CAPÍTULO 02

2. Utilizando o interpretador Python

2.1. Chamando o interpretador

O interpretador Python é frequentemente instalado como `/usr/local/bin/python3.10` nas máquinas onde está disponível; adicionando `/usr/local/bin` ao caminho de busca da shell de seu Unix torna-se possível iniciá-lo digitando o comando:

```
python3.10
```

no shell. 1 Considerando que a escolha do diretório onde o interpretador está é uma opção de instalação, outros locais são possíveis; verifique com seu guru local de Python ou administrador do sistema local. (Por exemplo, `/usr/local/python` é um local alternativo popular.)

Em máquinas Windows onde você instalou Python a partir da Microsoft Store, o comando `python3.10` estará disponível. Se você tem o lançador py.exe instalado, você pode usar o comando `py`. Veja Excursus: Configurando variáveis de ambiente para outras maneiras de executar o Python.

Digitando um caractere de fim-de-arquivo (`Control-D` no Unix, `Control-Z` no Windows) diretamente no prompt força o interpretador a sair com status de saída zero. Se isso não funcionar, você pode sair do interpretador digitando o seguinte comando: `quit()`.

Os recursos de edição de linha do interpretador incluem edição interativa, substituição de histórico e complemento de código, em sistemas com suporte à biblioteca GNU Readline. Talvez a verificação mais rápida para ver se o suporte à edição de linha de comando está disponível é digitando `Control-P` no primeiro prompt oferecido pelo Python. Se for emitido um bipe, você terá a edição da linha de comando; veja Apêndice Edição de entrada interativa e substituição de histórico para uma introdução às combinações. Se nada acontecer, ou se `^P` aparecer na tela, a edição da linha de comando não está disponível; você só poderá usar backspace para remover caracteres da linha atual.

O interpretador trabalha de forma semelhante a uma shell de Unix: quando chamado com a saída padrão conectada a um console de terminal, ele lê e executa comandos interativamente; quando chamado com um nome de arquivo como argumento, ou com redirecionamento da entrada padrão para ler um arquivo, o interpretador lê e executa o *script* contido no arquivo.

Uma segunda forma de iniciar o interpretador é `python -c comando [arg] ...`, que executa uma ou mais instruções especificadas na posição *comando*, analogamente à opção de shell `-c`. Considerando que

comandos Python frequentemente têm espaços em branco (ou outros caracteres que são especiais para a shell) é aconselhável que o *comando* esteja dentro de aspas duplas.

Alguns módulos Python são também úteis como scripts. Estes podem ser chamados usando `python -m módulo [arg] ...` que executa o arquivo fonte do *módulo* como se você tivesse digitado seu caminho completo na linha de comando.

Quando um arquivo de script é utilizado, às vezes é útil executá-lo e logo em seguida entrar em modo interativo. Isto pode ser feito acrescentando o argumento `-i` antes do nome do script.

Todas as opções de linha de comando são descritas em Linha de comando e ambiente.

2.1.1. Passagem de argumentos

Quando são de conhecimento do interpretador, o nome do script e demais argumentos da linha de comando da shell são acessíveis ao próprio script através da variável `argv` do módulo `sys`. Pode-se acessar essa lista executando `import sys`. Essa lista tem sempre ao menos um elemento; quando nenhum script ou argumento for passado para o interpretador, `sys.argv[0]` será uma string vazia. Quando o nome do script for `'-'` (significando entrada padrão), o conteúdo de `sys.argv[0]` será `'-'`. Quando for utilizado `-c` *comando*, `sys.argv[0]` conterá `'-c'`. Quando for utilizado `-m` *módulo*, `sys.argv[0]` conterá o caminho completo do módulo localizado. Opções especificadas após `-c` *comando* ou `-m` *módulo* não serão consumidas pelo interpretador mas deixadas em `sys.argv` para serem tratadas pelo comando ou módulo.

2.1.2. Modo interativo

Quando os comandos são lidos a partir do console, diz-se que o interpretador está em modo interativo. Nesse modo ele solicita um próximo comando através do *prompt primário*, tipicamente três sinais de maior (`>>>`); para linhas de continuação do comando atual, o *prompt secundário* padrão é formado por três pontos (`...`). O interpretador exibe uma mensagem de boas vindas, informando seu número de versão e um aviso de copyright antes de exibir o primeiro prompt:

```
$ python3.10
Python 3.10 (default, June 4 2019, 09:25:04)
[GCC 4.8.2] on linux
Type "help", "copyright", "credits" or "license" for more
information.
>>>
```

Linhas de continuação são necessárias em construções multi-linha. Como exemplo, dê uma olhada nesse comando `if`:

```
>>> the_world_is_flat = True
>>> if the_world_is_flat:
...     print("Be careful not to fall off!")
...
Be careful not to fall off!
```

Para mais informações sobre o modo interativo, veja Modo interativo.