


1 字符串之前加 r 表示不转义

1 isinstance(name, type) 判断是否是type类型的实例  
2 BIF (built-in functions) 内建函数，例如 isinstance，dir，input，print  
3 python3默认递归深度不超过100  
4



### BULLET POINTS

- 从命令行或在IDLE中运行Python 3。
- 标识符是指示数据对象的名字。标识符没有“类型”，不过标识符所指示的数据对象有类型。
- print() BIF会在屏幕上显示一个消息。
- 列表是一个数据集合，数据项之间用逗号分隔，整个列表用中括号包围。
- 列表就像是“打了激素”的数组。
- 可以用BIF处理列表，另外列表还支持一组列表方法。
- 列表可以存放任意数据，而且数据可以是混合类型。列表还可以包含其他列表。
- 列表可以随需要伸缩。数据使用的所有内存都由Python为你管理。
- Python使用缩进将语句归组在一起。
- len() BIF会提供某个数据对象的长度，或者统计一个集合中的项数，如列表中的项数。
- for循环允许迭代处理一个列表，这通常比使用一个等价的while循环更方便。
- 可以利用if... else...语句在代码中完成判定。
- isinstance() BIF会检查一个标识符是否指示某个指定类型的数据对象。
- 使用def来定义一个定制函数。

5

PyPI读作“pie-pie”。



Python包索引（Python Package Index, PyPI）为Internet上的第三方Python模块提供了一个集中的存储库。准备好之后，就可以使用PyPI来发布你的模块，从而使你的代码可供其他人使用。你的模块已经准备就绪，不过还有一个重要的补充。

6 """ 块注释代码  
# 行注释代码

7



## BULLET POINTS

- 模块是一个包含Python代码的文本文件。
- 发布工具允许将模块转换为可共享的包。
- `setup.py`程序提供了模块的元数据，用来构建、安装和上传打包的发布。
- 使用`import`语句可以将模块导入到其他程序中。
- Python中的各个模块提供了自己的命名空间，使用`module.function()`形式调用模块的函数时，要用命名空间名限定函数。
- 使用`import`语句的`from module import function`形式可以从一个模块将函数专门导入到当前命名空间。
- 使用`#`可以注释掉一行代码，或者为程序增加一个简短的单行注释。
- 内置函数（built-in functions, BIF）有自己的命名空间，名为`__builtins__`，这会自动包含在每一个Python程序中。
- `range()` BIF可以与`for`结合使用，从而迭代固定次数。
- 包含`end=''`作为`print()` BIF的一个参数会关闭其默认行为（即在输入中自动包含换行）。
- 如果为函数参数提供一个缺省值，这个函数参数就是可选的。

8

以下是IDLE编辑窗口中的代码。注意`split()`方法的额外参数。

```
data = open('sketch.txt')

for each_line in data:
    (role, line_spoken) = each_line.split(':', 1)
    print(role, end='')
    print(' said: ', end='')
    print(line_spoken, end='')

data.close()
```

Ln: 11 Col: 0

这个额外的参数控制“`split()`”如何分解。

9

# 先尝试，然后恢复

并非增加额外的代码和逻辑来阻止不好的事情发生，Python的异常处理机制允许错误出现，但监视它的发生，然后给你一个机会来恢复。

在正常的控制流期间，Python尝试运行你的代码，如果没有任何问题，代码会继续正常执行。在异常控制流期间，Python先尝试运行你的代码，如果发现有问题，就会执行恢复代码，然后继续正常执行你的代码。

10

**try:**

“try”和“except”  
都是Python关键字。

你的代码（可能导致一个运行时错误）

**except:**

错误恢复代码

11

**问：**有个问题困扰我很久了。split()方法执行时，它传回一个列表，不过目标标识符包围在小括号之间，而不是中括号，这真的是一个列表吗？

**答：**你观察得很仔细。Python中实际上有两种类型的列表：一种是可以改变的列表（用中括号包围），另一种一旦创建就不能改变（用小括号包围）。后者是一种不可变列表，更常见的称呼是元组（tuple）。可以认为元组等同于列表，不过有一点区别：一旦创建，元组中的数据在任何情况下都不能改变。还有一种理解，可以认为元组是一个“常量列表”，在Head First，我们把“tuple”拼作与“couple”押韵，也有人把“tuple”拼作与“rupal”押韵。至于哪一种正确，并没有一个明确的共识，所以你可以选择并采用任意一种方式。

12

## 放过错误

对于这个数据（和程序），最好能忽略不符合期望格式的数据行。如果split()方法调用导致一个异常，可以报告这是一个错误并使用pass继续执行代码。

如果遇到这样一种情况，也就是希望你提供一些代码，但是并不需要具体做什么，就可以使用Python的pass语句（可以把它认为是空语句或null语句）。

13

- `split()`方法可以将一个字符串分解为一个子串列表。
- Python中不可改变的常量列表称为元组（`tuple`）。一旦将列表数据赋至一个元组，就不能再改变。元组是不可改变的。
- 数据不符合期望的格式时会出现 `ValueError`。
- 数据无法正常访问时会出现 `IOError`（例如，可能你的数据文件已经被移走或者重命名）。
- `help()` BIF允许你在IDLE shell中访问Python的文档。
- `find()`方法会在一个字符串中查找一个特定子串。
- `not`关键字将一个条件取反。
- `try/except`语句提供了一个异常处理机制，从而保护可能导致运行时错误的某些代码行。
- `pass`语句就是Python的空语句或 `null`语句，它什么也不做。

14 `data_line.strip()` 去除空格

15

**问：** 还有其他Python数据类型也是不可变的吗？

**答：** 是的，很多都是不可变的。比如说元组，这是一个不可变的列表。另外，所有数值类型都是不可变的。

**问：** 除了具体了解每种类型，我怎么才能知道一个类型是不可变的呢？

**答：** 不用担心：你会知道的。如果你想改变一个不可变的值，Python会产生一个 `TypeError` 异常。

**问：** 当然了：会出现一个异常。异常在Python中无处不在，不是吗？

**答：** 确实是，正是异常才让这个世界变幻莫测。

不过做这个修改后试图运行代码时，会生成另一个异常：

```
Traceback (most recent call last):
```

```
File "<pysHELL#18>", line 5, in <module>
    print('File error:' + err)
```

```
TypeError: Can't convert 'IOError' object to str implicitly
```

唉呀！又有一个异常，这一次是一个“TypeError”。

这一次你的错误消息根本没有出现。看来异常对象和字符串类型不兼容，所以尝试把异常对象与字符串相连接会带来问题。可以使用`str()` BIF把异常对象转换（或强制转换）为字符串：

```
except IOError as err:
```

```
    print('File error: ' + str(err))
```

使用“`str()`” BIF要求异常对象表现为一个字符串。

现在，做最后这个修改后，代码终于有了你期望的表现：

```
File error: [Errno 2] No such file or directory: 'missing.txt'
```

现在你会得到一个特定的错误消息指出到底哪里出了问题。

17

```
try:
```

```
    with open('man_data.txt', 'w') as man_file:
```

```
        print(man, file=man_file)
```

```
    with open('other_data.txt', 'w') as other_file:
```

```
        print(other, file=other_file)
```

```
except IOError as err:
```

```
    print('File error: ' + str(err))
```

使用两个“with”语句重写代码，这一次不包括“finally”组。

或者将两个“`open()`”调用合并到一个“with”语句中。

注意逗号的使用。

```
with open('man_data.txt', 'w') as man_file, open('other_data.txt', 'w') as other_file:
```

```
    print(man, file=man_file)
```

```
    print(other, file=other_file)
```

18

## 用dump保存，用load恢复

使用pickle很简单：只需导入所需的模块，然后使用dump()保存数据，以后某个时间使用load()恢复数据。处理腌制数据时的唯一要求是，必须以二进制访问模式打开这些文件：

一定要记住导入“pickle”模块。

要保存数据，使用“dump()”。

将恢复后的数据赋至一个标识符。

```
import pickle
...
with open('mydata.pickle', 'wb') as mysavedata:
    pickle.dump([1, 2, 'three'], mysavedata)
...
with open('mydata.pickle', 'rb') as myrestoredata:
    a_list = pickle.load(myrestoredata)
print(a_list)
```

“b”告诉Python以二进制模式打开数据文件。

使用“load()”从文件恢复数据。

一旦数据回到程序中，就可以像任何其他数据对象一样处理了。

19

**问：**之前调用print\_lol()时，你只提供了两个参数，而函数签名要求提供4个参数。这是怎么回事？

**答：**在代码中调用一个Python函数时，有很多选择，特别是函数为一些参数提供了缺省值时，如果使用位置参数，参数在函数调用中的位置会指示将哪个数据赋至哪个参数。如果函数还有一些提供了缺省值的参数，则不必操心一定为位置参数赋值。

**问：**喂，你把我完全说糊涂了。能解释一下吗？

**答：**请考虑print()，它的签名如下：print(value, sep=' ', end='\n', file=sys.stdout)。默认地，这个BIF会显示到标准输出（屏幕），因为它有一个名为file的参数，其缺省值为sys.stdout。这个file参数是第4个位置参数。不过，如果想把数据发送到屏幕以外的其他地方，就不需要（也不必）为第2个和第3个位置参数提供值。它们也有缺省值，所以只有在缺省值不是你想要的值时才需要为它们提供值。如果你想做的只是将数据发送到一个文件，可以这样调用print() BIF：print("Dead Parrot Sketch", file='myfavmonty.txt')，第4个位置参数使用你指定的值，而其他位置参数使用其缺省值。在Python中，不仅BIF采用这种方式，你的定制函数也支持这种机制。

20



## BULLET POINTS

- `strip()`方法可以从字符串去除不想要的空白符。
- `print()` BIF的`file`参数控制将数据发送/保存到哪里。
- `finally`组总会执行，而不论`try/except`语句中出现什么异常。
- 会向`except`组传入一个异常对象，并使用`as`关键字赋至一个标识符。
- `str()` BIF可以用来访问任何数据对象（支持串转换）的串表示。
- `locals()` BIF返回当前作用域中的变量集合。
- `in`操作符用于检查成员关系。
- “+”操作符用于字符串时将联接两个字符串，用于数字时则会将两个数相加。
- `with`语句会自动处理所有已打开文件的关闭工作，即使出现异常也不例外。`with`语句也使用`as`关键字。
- `sys.stdout`是Python中所谓的“标准输出”，可以从标准库的`sys`模块访问。
- 标准库的`pickle`模块允许你容易而高效地将Python数据对象保存到磁盘以及从磁盘恢复。
- `pickle.dump()`函数将数据保存到磁盘。
- `pickle.load()`函数从磁盘恢复数据。

21 sort 就地排序 sorted 复制排序

22

下面作为列表推导完成同样的功能，这包括创建一个新列表，为此要指定将应用到一个现有列表中各个数据项的转换。

选择要使用的目标标识符（与常规的迭代类似）。

```
clean_mikey = [sanitize(each_t) for each_t in mikey]
```

创建新列表……

……为此要指定一个转换……

……应用于各个数据项……

……一个现有列表中。

有意思的是，在这里转换已经缩减为只有一行代码。另外，不再需要指定使用`append()`方法，因为这个动作已经隐含在列表推导中。很棒吧？

23

当然，如果需要，转换还可以是一个函数链：

```
>>> clean = [float(sanitize(t)) for t in ['2-22', '3:33', '4.44']]
>>> clean
[2.22, 3.33, 4.44]
```

还支持合并对数据项的转换！

24

# 用集合删除重复项

除了列表，Python还提供了集合数据结构，它的表现类似于你在数学课上学到的集合。

Python中集合最突出的特性是集合中的数据项是无序的，而且不允许重复。如果试图向一个集合增加一个数据项，而该集合中已经包含有这个数据项，Python就会将其忽略。

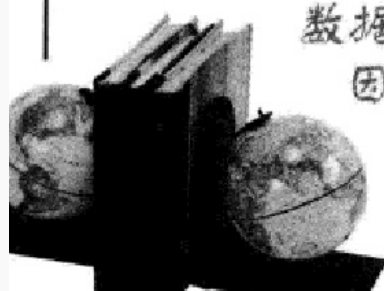
使用`set()` BIF创建一个空集合，这是工厂函数的一个例子：

创建一个新的空集合，  
并赋至一个变量。

`distances = set()`

## Scholar's Corner

**工厂函数：**工厂函数用于创建某种类型的新的数据项。例如，“`set()`”就是一个工厂函数，因为它会创建一个新的集合。在真实世界中，工厂会生产产品，这个概念因此而得名。







## BULLET POINTS

- `sort()` 方法可以在原地改变列表的顺序。
- `sorted()` BIF通过提供复制排序可以对几乎任何数据结构排序。
- 向`sort()`或`sorted()`传入`reverse=True`可以按降序排列数据。
- 如果有以下代码：

```
new_l = []  
for t in old_l:  
    new_l.append(len(t))
```

使用列表推导重写这个代码，可以写作：

```
new_l = [len(t) for t in old_l]
```

- 要访问一个列表中的多个数据项，可以使用分片。例如：  

```
my_list[3:6]
```
- 这会访问列表中从索引位置3直到（但不包括）索引位置6的列表项。
- 使用`set()`工厂方法可以创建一个集合。