CS001编程零基础 Python语言入门

第四讲

本讲内容

- for语句
- while语句
- break
- 嵌套循环

循环

- 计算机周而复始地重复同样的步骤，称为循环。

计数循环：重复一定次数，通常用for语句

条件循环：重复直至某种情况发生，通常用while语句

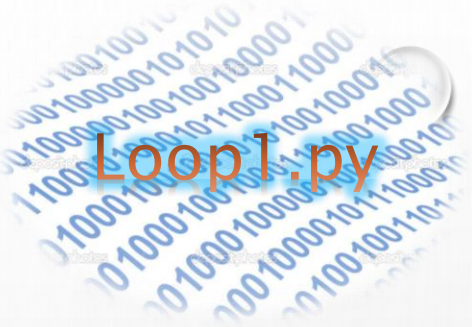
计数循环：for语句

for 变量 in 序列:

语句块

执行步骤：

1. 循环变量从序列中的第一个值开始
2. 对应序列中的某个值，完成一次语句块
3. 循环变量变为序列中的下一个值，重复上一步骤，直至穷尽序列所有的值。



使用列表

```
#Loop1.py
for i in [1,2,3,4,5]:
    print('hello')
```

- [1,2,3,4,5] 是列表。
- 列表是序列的一种。定界符是一对中括号，中间用逗号分隔。
- 列表就是一种“容器”，可以用来存放一组数据

每次循环称为一次迭代

循环变量名

一般变量起名字最好用有意义的单词

循环变量通常使用i、j、k等

使用列表

```
#Loop2.py
for i in [1,2,3,4,5]:
    print(i)
```

- 打印循环变量

请注意循环变量是如何变化的

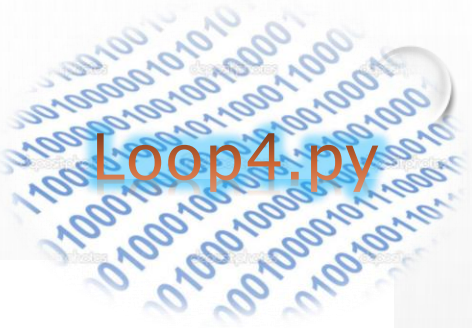
使用列表

```
#Loop3.py
for i in [1,2,3,4,5]:
    print(i,'* 8 =',i*8)
```

- 利用循环变量进行计算

如果没有循环，要写5句print。

range()函数




```
#Loop4.py
for i in range(1,5):
    print(i,'* 8 =',i*8)
```

range(1,5)给出的列表是[1,2,3,4]

- 当循环次数很多时，使用列表不方便。
- range()函数会创建一个列表，包含某个范围内的数。
- 为什么循环少了1次？

range()函数

Loop5.py

```
#Loop5.py
for i in range(5):
    print(i,'* 8 =',i*8)
```

range(5)给出的列表是[0,1,2,3,4]

- 为range函数只提供一个数时，得到的是从0开始的列表

range()函数的步长

Loop6.py

```
#Loop6.py
for i in range(1,10,2):
    print(i)
```

- 为range函数可以提供第3个数，称之为步长。
- 步长默认为1

试着改为range(5,26,5)

range()函数的步长

```
#Cuntdown.py
import time
for i in range(5,0,-1):
    print(i)
    time.sleep(1)
print("Time's up.")
```

- 步长也可以是负的。

time.sleep(1)让计算机等待1秒

range函数总结

range(start,end,step)

产生从start开始的序列。

start默认为0。

序列到end为止，

但不包括end。

step步长默认为1。

使用字符串作为序列


```
#Letters.py
for i in 'Hi there':
    print(i)
```

- 每行打印一个字母。

怎样让输出在同一行上？

for语句中的序列

可以是:

- 列表
- range()
- 字符串

条件循环

- 无法提前知道迭代多少次
- 无法提供序列

使用while语句

while语句

while 表达式:
    语句块

执行步骤：

1. 检查表达式的值(True 或 False)
2. 如果为True，完成一次语句块,重复上一步。
3. 如果为False，结束。

事先不知道循环次数


```
#Average.py
count = 0                #计数器初始为0
sum = 0                  #总分初始为0
score = float(input('Please input a score(-1 indicates end):'))#读取一个成绩
while score != -1:       #如果读到的不是-1，进入循环；如果是-1，离开循环
    sum += score         #把成绩累加到总分上
    count += 1           #计数器加一
    score = float(input('Please input a score(-1 indicates end):'))#读下一个成绩
print('You input', count, 'scores.')     #输出一共多少个成绩
print('The average is', sum/count)       #输出平均分
```

如果用户第一次就键入-1会怎样？

- 计算成绩平均分
- 成绩由用户键入，用户键入-1表示结束。
- sum+=score等价于sum=sum+score。+=是复合赋值，表示在原来基础上累加。

事先不知道循环次数

Average2.py

```
#Average2.py
count = 0                #计数器初始为0
sum = 0                  #总分初始为0
score = float(input('Please input a score(-1 indicates end):'))#读取一个成绩
while score != -1:       #如果读到的不是-1，进入循环；如果是-1，离开循环
    sum += score         #把成绩累加到总分上
    count += 1           #计数器加一
    score = float(input('Please input a score(-1 indicates end):'))#读下一个成绩
print('You input', count, 'scores.')     #输出一共多少个成绩
if count!=0:                             #如果不是0个成绩
    print('The average is',sum/count)    #输出平均分
```

能否只用一次input?

在循环之前预先读取score的值，然后才能进行条件检测。
每次循环语句块结束还要再读取一次score，为下次条件检测做准备。
所以input出现了两次。
循环语句块最后如果不做input，就会死循环。用Ctrl+C中断。

死循环和break配合使用


```
#Average3.py
count = 0
sum = 0
while True:                #死循环
    score = float(input('Please input a score(-1 indicates end):'))#读取一个成绩
    if score == -1: #如果读到-1
        break          #退出死循环
    sum += score       #累加总分
    count += 1         #计数加一
print('You input', count, 'scores.')
if count != 0:
    print('The average is', sum/count)
```

while True产生死循环。因为没有在循环前input读取，所以无法检测是否为-1。
循环里input读取score，如果为-1，结束循环。
无论循环执行到第几次迭代，无论当前迭代执行到语句块的哪里，一旦执行break，立即退出循环。
本次迭代余下的部分以及余下的几次迭代都不会再被执行了。

打印三阶乘法表

Times.py

```
#Times.py
for i in range(1, 4):          #外层循环，循环变量i从1到3，输出3行
    for j in range(1, 4):      #内层循环，当第i行时，j从1到3
        print(i, '*', j, '=', i*j, '\t', end='')#输出此时的i和j的乘积
    print()                    #本行结束，输出换行
```

```
1 * 1 = 1       1 * 2 = 2       1 * 3 = 3
2 * 1 = 2       2 * 2 = 4       2 * 3 = 6
3 * 1 = 3       3 * 2 = 6       3 * 3 = 9
```

外层循环变量i为某个数值（例如1）时，内层循环变量j从1变化到3，打印出3个算式。
‘\t’是转义符，代表制表符，其作用是让光标跳到下一栏。
传统的显示器一行只能显示80个字符，分成10栏，每栏8个字符。
因此，’abc\t’如果在行首输出，我们会看到abc后面有5个空格。而‘abcde\t’相当于abcde后面有3个空格。
print()相当于换行，因为end默认为’\n’（换行符）。

打印三阶乘法表(左下三角形）

Times2.py

```
#Times2.py
for i in range(1, 4):                    #外层循环，循环变量i从1到3，输出3行
    for j in range(1, i+1):              #内层循环，当第i行时，j从1到i
        print(i, '*', j, '=', i*j, '\t', end='')#输出此时的i和j的乘积
    print()                              #本行结束，输出换行
```

```
1 * 1 = 1
2 * 1 = 2       2 * 2 = 4
3 * 1 = 3       3 * 2 = 6       3 * 3 = 9
>>>
```

注意：j的变化范围取决于i的大小。

左上三角形怎么做？
九九乘法表呢？

打印三角形

```
#triangle.py
n = 3
for i in range(n):
    s = '*' * (2*i+1)
    print(s.center(2*n-1))
```

```
#三角形的高度为3行
#循环3遍，i从0到2
#本行有2i+1个星号
#居中显示，总宽度为2n-1
```

str.center(width)表示把字符串str居中显示，总宽度为width。
还有str.rjust(width) 函数可以右对齐输出str。

```
  *
 ***
*****
```

复习阅读

- 课本第4章的4.4～4.7