

ZED A. SHAW

Aprenda **PYTHON 3** do JEITO CERTO

Uma introdução muito simples
ao incrível mundo dos
computadores e da codificação



ALTA BOOKS
E D I T O R A
Rio de Janeiro, 2019

Sumário

Prefácio.....	xvii
Melhorias na Edição Python 3.....	xvii
O Modo Certo É Mais Fácil	xviii
Leitura e Escrita	xviii
Atenção aos Detalhes.....	xviii
Identificação das Diferenças.....	xix
Pergunte, Não Fique Olhando	xix
Não Copie e Cole.....	xix
Usando os Vídeos Incluídos	xix
Observação sobre Prática e Persistência.....	xx
Agradecimentos	xxi
Exercício 0 Configuração	1
macOS.....	1
macOS: O que Você Deve Ver	2
Windows	2
Windows: O que Você Deve Ver.....	3
Linux.....	3
Linux: O que Você Deve Ver.....	4
Descobrimos Coisas na Internet	5
Avisos para os Iniciantes.....	5
Editores de Texto Alternativos	6
Exercício 1 Um Bom Programa Inicial.....	7
O que Você Deve Ver.....	9
Exercícios Simulados	11
Perguntas Comuns dos Alunos	11
Exercício 2 Comentários e Cerquilhas	13
O que Você Deve Ver.....	13
Exercícios Simulados	14
Perguntas Comuns dos Alunos	14
Exercício 3 Números e Matemática.....	15
O que Você Deve Ver.....	16

Exercícios Simulados	16
Perguntas Comuns dos Alunos	17
Exercício 4 Variáveis e Nomes	19
O que Você Deve Ver	20
Exercícios Simulados	20
Perguntas Comuns dos Alunos	20
Exercício 5 Mais Variáveis e Impressão	23
O que Você Deve Ver	23
Exercícios Simulados	24
Perguntas Comuns dos Alunos	24
Exercício 6 Strings e Texto	25
O que Você Deve Ver	26
Exercícios Simulados	26
Corrompa.....	26
Perguntas Comuns dos Alunos	27
Exercício 7 Mais Impressão	29
O que Você Deve Ver	29
Exercícios Simulados	30
Corrompa.....	30
Perguntas Comuns dos Alunos	30
Exercício 8 Imprimindo, Imprimindo	31
O que Você Deve Ver	31
Exercícios Simulados	32
Perguntas Comuns dos Alunos	32
Exercício 9 Imprimindo, Imprimindo, Imprimindo	33
O que Você Deve Ver	33
Exercícios Simulados	34
Perguntas Comuns dos Alunos	34
Exercício 10 O que Foi Isso?	35
O que Você Deve Ver	36
Sequências de Escape.....	36
Exercícios Simulados	37
Perguntas Comuns dos Alunos	37

Exercício 11 Fazendo Perguntas.....	39
O que Você Deve Ver.....	40
Exercícios Simulados.....	40
Perguntas Comuns dos Alunos.....	40
Exercício 12 Perguntando às Pessoas.....	41
O que Você Deve Ver.....	41
Exercícios Simulados.....	41
Perguntas Comuns dos Alunos.....	42
Exercício 13 Parâmetros, Descompactação, Variáveis.....	43
Pare! Os Recursos Têm Outro Nome.....	44
O que Você Deve Ver.....	44
Exercícios Simulados.....	45
Perguntas Comuns dos Alunos.....	45
Exercício 14 Prompt e Passagem.....	47
O que Você Deve Ver.....	48
Exercícios Simulados.....	48
Perguntas Comuns dos Alunos.....	48
Exercício 15 Lendo Arquivos.....	51
O que Você Deve Ver.....	52
Exercícios Simulados.....	53
Perguntas Comuns dos Alunos.....	53
Exercício 16 Lendo e Gravando Arquivos.....	55
O que Você Deve Ver.....	56
Exercícios Simulados.....	57
Perguntas Comuns dos Alunos.....	57
Exercício 17 Mais Arquivos.....	59
O que Você Deve Ver.....	60
Exercícios Simulados.....	60
Perguntas Comuns dos Alunos.....	61
Exercício 18 Nomes, Variáveis, Código, Funções.....	63
O que Você Deve Ver.....	64
Exercícios Simulados.....	65
Perguntas Comuns dos Alunos.....	66

Exercício 19 Funções e Variáveis	67
O que Você Deve Ver	68
Exercícios Simulados	68
Perguntas Comuns dos Alunos	68
Exercício 20 Funções e Arquivos	71
O que Você Deve Ver	72
Exercícios Simulados	72
Perguntas Comuns dos Alunos	72
Exercício 21 As Funções Podem Retornar Algo	75
O que Você Deve Ver	76
Exercícios Simulados	76
Perguntas Comuns dos Alunos	77
Exercício 22 O que Você Sabe até Agora?	79
O que Você Está Aprendendo	79
Exercício 23 Strings, Bytes e Codificações de Caracteres	81
Pesquisa Inicial	81
Interruptores, Convenções e Codificações	83
Dissecando a Saída	85
Dissecando o Código	85
Mergulhando nas Codificações	87
Corrompendo	88
Exercício 24 Mais Prática	89
O que Você Deve Ver	90
Exercícios Simulados	90
Perguntas Comuns dos Alunos	91
Exercício 25 Mais Prática Ainda	93
O que Você Deve Ver	94
Exercícios Simulados	95
Perguntas Comuns dos Alunos	96
Exercício 26 Parabéns, Faça um Teste!	97
Perguntas Comuns dos Alunos	98
Exercício 27 Memorizando a Lógica	99
Termos da Verdade	100

Tabelas-verdade	100
Perguntas Comuns dos Alunos	101
Exercício 28 Prática com Booleanos	103
O que Você Deve Ver	105
Exercícios Simulados	105
Perguntas Comuns dos Alunos	105
Exercício 29 A Instrução If	107
O que Você Deve Ver	108
Exercícios Simulados	108
Perguntas Comuns dos Alunos	108
Exercício 30 Else e If	109
O que Você Deve Ver	110
Exercícios Simulados	110
Perguntas Comuns dos Alunos	110
Exercício 31 Tomando Decisões	111
O que Você Deve Ver	112
Exercícios Simulados	112
Perguntas Comuns dos Alunos	112
Exercício 32 Loops e Listas	115
O que Você Deve Ver	116
Exercícios Simulados	117
Perguntas Comuns dos Alunos	117
Exercício 33 Loops While	119
O que Você Deve Ver	120
Exercícios Simulados	120
Perguntas Comuns dos Alunos	121
Exercício 34 Acessando os Elementos das Listas	123
Exercícios Simulados	124
Exercício 35 Desvios e Funções	125
O que Você Deve Ver	126
Exercícios Simulados	127
Perguntas Comuns dos Alunos	127

Exercício 36 Criando e Depurando	129
Regras para as instruções If	129
Regras para Loops	129
Dicas para Depurar	130
Dever de Casa	130
Exercício 37 Revisão dos Símbolos	131
Palavras-chave.....	131
Tipos de Dados	132
Sequências de Escape de String	133
Antigos Formatos de String	133
Operadores	134
Leitura de Código	135
Exercícios Simulados	136
Perguntas Comuns dos Alunos	136
Exercício 38 Fazendo Coisas com Listas.....	137
O que Você Deve Ver.....	138
O que as Listas Podem Fazer.....	139
Quando Usar as Listas.....	140
Exercícios Simulados	140
Perguntas Comuns dos Alunos	141
Exercício 39 Dicionários, Ah, os Adoráveis Dicionários	143
Exemplo de Dicionário	144
O que Você Deve Ver.....	146
O que os Dicionários Podem Fazer	146
Exercícios Simulados	147
Perguntas Comuns dos Alunos	147
Exercício 40 Módulos, Classes e Objetos	149
Os Módulos São Como Dicionários	149
As Classes São Como Módulos	150
Os Objetos São Como uma Importação	151
Obtendo Coisas de Coisas	152
Um Primeiro Exemplo de Classe	153
O que Você Deve Ver.....	153
Exercícios Simulados	153
Perguntas Comuns dos Alunos	154

Exercício 41 Aprendendo o Jargão da Orientação a Objetos	155
Exercícios de Palavras	155
Exercícios de Frases	156
Exercícios Combinados	156
Teste de Leitura	157
Pratique Traduzir de Inglês para Código	159
Lendo Mais Código	159
Perguntas Comuns dos Alunos	159
Exercício 42 É-Um, Tem-Um, Objetos e Classes	161
Como Fica no Código	162
Sobre class Name(object)	164
Exercícios Simulados	164
Perguntas Comuns dos Alunos	165
Exercício 43 Básico de Análise e Design Orientados a Objetos	167
Análise de um Mecanismo de Jogo Simples	168
Escreva ou Desenhe o Problema	168
Extraia os Conceitos-chave e Pesquise	169
Crie uma Hierarquia de Classes e um Mapa de Objetos para os Conceitos	169
Codifique as Classes e um Teste para Executá-las	170
Repita e Refine	172
De Cima para Baixo versus de Baixo para Cima	172
Código para "Gothons from Planet Percal #25"	173
O que Você Deve Ver	179
Exercícios Simulados	180
Perguntas Comuns dos Alunos	180
Exercício 44 Herança versus Composição	181
O que É Herança?	181
Herança Implícita	182
Sobrescreva Explicitamente	183
Altere Antes ou Depois	183
Os Três Combinados	185
O Motivo de super()	186
Usando super() com __init__	186
Composição	187
Quando Usar a Herança ou a Composição	188

Exercícios Simulados	189
Perguntas Comuns dos Alunos	189
Exercício 45 Você Cria um Jogo	191
Avaliando o Jogo	191
Estilo de Função	192
Estilo de Classe	192
Estilo de Código	193
Bons Comentários	193
Avalie Seu Jogo	194
Exercício 46 Esqueleto do Projeto	195
Configuração do macOS/Linux	195
Configuração do Windows 10	197
Criando o Diretório de Esqueletos de Projetos	198
Estrutura Final do Diretório	199
Testando a Configuração	201
Usando o Esqueleto	201
Teste Requerido	201
Perguntas Comuns dos Alunos	202
Exercício 47 Teste Automático	203
Escrevendo um Caso de Teste	203
Diretrizes do Teste	205
O que Você Deve Ver	206
Exercícios Simulados	206
Perguntas Comuns dos Alunos	206
Exercício 48 Entrada Avançada do Usuário	207
Léxico do Jogo	207
Dividindo uma Frase	208
Tuplas do Léxico	208
Examinando a Entrada	208
Exceções e Números	209
Desafio do Teste Primeiro	209
O que Você Deve Testar	211
Exercícios Simulados	212
Perguntas Comuns dos Alunos	213

Exercício 49 Criando Frases	215
Combine e Olhe	215
Gramática da Frase.....	216
Uma Observação sobre as Exceções	216
Código do Analisador	216
Lidando com o Analisador	219
O que Você Deve Testar	220
Exercícios Simulados	220
Perguntas Comuns dos Alunos	220
Exercício 50 Seu Primeiro Website	221
Instalando o flask	221
Crie um Projeto “Hello World” Simples.....	222
O que Está Acontecendo?.....	223
Corrigindo os Erros	223
Crie Templates Básicos	225
Exercícios Simulados	226
Perguntas Comuns dos Alunos	227
Exercício 51 Obtendo Entrada de um Navegador.....	229
Como Funciona a Web.....	229
Como os Formulários Funcionam	231
Criando Formulários HTML	232
Criando um Template do Layout	234
Escrevendo Testes Automáticos para Formulários	236
Exercícios Simulados	237
Corrompendo	237
Exercício 52 O Início do Seu Jogo da Web.....	239
Refatorando o Jogo do Exercício 43	239
Criando um Engine.....	244
Seu Exame Final	247
Perguntas Comuns dos Alunos	247
Próximas Etapas.....	249
Como Aprender Qualquer Linguagem de Programação	250
Conselhos de um Velho Programador.....	253

Curso Rápido da Linha de Comando	255
Introdução: Boca Calada e Shell	255
Como Usar o Apêndice	255
Você Irá Memorizar as Coisas	256
Configuração	257
Faça Isto	257
Você Aprendeu Isto	258
Faça Mais	258
Caminhos, Pastas, Diretórios (pwd)	260
Faça Isto	260
Você Aprendeu Isto	261
Faça Mais	262
Se Ficar Perdido	262
Faça Isto	262
Você Aprendeu Isto	262
Crie um Diretório (mkdir)	263
Faça Isto	263
Você Aprendeu Isto	264
Faça Mais	265
Mude o Diretório (cd)	265
Faça Isto	265
Você Aprendeu Isto	268
Faça Mais	268
Liste o Diretório (ls)	269
Faça Isto	269
Você Aprendeu Isto	272
Faça Mais	272
Remova o Diretório (rmdir)	272
Faça Isto	273
Você Aprendeu Isto	275
Faça Mais	275
Movendo-se (pushd, popd)	275
Faça Isto	275
Você Aprendeu Isto	277
Faça Mais	278

Criando Arquivos Vazios (touch/New-Item).....	278
Faça Isto	278
Você Aprendeu Isto.....	279
Faça Mais	279
Copie um Arquivo (cp).....	279
Faça Isto	279
Você Aprendeu Isto.....	281
Faça Mais	282
Movendo um Arquivo (mv)	282
Faça Isto	282
Você Aprendeu Isto.....	284
Faça Mais	284
Exiba um Arquivo (less/more)	284
Faça Isto	285
Você Aprendeu Isto.....	285
Faça Mais	285
Envie um Arquivo (cat)	286
Faça Isto	286
Você Aprendeu Isto.....	287
Faça Mais	287
Removendo um Arquivo (rm)	287
Faça Isto	287
Você Aprendeu Isto.....	289
Faça Mais	289
Saindo do Terminal (exit).....	289
Faça Isto	289
Você Aprendeu Isto.....	289
Faça Mais	290
Próximas Etapas da Linha de Comando.....	290
Referências do Bash do Unix	290
Referências do PowerShell.....	291
Índice	293

Configuração

Este exercício não tem código. É apenas o exercício que você conclui para fazer o computador executar o Python. Você deve seguir as instruções com maior exatidão possível. Se tiver problemas para acompanhar as instruções escritas, assista aos vídeos para sua plataforma.

AVISO! Se não souber como usar o PowerShell no Windows, Terminal no macOS ou bash no Linux, precisará aprender primeiro. Você deve fazer os exercícios no apêndice antes de continuar.

macOS

Faça as seguintes tarefas para concluir o exercício:

1. Vá para <https://www.python.org/downloads/release/python-360/> (conteúdo em inglês) e baixe a versão “Mac OS X 64-bit/32-bit installer”. Instale-a como qualquer outro software.
2. Vá para <https://atom.io> (conteúdo em inglês) com seu navegador, acesse o editor de texto Atom e instale-o. Se o Atom não for adequado, veja a seção *Editores de Texto Alternativos* no final deste exercício.
3. Posicione o Atom (seu editor de texto) na barra de tarefas para poder acessá-lo com facilidade.
4. Encontre o programa Terminal. Procure. Você achará.
5. Posicione o Terminal na barra de tarefas também.
6. Execute o programa Terminal. Ele não vai parecer lá grande coisa.
7. Em seu programa Terminal, execute o `python3.6`. Você executa as coisas no Terminal ao digitar o nome e pressionar RETURN.
8. Digite `quit()`, Enter e saia do `python3.6`.
9. Deverá voltar para um prompt parecido com o que tinha antes de ter digitado Python. Se não, descubra o motivo.
10. Aprenda a criar um diretório no Terminal.
11. Aprenda a mudar para um diretório no Terminal.
12. Use o editor para criar um arquivo nesse diretório. Crie o arquivo, salve-o com `Save` ou `Save As...` e escolha o diretório.

13. Volte para o Terminal usando o teclado para trocar as janelas.
14. De volta ao Terminal, liste o diretório com `ls` para ver o arquivo recém-criado.

macOS: O que Você Deve Ver

Abaixo sou eu fazendo isso no meu computador macOS, no Terminal. Seu computador pode ser diferente, mas deve ficar parecido com isso:

```
$ python3.6
Python 3.6.0 (default, Feb  2 2017, 12:48:29)
[GCC 4.2.1 Compatible Apple LLVM 7.0.2 (clang-700.1.81)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>>
~ $ mkdir lpthw
~ $ cd lpthw
lpthw $ ls
# ... Use seu editor de texto aqui para editar o test.txt...
lpthw $ ls
test.txt
lpthw $
```

Windows

1. Vá para <https://atom.io> (conteúdo em inglês) com seu navegador, acesse o editor de texto Atom e instale-o. Você não precisa ser administrador para fazer isso.
2. Acesse o Atom com facilidade colocando-o na área de trabalho e/ou no Quick Launch. As duas opções ficam disponíveis durante a configuração. Se você não conseguir executar o Atom porque o computador não é rápido o bastante, veja a seção *Editores de Texto Alternativos* no final do exercício.
3. Execute o PowerShell no menu Start. Procure e pressione Enter para executá-lo.
4. Crie um atalho para ele na área de trabalho e/ou no Quick Launch para ter fácil acesso.
5. Execute o programa PowerShell (que chamarei de Terminal mais tarde). Ele não vai parecer lá grande coisa.
6. Baixe o Python 3.6 em <https://www.python.org/downloads/release/python-360/> (conteúdo em inglês) e instale-o. *Marque a caixa que informa para adicionar o Python 3.6 ao seu path.*
7. No programa PowerShell (Terminal), execute o Python. Você executa as coisas no Terminal digitando o nome e pressionando Enter. Se digitar python e ele não for executado, terá que reinstalar o Python e marcar a caixa "Add python to the PATH". Ela é muito pequena, portanto, preste atenção.
8. Digite `quit()` e pressione Enter para sair do Python.

9. Você deverá estar de volta para um prompt parecido com o que tinha antes de ter digitado python. Se não, descubra o motivo.
10. Aprenda a criar um diretório no PowerShell (Terminal).
11. Aprenda a mudar para um diretório no PowerShell (Terminal).
12. Use seu editor para criar um arquivo nesse diretório. Crie o arquivo, salve-o com Save ou Save As . . . e escolha esse diretório.
13. Volte para o PowerShell (Terminal) usando apenas o teclado para trocar as janelas. Descubra como fazer isso, se não souber.
14. De novo no PowerShell (Terminal), liste o diretório para ver o arquivo recém-criado.

De agora em diante, quando eu mencionar “Terminal” ou “shell”, quero dizer PowerShell, e é o que você deve usar. Quando eu executo o python3.6, você pode digitar apenas python.

Windows: O que Você Deve Ver

```
> python
>>> quit()
> mkdir lpthw
> cd lpthw
... Aqui você usará seu editor de texto para criar o test.txt no lpthw...
>
> dir
Volume in drive C is
Volume Serial Number is 085C-7E02

Directory of C:\Documents and Settings\you\lpthw

04.05.2010  23:32    <DIR>          .
04.05.2010  23:32    <DIR>          ..
04.05.2010  23:32                6 test.txt
                1 File(s)                6 bytes
                2 Dir(s)  14 804 623 360 bytes free

>
```

Ainda estará correto se você vir informações diferentes das minhas, mas elas devem ser parecidas.

Linux

O Linux é um sistema operacional variado, com muitos modos diferentes de instalar o software. Suponho que, se você está executando o Linux, então sabe como instalar os pacotes, portanto, aqui estão suas instruções:

1. Use o gerenciador de pacotes para instalar o Python 3.6 e, se não conseguir, baixe a fonte em <https://www.python.org/downloads/release/python-360/> (conteúdo em inglês) e instale a partir do código-fonte.
2. Use o gerenciador de pacotes Linux e instale o editor de texto Atom. Se o Atom não for adequado para você, veja a seção *Editores de Texto Alternativos* no final do exercício.
3. Verifique se você pode acessar o Atom com facilidade posicionando-o no menu do gerenciador de janelas.
4. Encontre o programa Terminal. Ele pode estar nomeado como GNOME Terminal, Konsole ou xterm.
5. Posicione o Terminal na barra de tarefas também.
6. Execute o programa Terminal. Ele não vai parecer lá grande coisa.
7. No programa Terminal, execute o `python3.6`. Você executa as coisas no Terminal digitando o nome do comando e pressionando Enter. Se não conseguir executar o `python3.6`, tente executar apenas Python.
8. Digite `quit()` e pressione Enter para sair do Python.
9. Deverá voltar para um prompt parecido com o que tinha antes de ter digitado `python`. Se não, descubra o motivo.
10. Aprenda como criar um diretório no Terminal.
11. Aprenda como mudar para um diretório no Terminal.
12. Use o editor para criar um arquivo nesse diretório. Em geral, você irá criar o arquivo, salvá-lo com Save ou Save As... e escolher esse diretório.
13. Volte para o Terminal usando apenas o teclado para trocar as janelas. Procure se não conseguir encontrá-lo.
14. De volta no Terminal, liste o diretório para ver o arquivo recém-criado.

Linux: O que Você Deve Ver

```
$ python
>>> quit()
$ mkdir lpthw
$ cd lpthw
# ... Use seu editor de texto aqui para editar o test.txt...
$ ls
test.txt
$
```

Ainda estará correto se você vir informações diferentes das minhas, mas elas devem ser parecidas.

Descobrimos Coisas na Internet

Grande parte deste livro é aprender a pesquisar os tópicos de programação online. Pedirei a você para “pesquisar isto na internet” e seu trabalho será usar um mecanismo de busca para encontrar a resposta. O motivo de você pesquisar, ao invés de eu dar a resposta, é porque quero que seja um aluno independente, que não precisa do meu livro quando terminar de lê-lo. Se você conseguir encontrar as respostas para suas perguntas online, estará mais perto de não precisar de mim, e esse é meu objetivo.

Graças a mecanismos de busca, como o Google, você pode encontrar facilmente qualquer coisa que eu pedir. Se eu pedir, “pesquise online as funções list do Python”, você fará simplesmente isto:

1. Acessar <http://google.com>.
2. Digitar: `python3 list functions`.
3. Ler os sites listados para encontrar a melhor resposta.

Avisos para os Iniciantes

Este exercício acabou. Ele poderá ser difícil dependendo de sua familiaridade com o computador. Se for difícil, reserve um tempo para ler, estudar e concluir o exercício, porque até que você possa fazer essas coisas muito básicas, achará difícil programar.

Se alguém disser para você parar em um exercício específico neste livro ou pular outros, deverá ignorar essa pessoa. Qualquer pessoa que tenta ocultar o conhecimento de você, ou pior, faz você obtê-lo através dela, ao invés de desenvolvê-lo através do seu próprio esforço, está tentando torná-lo dependente dela para suas habilidades. Não dê ouvidos e faça os exercícios para que aprenda como educar a si mesmo.

Eventualmente algum programador lhe dirá para usar o macOS ou o Linux. Se ele gosta de fontes e tipografia, pedirá que use um computador macOS. Se gosta de controle e tem uma barba enorme, ele (ou ela, se você prefere pronomes de tratamento sem gênero) pedirá para instalar o Linux. Mais uma vez: use qualquer computador que tiver agora e que funcione. Tudo o que precisa é de um editor, um Terminal e o Python.

Finalmente, o propósito desta configuração é ajudá-lo a fazer três coisas com muita segurança enquanto trabalha nos exercícios:

1. *Escrever* os exercícios usando o editor de texto.
2. *Executar* os exercícios escritos.
3. *Corrigir* quando tiverem problemas.
4. Repetir.

Tudo mais apenas irá confundi-lo, portanto, mantenha o plano.

Editores de Texto Alternativos

Os editores de texto são muito importantes para um programador, mas, como iniciante, você só precisa de um simples editor de texto de programação. Eles são diferentes do software para escrever histórias e livros porque funcionam com as necessidades únicas do código do computador. Recomendo o Atom neste livro porque é gratuito e funciona em quase toda situação. Contudo, o Atom pode não executar bem em seu computador, portanto, veja algumas alternativas:

Nome do Editor	Funciona no	Onde obter (conteúdos em inglês)
Visual Studio Code	Windows, macOS, Linux	https://code.visualstudio.com
Notepad++	Windows	https://notepad-plus-plus.org
gEdit	Linux, macOS, Windows	https://github.com/GNOME/gedit
Textmate	macOS	https://github.com/textmate/textmate
SciTE	Windows, Linux	http://www.scintilla.org/SciTE.html
jEdit	Linux, macOS, Windows	http://www.jedit.org

Eles estão classificados pela probabilidade de funcionar. Lembre que esses projetos podem estar abandonados, mortos ou não funcionar mais em seu computador. Se você experimentar um e ele não funcionar, tente outro. Também listei a coluna “Funciona no” em ordem de onde é mais provável que ele funcione, portanto, se você estiver no Windows, veja os editores nos quais o Windows é listado primeiro.

Se já sabe como usar o Vim ou o Emacs, fique à vontade para usá-los. Se nunca usou, evite. Os programadores podem tentar convencê-lo a usar o Vim ou o Emacs, mas só irá atrapalhar. Seu foco é aprender o Python, não o Vim nem o Emacs. Se tentar usar o Vim e não souber como sair, digite `:q!` ou `ZZ`. Se alguém disser para usar o Vim e nem sequer informou isso, então agora você sabe porque não deve dar ouvidos a essa pessoa.

Não use um Ambiente de Desenvolvimento Integrado (IDE) neste livro. Contar com um IDE significa que você não consegue trabalhar com novas linguagens de programação até que alguma empresa decida vender um IDE para tal linguagem. Também significa que você não poderá usar essa nova linguagem até que ela seja grande o bastante para justificar uma base lucrativa de clientes. Se tiver confiança de que pode trabalhar apenas com um editor de texto de programação (como Vim, Emacs, Atom etc.), então não precisa esperar por terceiros. Os IDEs são ótimos em algumas situações (como trabalhar com uma base de código existente enorme), mas ser dependente deles limitará seu futuro.

Você também não deve usar o IDLE. Ele tem sérias limitações em seu funcionamento e não é um software muito bom. Tudo o que você precisa é de um editor de texto simples, um shell e o Python.

Um Bom Programa Inicial

AVISO! Se você pulou o Exercício 0, não está usando corretamente este livro. Está tentando usar o IDLE ou um IDE? Eu disse para não usar no Exercício 0, portanto, não use. Se pulou o Exercício 0, volte e leia.

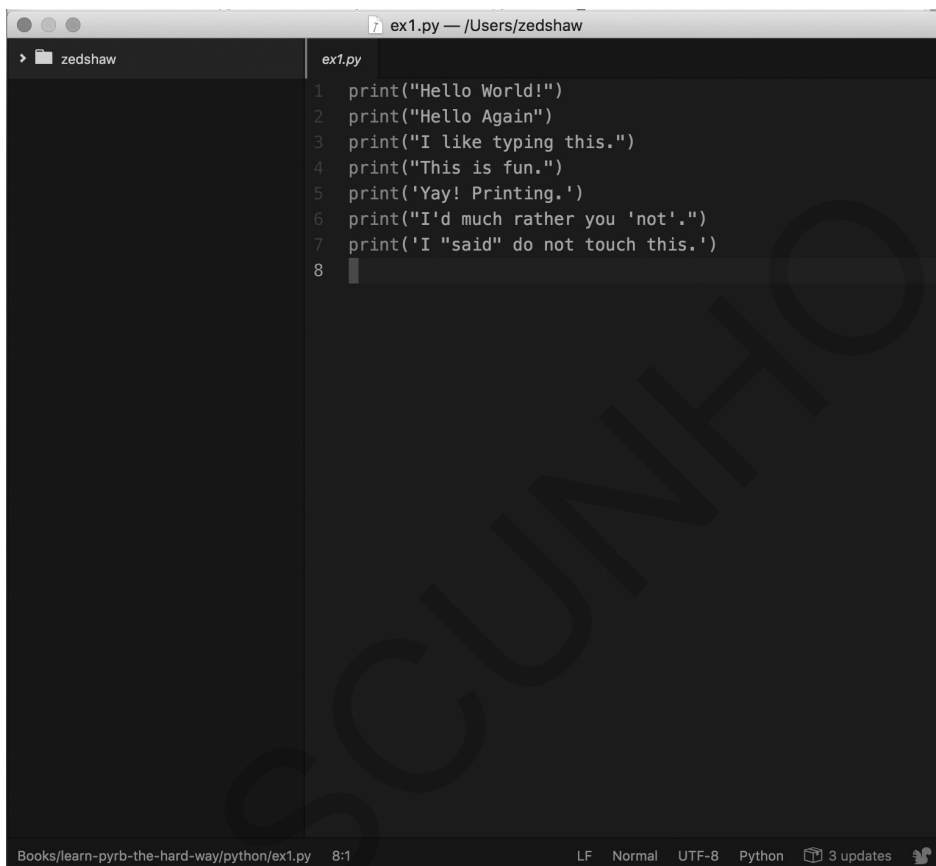
Você deve ter passado um bom tempo no Exercício 0, aprendendo a instalar um editor de texto, executar o editor, executar o Terminal e trabalhar com os dois. Se não fez isso, não continue. Você terá problemas. Esta é a única vez em que começarei um exercício com um aviso de que não deve pular ou colocar o carro na frente dos bois.

Digite o seguinte texto em um arquivo simples chamado `ex1.py`. O Python funciona melhor com arquivos terminando com `.py`.

`ex1.py`

```
1 print("Hello World!")
2 print("Hello Again")
3 print("I like typing this.")
4 print("This is fun.")
5 print('Yay! Printing.')
6 print("I'd much rather you 'not'.")
7 print('I "said" do not touch this.')
```

Seu editor de texto Atom deve ficar assim em todas as plataformas:



```
ex1.py
1 print("Hello World!")
2 print("Hello Again")
3 print("I like typing this.")
4 print("This is fun.")
5 print('Yay! Printing.')
6 print("I'd much rather you 'not'.")
7 print('I "said" do not touch this.')
8
```

Books/learn-pyrb-the-hard-way/python/ex1.py 8:1 LF Normal UTF-8 Python 3 updates

Não se preocupe se não for exatamente igual; deve ser parecido. Você pode ter um cabeçalho da janela um pouco diferente, talvez com cores diferentes e o lado esquerdo da janela Atom não informará “zedshaw”, mas mostrará o diretório usado para salvar seus arquivos. Todas essas diferenças não são um problema.

Quando criar o arquivo, lembre destes pontos:

1. Eu não digitei os números das linhas à esquerda. Eles são impressos no livro para eu poder falar sobre linhas específicas informando: “Veja a linha 5...” Você não digitará os números das linhas nos scripts do Python.
2. Tenho o `print` no início da linha e ele é exatamente igual ao que tenho em `ex1.py`. Exatamente significa exatamente, não quase igual. Todo caractere tem que corresponder para funcionar. A cor não importa, apenas os caracteres digitados.

No Terminal macOS ou (talvez) Linux, execute o arquivo digitando:

```
python3.6 ex1.py
```

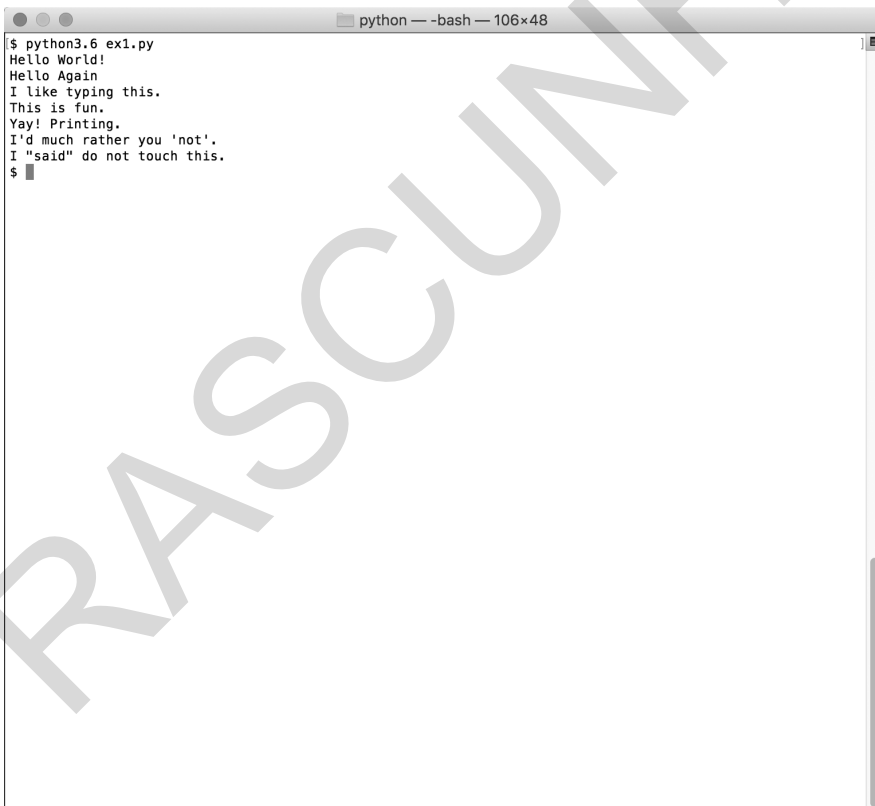
No Windows, sempre lembre de digitar `python` ao invés de `python3.6`, assim:

```
python ex1.py
```

Se você fez corretamente, deverá ver a mesma saída que tenho na seção *O que Você Deve Ver* deste exercício. Se não, fez algo errado. Não, o computador não errou.

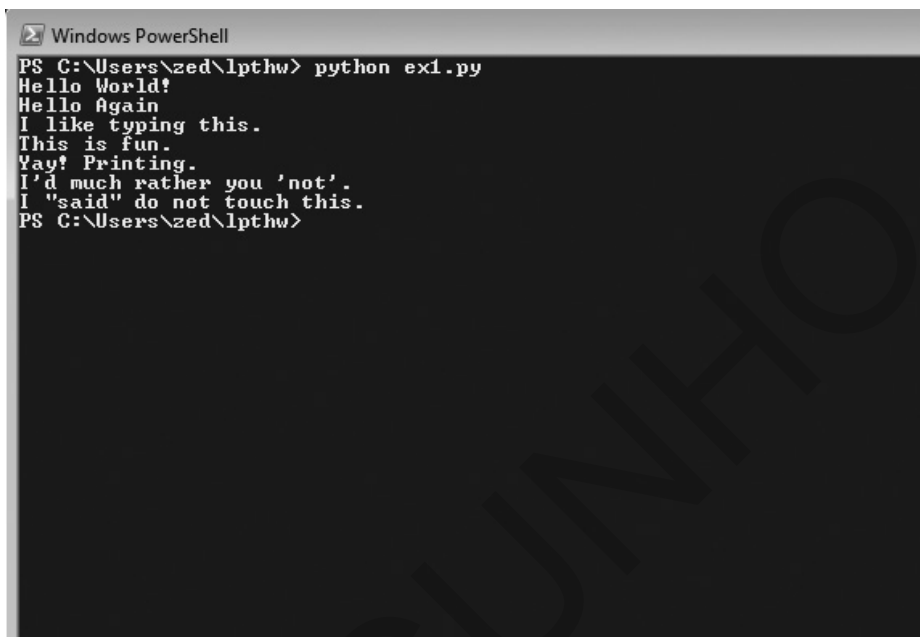
O que Você Deve Ver

No macOS, no Terminal, deverá ver isto:



```
python --bash-- 106x48
$ python3.6 ex1.py
Hello World!
Hello Again
I like typing this.
This is fun.
Yay! Printing.
I'd much rather you 'not'.
I "said" do not touch this.
$
```

No Windows, no PowerShell, deverá ver isto:

A screenshot of a Windows PowerShell window. The title bar says "Windows PowerShell". The command prompt shows the user running `python ex1.py` from the directory `C:\Users\zed\lpthw`. The output of the script is displayed line by line: `Hello World!`, `Hello Again`, `I like typing this.`, `This is fun.`, `Yay! Printing.`, `I'd much rather you 'not'.`, and `I "said" do not touch this.`. The prompt returns to `PS C:\Users\zed\lpthw>`.

```
PS C:\Users\zed\lpthw> python ex1.py
Hello World!
Hello Again
I like typing this.
This is fun.
Yay! Printing.
I'd much rather you 'not'.
I "said" do not touch this.
PS C:\Users\zed\lpthw>
```

É possível ver nomes diferentes antes do comando `python3.6 ex1.py`, mas o importante é que você digite o comando e veja se a saída é igual à minha.

Se houver um erro, será assim:

```
$ python3.6 python/ex1.py
File "python/ex1.py", line 3
    print("I like typing this.
          ^
SyntaxError: EOL while scanning string literal
```

É importante que você possa ler essas mensagens de erro porque cometerá muitos. Até eu cometo muitos erros. Vejamos linha por linha.

1. Executamos o comando no Terminal para rodar o script `ex1.py`.
2. O Python informa que o arquivo `ex1.py` tem um erro na linha 3 do `ex1.py`.
3. Ele imprime a linha de código para ser exibida.
4. Então, coloca um caractere `^` (circunflexo) para indicar onde está o problema. Notou que falta o caractere `"` (aspas duplas)?

5. Finalmente, ele imprime “SyntaxError” e informa algo sobre o que pode ser o erro. Em geral são muito enigmáticos, mas, se você copiar o texto em um mecanismo de busca, encontrará alguém que teve o mesmo erro e provavelmente conseguirá descobrir como corrigi-lo.

Exercícios Simulados

Os Exercícios Simulados têm coisas que você deve *tentar* fazer. Se não conseguir, pule e volte mais tarde. Para este exercício, experimente isto:

1. Faça o script imprimir outra linha.
2. Faça o script imprimir apenas uma das linhas.
3. Coloque um caractere # (cerquilha) no início de uma linha. O que ele fez? Tente descobrir.

De agora em diante não explicarei como cada exercício funciona, a menos que seja diferente.

AVISO! Uma “cerquilha” também é chamada de “hashtag”, “tralha”, “jogo da velha” e um monte de outros nomes. Escolha o seu preferido.

Perguntas Comuns dos Alunos

Estas são perguntas *reais* dos alunos ao fazer o exercício.

Posso usar o IDLE? Não, você deve usar o Terminal no macOS e o PowerShell no Windows, como é feito aqui. Se não souber usá-los, poderá ler o apêndice.

Como as cores são colocadas no editor? Salve seu arquivo primeiro como `.py`, por exemplo, `ex1.py`. Então, as cores aparecerão quando digitar.

Vejo SyntaxError: invalid syntax quando executo ex1.py. Provavelmente você está tentando executar o Python e digitar Python de novo. Feche o Terminal, inicie de novo e digite imediatamente apenas `python3.6 ex1.py`.

Vejo can't open file 'ex1.py': [Errno 2] No such file or directory. Você precisa estar no mesmo diretório do arquivo criado. Use o comando `cd` para ir primeiro para o diretório. Por exemplo, se você salvou o arquivo em `lpthw/ex1.py`, usaria `cd lpthw/` antes de tentar executar o `python3.6 ex1.py`. Se não sabe o que isso significa, vá para o apêndice.

Meu arquivo não é executado; o prompt volta sem nenhuma saída. Provavelmente, você pegou o código no meu arquivo `ex1.py` literalmente e achou que `print("Hello World!")` significava digitar apenas "Hello World!" no arquivo, sem `print`. O arquivo tem que ser *exatamente* igual ao meu.

RASCUNHO

Comentários e Cerquilhas

Os comentários são muito importantes em seus programas. São usados para informá-lo sobre a função de algo, assim como para desativar partes do programa, caso seja necessário removê-las temporariamente.

Veja como usar comentários no Python:

ex2.py

```
1  # Um comentário, assim você pode ler seu programa mais tarde.
2  # Qualquer coisa depois de # é ignorada pelo python.
3
4  print("I could have code like this.") # e o comentário depois é ignorado
5
6  # Você também pode usar um comentário para "desabilitar" um código:
7  # print("This won't run.")
8
9  print("This will run.")
```

De agora em diante, vou escrever o código assim. É importante que você entenda que as coisas não precisam ser literais. Sua tela e programa podem ser diferentes visualmente, o importante é o texto que você está escrevendo no arquivo do editor de texto. Na verdade, eu poderia trabalhar com qualquer editor de texto e os resultados seriam os mesmos.

O que Você Deve Ver

Sessão do Exercício 2

```
$ python3.6 ex2.py
I could have code like this.
This will run.
```

Novamente, não mostrarei as telas de todos os possíveis Terminais. Você precisa entender que o fragmento anterior não é uma tradução literal de como deve ser sua saída visualmente, mas o texto entre o primeiro \$ python3.6 ... e as últimas linhas \$ serão seu foco.

Exercícios Simulados

1. Descubra se você estava certo sobre o que o caractere # faz e verifique se sabe como ele é chamado (cerquilha ou hashtag).
2. Pegue o arquivo `ex2.py` e analise cada linha de trás para frente. Inicie na última linha e compare cada palavra na ordem inversa com o que você digitou.
3. Encontrou mais erros? Corrija.
4. Leia o que digitou acima em voz alta, inclusive dizendo o nome de cada caractere. Encontrou mais erros? Corrija.

Perguntas Comuns dos Alunos

Tem certeza que # é chamado de hashtag? Chamo de cerquilha porque é o único nome que nenhum país usa e funciona em todos eles. Todo país acha que o nome escolhido para esse caractere é o mais importante e único. Para mim, é muito arrogante e realmente todos devemos relaxar e focar as coisas mais importantes, como aprender o código.

Por que # em `print("Hi # there.")` não é ignorado? O símbolo # nesse código está dentro de uma string, portanto, ficará na string até que o caractere " final seja alcançado. As cerquilhas nas strings são consideradas caracteres, não comentários.

Como eu comento várias linhas? Coloque # na frente de cada uma

Não consigo descobrir como digitar um caractere # no teclado do meu país. Como faço isso? Alguns países usam a tecla ALT e combinações de outras teclas para imprimir caracteres que não pertencem ao seu idioma. Você terá que pesquisar online em um mecanismo de pesquisa para ver como digitá-lo.

Por que preciso ler o código de trás para frente? É um truque para fazer seu cérebro não se prender ao significado de cada parte do código, isso o faz processar cada parte com exatidão. É uma técnica de verificação de erros útil.