se beneficiem deles sem cometer crime algum.

1.4.9 Software livre

Aliás, você tem licença de uso para todos os seus programas?

Se a resposta é não, saiba que programadores aprendem Linux e outros sistemas operacionais muito mais rapidamente. Também conseguem tirar maior proveito desses sistemas porque conseguem programá-los.

Se não existe, crie. Se é ruim, melhore.

Separar a programação em um mundo à parte pode ser sua primeira ideia estranha entre muitas.

Criar mundos dentro de programas e computadores pode ser a segunda.

1.5 Por que Python?

A linguagem de programação Python é muito interessante como primeira linguagem de programação devido à sua simplicidade e clareza. Embora simples, é também uma linguagem poderosa, podendo ser usada para administrar sistemas e desenvolver grandes projetos. É uma linguagem clara e objetiva, pois vai direto ao ponto, sem rodeios.

Python é software livre, ou seja, pode ser utilizada gratuitamente, graças ao trabalho da Python Foundation¹ e de inúmeros colaboradores. Você pode utilizar Python em praticamente qualquer arquitetura de computadores ou sistema operacional, como Linux², FreeBSD³, Microsoft Windows ou Mac OS X⁴.

Python vem crescendo em várias áreas da computação, como inteligência artificial, banco de dados, biotecnologia, animação 3D, aplicativos móveis (celulares), jogos e mesmo como plataforma web. Isso explica porque Python é famosa por ter “batteries included”, ou seja, baterias inclusas, fazendo referência a um produto completo que pode ser usado prontamente (quem nunca ganhou um presente de Natal que veio sem pilhas?). Hoje é difícil encontrar uma biblioteca que não tenha *bindings* (versão) em Python. Esse fato torna o aprendizado da linguagem muito mais interessante, uma vez que aprender a programar em Python é poder continuar a utilizar os conhecimentos adquiridos mesmo depois de aprender a programar para resolver problemas reais.

Uma grande vantagem de Python é a legibilidade dos programas escritos nessa linguagem. Outras linguagens de programação utilizam inúmeras marcações, como ponto (.) ou ponto e vírgula (;), no fim de cada linha, além dos marcadores de início e fim de bloco como chaves ({ }) ou palavras especiais (begin/end). Esses marcadores tornam os programas um tanto mais difíceis de ler e felizmente não são usados em Python. Veremos mais sobre blocos e marcações nos capítulos seguintes.

Outro bom motivo para aprender Python é poder obter resultados em pouco tempo. Como Python é uma linguagem completa, contando com bibliotecas para acessar bancos de dados, processar arquivos XML, construir interfaces gráficas e mesmo jogos, podemos utilizar muitas funções já existentes escrevendo poucas linhas de código. Isso aumenta a produtividade do programador, pois ao utilizarmos bibliotecas usamos programas desenvolvidos e testados por outras pessoas. Isso reduz o número de erros e permite que você se concentre realmente no problema que quer resolver.

Vejamos um pequeno programa escrito em Python na listagem 1.1.

► Listagem 1.1 – Programa Olá Mundo

```
print ("Olá!")
```

¹ <http://www.python.org>

² <http://www.kernel.org> ou <http://www.ubuntu.com> para obter o pacote completo

³ <http://www.freebsd.org>

⁴ <http://www.apple.com/macosx>

A listagem do programa 1.1 tem apenas uma linha de código. A palavra `print` é uma função utilizada para enviar dados para a tela do computador. Ao escrevermos `print ("Olá")`, ordenamos ao computador que exiba o texto “Olá!” na tela. Veja o que seria exibido na tela ao se executar esse programa no computador:

```
Olá!
```

Observe que as aspas (") não aparecem na tela. Esse é um dos detalhes da programação: precisamos marcar ou limitar o início e o fim de nossas mensagens com um símbolo, nesse caso, aspas. Como podemos exibir praticamente qualquer texto na tela, as primeiras aspas indicam o início da mensagem, e as seguintes, o fim. Ao programar, não podemos esquecer as limitações do computador. Um computador não interpreta textos como seres humanos. A máquina não consegue diferenciar o que é programa ou mensagem. Se não utilizarmos as aspas, o computador interpretará nossa mensagem como um comando da linguagem Python, gerando um erro.

O interpretador Python é uma grande ferramenta para o aprendizado da linguagem. O interpretador é o programa que permite digitar e testar comandos escritos em Python e verificar os resultados instantaneamente. Veremos como utilizar o interpretador na seção 2.2.

A linguagem Python foi escolhida para este livro por simplificar o trabalho de aprendizado e fornecer grande poder de programação. Como é um software livre, disponível praticamente para qualquer tipo de computador, sua utilização não envolve a aquisição de licenças de uso, muitas vezes a um custo proibitivo.

Bem-vindo ao mundo da programação!

CAPÍTULO 2

Preparando o ambiente

Antes de começarmos, precisamos instalar o interpretador da linguagem Python. O interpretador é um programa que aceita comandos escritos em Python e os executa, linha a linha. É ele quem vai traduzir nossos programas em um formato que pode ser executado pelo computador. Sem o interpretador Python, nossos programas não podem ser executados, sendo considerados apenas como texto. O interpretador também é responsável por verificar se escrevemos corretamente nossos programas, mostrando mensagens de erro, caso encontre algum problema.

O interpretador Python não vem instalado com o Microsoft Windows: você deverá instalá-lo fazendo um download da internet. Se você utiliza Mac OS X ou Linux, provavelmente isso já foi feito. Veja, a seguir, como verificar se você tem a versão correta.

2.1 Instalação do Python

Neste livro, usamos o Python versão 3.4. A linguagem sofreu diversas modificações entre as versões 2 e 3. A versão 3.4 foi escolhida por permitir a correta utilização dos novos recursos, além do fato de que é interessante aprendermos a versão mais nova. A versão 2.7 é também muito interessante, mas permite misturar a sintaxe (forma de escrever) do Python 2 com a do Python 3. Para evitar complicações desnecessárias, apresentaremos apenas as novas formas de escrever, e deixaremos o Python 3.4 verificar se fizemos tudo corretamente. O Python 3.4 ainda não estava disponível para todas as distribuições Linux na época de lançamento desta edição. Você pode utilizar também o Python 3.3 com todos os exemplos do livro.