构建工具：Python

Python 编程语言附带一个名为 的包管理器 `pip` 。找到提供它的包并安装它（**提示**：我们如何在 C 练习中找到丢失的库？）。

我们将练习安装槲寄生模块，它将 markdown 渲染为 HTML。

- 在 python 中，尝试该行 `import mistletoe` 并注意您得到了 `ModuleNotFoundError: No module named 'mistletoe'` .
- 再次退出 python (Control-D) 并尝试 `pip3 install --user mistletoe` . 您应该收到一条成功消息（可能还有一条警告，如下所述）。
- 再次打开 python 并重复 `import mistletoe` 。这不会产生任何输出，因此模块已加载。

创建一个小型示例 Markdown 文件，如下所示， `hello.md` 例如：

```
# Markdown Example

Markdown is a *markup* language.
```

再次打开 python 并输入以下内容。您需要缩进最后一行（通常是四个空格）并在末尾按 ENTER 两次。

```
import mistletoe
with open('hello.md', 'r') as file:
    mistletoe.markdown(file)
```

这应该打印渲染为 HTML 的 markdown，例如

```
<h1>Markdown Example</h1>\n<p>Markdown is a <em>markup</em> language.</p>
```

Advanced note

Python 版本 3 于 2008 年发布，与 Python 2 相比有一些语法更改（ `print "hello world"` 成为 `print("hello world")` ）。版本 2 现已被视为已弃用；但这种转变是*漫长且极其痛苦的*，因为改变 print 语句之类的语法会导致大量代码破坏，而且很多人宁愿不修复他们的代码，而只是保留安装旧版本的 Python。

因此，当我们处理这个问题时，系统通常会 `python2` 为旧版本和 `python3` 新版本安装多个版本的Python（即使如此，这些版本通常是指向特定颠覆的符号链接，例如 `python2.6` ），然后 `python` 成为任何内容的符号链接您的操作系统被视为“受支持”的版本。

不同的操作系统绝对有不同版本的 Python（MacOS 尤其令人震惊的是，它使用 Python 2 的时间远远超过了必要的时间），因此需要一个解决方案，因为这只会在操作系统设计者争吵时破

坏事物。

解决方案是，对于*大多数*依赖项（编译库除外），我们通常使用编程语言自己的包管理器并忽略操作系统提供的内容。对于 Python 来说，这意味着 `pip` （有时称为 `pip3` or `pip2` ）。

有时您会看到一些信息告诉您安装软件包 `sudo pip install` ，但不要这样做！它最终会严重破坏事情。您可以在没有 sudo 的情况下使用 pip，通过传递 `--user` 将软件包安装到主目录 (`~/.local`) 中的文件夹中的选项，而不是 `/usr` 通常需要 root 权限的文件夹中。

有时您仍然需要通过操作系统包管理器安装包（ `numpy` 这 `scipy` 很常见，因为它们依赖于大量 C 代码，因此安装起来很痛苦， `pip` 因为您必须手动修复库路径和依赖项）但是一般来说，尽量避免它。

用于管理操作系统的Python应该由系统设计者运行；用于开发工作的 Python 应该由您管理。两人永远不会见面。

西皮

我们经常用于 `scipy` 统计，所以你也可以安装它。不幸的是， `pip` 这对你没有帮助，因为 scipy 依赖于 C 库来实现快速线性代数。您可以去安装所有依赖项（如果您需要它的特定版本，您可能必须这样做），但事实证明 Debian 已将其全部打包为系统包：尝试使用 `apt search scipy` .

以下命令通过从均值 200、标准差 10 的正态分布中采样 5 次来显示安装是否正确：

```
from scipy.stats import norm
norm(loc=200, scale=10).rvs(5)
```

这应该打印一个由五个值组成的数组，这些值与 200 相差不太远（准确地说，大约 95% 的置信度它们将在 180 到 220 之间 - 稍后在数学 B 中对此进行更多介绍）。

避免使用 sudo

如果您需要安装库，您可能会想通过使用为所有用户安装它们， `sudo pip` 但这可能会导致痛苦！如果您更改系统库并且系统中的某些内容取决于库的特定版本，那么可能会导致可怕的损坏和无法正常工作（特别是在像 Mac OS 这样的操作系统上，这些操作系统往往更新库的频率较低）。

Python 附带了一种名为venv的机制，它可以让您创建一个由用户拥有的虚拟 python 安装：您可以更改其中的库 `sudo` ，而不必担心弄乱您的主机系统。阅读文档并习惯使用它——以后它将为您省去很多痛苦！

Advanced note

`pip freeze | tee requirements.txt` 将列出您使用的所有软件包以及它们的版本，并将它们保存在名为 `requirements.txt` .

`pip install -r requirements.txt` 将再次安装它们！

这使得确保查看您的代码的人拥有所有正确的依赖项变得*非常容易，而无需列出安装这些库*的列表（并且将使任何必须标记您的代码的人感到高兴并且更倾向于给您标记）。