1.2.7 表示字符集的特殊字符

我们还提到有一些特殊字符能够表示字符集。与使用“0-9”这个范围表示十进制数相比，可以简单地使用 d 表示匹配任何十进制数字。另一个特殊字符（\w）能够用于表示全部字母数字的字符集，相当于[A-Za-z0-9_]的缩写形式，\s 可以用来表示空格字符。这些特殊字符的大写版本表示不匹配；例如，\D 表示任何非十进制数（与[^0-9]相同），等等。

使用这些缩写，可以表示如下一些更复杂的示例。

正则表达式模式	匹配的字符串
\w+-\d+	一个由字母数字组成的字符串和一串由一个连字符分隔的数字
[A-Za-z]\w*	第一个字符是字母；其余字符（如果存在）可以是字母或者数字（几乎等价于 Python 中的有效标识符［参见练习］）
\d{3}-\d{3}-\d{4}	美国电话号码的格式，前面是区号前缀，例如 800-555-1212
\w+@\w+\.com	以 *XXX@YYY*.com 格式表示的简单电子邮件地址

1.2.8 使用圆括号指定分组

现在，我们已经可以实现匹配某个字符串以及丢弃不匹配的字符串，但有些时候，我们可能会对之前匹配成功的数据更感兴趣。我们不仅想要知道整个字符串是否匹配我们的标准，而且想要知道能否提取任何已经成功匹配的特定字符串或者子字符串。答案是可以，要实现这个目标，只要用一对圆括号包裹任何正则表达式。

当使用正则表达式时，一对圆括号可以实现以下任意一个（或者两个）功能：

- 对正则表达式进行分组；
- 匹配子组。

关于为何想要对正则表达式进行分组的一个很好的示例是：当有两个不同的正则表达式而且想用它们来比较同一个字符串时。另一个原因是对正则表达式进行分组可以在整个正则表达式中使用重复操作符（而不是一个单独的字符或者字符集）。

使用圆括号进行分组的一个副作用就是，匹配模式的子字符串可以保存起来供后续使用。这些子组能够被同一次的匹配或者搜索重复调用，或者提取出来用于后续处理。1.3.9 节的结尾将给出一些提取子组的示例。

为什么匹配子组这么重要呢？主要原因是在很多时候除了进行匹配操作以外，我们还想要提取所匹配的模式。例如，如果决定匹配模式\w+-\d+，但是想要分别保存第一部分的字母和第二部分的数字，该如何实现？我们可能想要这样做的原因是，对于任何成功的匹配，我们可能想要看到这些匹配正则表达式模式的字符串究竟是什么。

如果为两个子模式都加上圆括号，例如(\w+)-(\d+)，然后就能够分别访问每一个匹配子组。我们更倾向于使用子组，这是因为择一匹配通过编写代码来判断是否匹配，然后

执行另一个单独的程序（该程序也需要另行创建）来解析整个匹配仅仅用于提取两个部分。为什么不让 Python 自己实现呢？这是 re 模块支持的一个特性，所以为什么非要重蹈覆辙呢？

正则表达式模式	匹配的字符串
\d+(\.\d*)?	表示简单浮点数的字符串；也就是说，任何十进制数字，后面可以接一个小数点和零个或者多个十进制数字，例如“0.004”、“2”、“75.”等
(Mr?s?\.)?[A-Z][a-z]*[A-Za-z-]+	名字和姓氏，以及对名字的限制（如果有，首字母必须大写，后续字母小写），全名前可以有可选的“Mr.”、“Mrs.”、“Ms.”或者“M.”作为称谓，以及灵活可选的姓氏，可以有多个单词、横线以及大写字母

1.2.9　扩展表示法

我们还没介绍过的正则表达式的最后一个方面是扩展表示法，它们是以问号开始（?...）。我们不会为此花费太多时间，因为它们通常用于在判断匹配之前提供标记，实现一个前视（或者后视）匹配，或者条件检查。尽管圆括号使用这些符号，但是只有（?P<name>）表述一个分组匹配。所有其他的都没有创建一个分组。然而，你仍然需要知道它们是什么，因为它们可能最适合用于你所需要完成的任务。

正则表达式模式	匹配的字符串
(?:\w+\.)*	以句点作为结尾的字符串，例如“google.”、“twitter.”、“facebook.”，但是这些匹配不会保存下来供后续的使用和数据检索
(?#comment)	此处并不做匹配，只是作为注释
(?=.com)	如果一个字符串后面跟着“.com”才做匹配操作，并不使用任何目标字符串
(?!.net)	如果一个字符串后面不是跟着“.net”才做匹配操作
(?<=800-)	如果字符串之前为“800-”才做匹配，假定为电话号码，同样，并不使用任何输入字符串
(?<!192\.168\.)	如果一个字符串之前不是“192.168.”才做匹配操作，假定用于过滤掉一组 C 类 IP 地址
(?(1)y|x)	如果一个匹配组 1（\1）存在，就与 y 匹配；否则，就与 x 匹配

1.3　正则表达式和 Python 语言

在了解了关于正则表达式的全部知识后，开始查看 Python 当前如何通过使用 re 模块来支持正则表达式，re 模块在古老的 Python 1.5 版中引入，用于替换那些已过时的 regex 模块和 regsub 模块——这两个模块在 Python 2.5 版中移除，而且此后导入这两个模块中的任意一个都会触发 ImportError 异常。

re 模块支持更强大而且更通用的 Perl 风格（Perl 5 风格）的正则表达式，该模块允许多个线程共享同一个已编译的正则表达式对象，也支持命名子组。