1. 课前甜点

如果您的工作主要是用电脑完成的，总有一天您会想能不能自动执行一些任务。比如，对大量文本文件执行查找、替换操作；利用复杂的规则重命名、重排序一堆照片文件；也可能您想编写一个小型数据库、或开发专用的图形界面应用，甚至是开发一个简单的游戏。

作为一名专业软件开发人员，您可能要处理 C/C++/Java 库，但编码、编译、测试、再编译这些开发流程太慢了；也许您正在给这些库开发测试套件，但总觉得这项工作真是枯燥乏味。又或许，您开发了个使用扩展语言的软件，却不想为这个软件专门设计一种新语言。

那么，Python 正好能满足您的需要。

你可以针对这些任务编写 Unix shell 脚本或 Windows 批处理文件，但 shell 脚本擅长的是移动文件和改变文本数据，而不适合编写 GUI 应用或游戏。 你可以编写 C/C++/Java 程序，但即使只完成一个初始版程序也需要耗费很长的开发时间。 Python 则更为简单易用，同时支持 Windows, macOS 和 Unix 操作系统，并能帮助你更快速地完成工作。

Python 虽然简单易用，但它可是真正的编程语言，提供了大量的数据结构，也支持开发大型程序，远超 shell 脚本或批处理文件；Python 提供的错误检查比 C 还多；作为一种“非常高级的语言”，它内置了灵活的数组与字典等高级数据类型。正因为配备了更通用的数据类型，Python 比 Awk，甚至 Perl 能解决更多问题，而且，很多时候，Python 比这些语言更简单。

Python 支持把程序分割为模块，以便在其他 Python 程序中复用。它还内置了大量标准模块，作为开发程序的基础 —— 您还可以把这些模块当作学习 Python 编程的实例。这些模块包括 I/O、系统调用、套接字，甚至还包括 Tk 图形用户界面工作套件。

Python 是一种解释型语言，不需要编译和链接，可以节省大量开发时间。它的解释器实现了交互式操作，轻而易举地就能试用各种语言功能，编写临时程序，或在自底向上的程序开发中测试功能。同时，它还是一个超好用的计算器。

Python 程序简洁、易读，通常比实现同种功能的 C、C++、Java 代码短很多，原因如下：

- 高级数据类型允许在单一语句中表述复杂操作；
- 使用缩进，而不是括号实现代码块分组；
- 无需预声明变量或参数。

Python “可以扩展”：会开发 C 语言程序，就能快速上手为解释器增加新的内置函数或模块，不论是让核心程序以最高速度运行，还是把 Python 程序链接到只提供预编译程序的库（比如，硬件图形库）。只要下点功夫，就能把 Python 解释器和用 C 开发的应用链接在一起，用它来扩展和控制该应用。

顺便提一句，本语言的命名源自 BBC 的 “Monty Python 飞行马戏团”，与爬行动物无关（Python 原义为“蟒蛇”）。欢迎大家在文档中引用 Monty Python 小品短篇集，多多益善！

现在，您已经对 Python 跃跃欲试，想深入了解一些细节了吧。要知道，学习语言的最佳方式是上手实践，建议您边阅读本教程，边在 Python 解释器中练习。

下一章介绍解释器的用法。这部分内容有些单调乏味，但对上手实践后面的例子来说却至关重要。

本教程的其他部分将利用各种示例，介绍 Python 语言、系统的功能，开始只是简单的表达式、语句和数据类型，然后是函数、模块，最后，介绍一些高级概念，如，异常、用户定义的类等功能。