第3章 -条件控制 (if) 与 (循环控制 for 和while)

作者： 何吉波博士，优视眼动科技公司创始人，hejibo@usee.tech, http://www.usee.tech

2017年以来，由于Google Alpha Go战胜了世界围棋冠军，人工智能变得非常热门。人们通常会担心机器或者人工智能会挑战人类。其实，我们人类胜于机器的地方在于创新能力和解决陌生模糊问题的能力。而机器代码的主要长处在于存储、循环和条件判断。 机器可以存储的数据量远远超过人类脑力可以存数的数据。机器的另外一个特长就是，只给你给机器电吃，它就可以一直做重复的事情，而且毫无怨言。如果我们让员工经常完成机器重复的工作，员工通常会抗议的。因此，就让我们学会编程，把这些重复的工作交给机器，而不是交给我们的研究助手或者本科生吧。请善待我们的研究生和研究助手。尤其是当您没有足够的钱雇佣研究助手时，学会本节的条件控制 (if) 与 (循环控制 for 和while)，对您就非常关键了。

比如，您期望计算一个眼动追踪(Eye Tracking)研究的被试的平均反应时。我们首选要过滤掉非常短的注视时间，比如小于80毫秒的注视，都应该被过滤掉。这种数据过滤操作就应该使用条件控制 (if)语句。然后我们要遍历所有的过滤后的注视时间，这对应的就是循环控制语句for 或者while。 最后使用numpy计算注视的平均时间。

条件控制 (if)

条件控制就是当条件成立时，执行一段代码块，否则不执行。 条件控制的首先就是进行条件判断。这就需要用到我们的在第二章讲到的逻辑变量 (Booleans)的知识了。 请朋友们复习一下这一小节。

条件控制有三种主要变式，1. if, 2. if else和3. if elif else。Python语言的代码块，如条件控制，循环控制，函数等，都是统括:以及缩进格式来控制代码块的，而不是使用Java或者C语言的{}来代码一个代码块。因此，如果您们得知一个程序员的制表符(TAB按键)坏了，或者他的制表符按钮非常亮，没有灰尘，这就说明

，他很有可能是一个用功的Python程序员。因为我们Python程序员经常按制表符(TAB按键)来进行代码的缩进。

if

最简单的if语句如下：

```
if RT <100:print "outlier"

if fixationDuration<80:
    print "fixation is too small, we shouyld delete
it"
    print fixationDuration
```

当条件成立时只需要执行一行代码，可以将这行代码与条件判断放在同一行，如上例中的if RT <100:print "outlier"。如果条件判断成立时要执行多行代码，则需要换行和缩进。

if else

if else语句用于条件成立时执行段代码，不成立时执行另外一段代码的情况。示例如下：

```
if fixationDuration <80:
    print "fixation duration is too short"
else:
    print "fixation duration is acceptable"
    print fixationDuration
```

if elif else

if elif else用于有两个及以上条件的判断。很多语言有switch case这样类似的根据案例来执行一段语句。Python没有switch case这样类似的结构。当我们需要

做多重条件判断时，我们就使用if elif else。我们可以有多个elif。示例代码如下：

```
dateOfWeek=1
if dateOfWeek ==1:
    print "Today is Monday"
elif dateOfWeek ==2:
    print "Today is Tuesday"
elif dateOfWeek ==3:
    print "Today is Wednesday"
elif dateOfWeek ==4:
    print "Today is Thursday"
elif dateOfWeek ==5:
    print "Today is Friday"
elif dateOfWeek ==6:
    print "Today is Saturday"
elif dateOfWeek ==7:
    print "Today is Sunday"
else:
    print "Please enter a valid number for
dateOfWeek."
```

循环控制 （for)

我们经常需要让电脑循环计算，直到完成某个条件。Python主要有两种循环控制方法，for 和while。for 主要是遍历一个列表或者有限的元素条件，通常我们可以知道要遍历训练的次数。而while是运行次数可能不固定，只要循环条件满足，就会一直执行下去。

一个实验的运行次数，或者被试数目通常是固定的，而且预先可以知道数目的。因此，我们要分析所有的被试的数据时，我们通常就使用for循环语句。

例如，在第2章的字典小节处，我们提到了使用字典来存储被试的反应时和准确性。如果我们要计算所有的被试的平均反应时。当一个操作设计到”所有“或者”遍历“，或者”循环“，我们就知道我们需要使用循环语句了。我们知道，对于一个字

典，可以使用data.keys()函数来找到所有的检索值。对于for 循环语句，我们就可以使用for subjectID in data.keys():这样的方式去遍历所有的检索值。其中，subjectID对应的就是字典的一个检索值，也就是下例的'subject-1'，'subject-2'和'subject-3'。进一步的，我们知道如何根据字典的检索值查看字典的值。那么data[subjectID]对应的就是一个被试的数据。进一步的，data[subjectID]["RT"]就是一个被试的反应时数据。我们使用RTList.extend(data[subjectID]["RT"])的方式，将所有的反应时数据存放在RTList这个列表中。最后，我们可以通过第二章讲解的numpy.mean(RTList),numpy.std(RTList)得到反应时的平均值和标准差。

与if条件语句类似，for 循环语句也使用:和缩进来表示一个代码块。相同缩进程度的的代码处于同一个代码块中。读者朋友们应该非常注意，缩进算Python语言的重要语法部分。 如下面所示，print "mean and standard deviation of RT:",numpy.mean(RTList),numpy.std(RTList)这行代码只运行了一次，因为它与for 循环不在同一个缩进层级。与之对比的，print "mean and standard deviation of RT:", numpy.mean(RTListIndividual),numpy.std(RTListIndividual)会运行三次，因为它与for 循环在同一个缩进层级。

```
import numpy
data = {}
data['subject-1'] = {'RT':[300, 256, 35], 'acc':[1,
1, 0]}
data['subject-2'] = {'RT':[400, 512, 100009], 'acc':
[1, 0, 1]}
data['subject-3'] = {'RT':[732, 542, 839], 'acc':[1,
1, 1]}
print data

# find all subject data information using the for
loop.
RTList = []
for subjectID in data.keys():
    print data[subjectID]["RT"]
    RTList.extend(data[subjectID]["RT"])
```

```
# calculate mean and std at group level. This is only
calculated once as it is out of the indention level
of the for group.
print "mean and standard deviation of
RT:",numpy.mean(RTList),numpy.std(RTList)

print "-----------section separator--------------"
# demostrate indention is part of the grammar of
Python
RTListIndividual = []
for subjectID in data.keys():
    print data[subjectID]["RT"]
    RTListIndividual.extend(data[subjectID]["RT"])
    # calculate mean and std at individual
participant level.This is calculated three times as
it is within the same indention level of the for
group.
    print "mean and standard deviation of RT:",
numpy.mean(RTListIndividual),numpy.std(RTListIndividu
al)
```

请读者朋友们结合上面的代码和”图3.1. for 循环示例，及缩进效果展示“仔细了解for 循环和缩进的重要性、

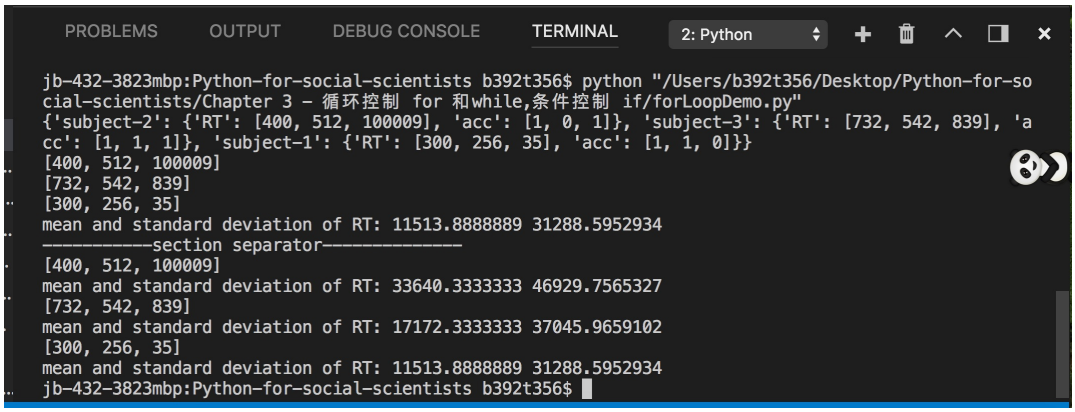

图3.1. for 循环示例，及缩进效果展示

循环控制 (while)

while也可以实现循环。 与for 循环通常用于遍历一个已经元素集不同，while通常用于在一定条件下执行语句块，直到条件不满足。

例如，我们知道我们的实验一共暂时只有3个被试。我们就可以用subjectID这个变量来计数。当subjbectID小于4时，我们都去寻找subjectID的数据用于计算。当subjbectID大于或者等于4时，我们就终止while循环，结束任务。如下们的代码块所示。while subjectIndex<4:表示为循环执行条件。对于while循环语句，一定要对循环条件中用到的变量进行值更新，如subjectIndex=subjectIndex+1，否则while 循环将永远执行下去，造成死循环。下面的while 循环和上方的for 循环是等价的。

```
import numpy
data = {}
data['subject-1'] = {'RT':[300, 256, 35], 'acc':[1,
1, 0]}
data['subject-2'] = {'RT':[400, 512, 100009], 'acc':
[1, 0, 1]}
data['subject-3'] = {'RT':[732, 542, 839], 'acc':[1,
1, 1]}
print data

# find all subject data information using the for
loop.
subjectIndex=1
RTList = []
while subjectIndex<4:
    key = 'subject-%d'%subjectIndex
    print data[key]["RT"]
    RTList.extend(data[key]["RT"])
    subjectIndex=subjectIndex+1

# calculate mean and std at group level. This is only
calculated once as it is out of the indention level
of the for group.
print "mean and standard deviation of
RT:",numpy.mean(RTList),numpy.std(RTList)
```

循环中断与继续 (break与continue)

上面的while循环条件while subjectIndex<4:与while并列写在了一起。另外一种操作方式是while处没有循环条件，而是永远成立的条件，比如while True:，而将循环退出条件放在代码块中。这时，我们就要用到if条件语句和break。当if条件满足时，我们就用break退出while 循环。与上面for和while等效的代码，也可以用通过if和break重写为如下：

```
import numpy
data = {}
data['subject-1'] = {'RT':[300, 256, 35], 'acc':[1,
1, 0]}
data['subject-2'] = {'RT':[400, 512, 100009], 'acc':
[1, 0, 1]}
data['subject-3'] = {'RT':[732, 542, 839], 'acc':[1,
1, 1]}
print data

# find all subject data information using the for
loop.
subjectIndex=1
RTList = []
while True:
    if subjectIndex>3:break
    key = 'subject-%d'%subjectIndex
    print data[key]["RT"]
    RTList.extend(data[key]["RT"])
    subjectIndex=subjectIndex+1

# calculate mean and std at group level. This is only
calculated once as it is out of the indention level
of the for group.
print "mean and standard deviation of
RT:",numpy.mean(RTList),numpy.std(RTList)
```

break是当条件满足时，就中断整个循环。有时，我们只需要中断某一次循环，放弃继续执行剩余的代码片断，而不是中断整个循环。这时，我们就应该使用continue关键词。Continue用于中断单次循环。

例如，我们发现上面的数据的第2号被试（'subject-2'）的有一个反应时长达100009。而且我们事后发现，被试2也不满足我们的实验筛选条件。因此，我们想在数据分析中，完全的删除掉被试2的数据。我们依然保留被试1和被试3的数据。对于这个需求，while或者for 和continue的结合是最合适的了。下面是这个功能通过for和continue结合的代码示例。 请读者朋友们通过while和continue结合，实现for与continue结合相同的功能。学习编程的捷径就在于，现在立刻马上开始编程。 纸上得来终觉浅。

```
import numpy
data = {}
data['subject-1'] = {'RT':[300, 256, 35], 'acc':[1,
1, 0]}
data['subject-2'] = {'RT':[400, 512, 100009], 'acc':
[1, 0, 1]}
data['subject-3'] = {'RT':[732, 542, 839], 'acc':[1,
1, 1]}
print data

# find all subject data information using the for
loop.
RTList = []
for subjectID in data.keys():
    if subjectID =='subject-2':
        print "skip 'subject-2'"
        continue
    print data[subjectID]["RT"]
    RTList.extend(data[subjectID]["RT"])

# calculate mean and std at group level. This is only
calculated once as it is out of the indention level
of the for group.
print "mean and standard deviation of
RT:",numpy.mean(RTList),numpy.std(RTList)
```

综合案例，条件语句和循环语句的综合使用