

# Python 语言程序设计: 流程控制之循环结构

李宽

likuan@dgut.edu.cn

东莞理工学院

2019.10



## 1 循环结构

- 循环结构
- 条件循环
- 循环控制保留字
- 循环相关其他内容
- 实例: 打印九九乘法表

# 循环结构

- 计数循环 (for)
- 条件循环 (while)
- 循环控制 (continue,break)
- 循环扩展 (循环 +else)

## 1 循环结构

- 循环结构
- 条件循环
- 循环控制保留字
- 循环相关其他内容
- 实例: 打印九九乘法表

# 遍历循环

遍历**某个结构**形成的循环运行方式

```
1 for <循环变量> in <遍历结构>:  
2     <语句块>
```

**重申:** 语句块概念

从遍历结构中逐一提取元素, 放在循环变量中

- 由关键字 for 和 in 组成, 完整遍历所有元素后结束
- 每次循环, 所获得元素放入循环变量, 并执行一次语句块

代码举例 (for-ex.py)

```
1 # 遍历循环示例  
2  
3 #n 为循环变量, range(7) 返回 [0,1,2,3,4,5,6]  
4 for n in range(7):  
5     print(n) # 循环中的语句块
```

# 遍历循环:range 补充

range() 函数创建一个**整数列表**, 一般用在 for 循环中。

---

```
1 range(start=0,stop[,step=1])
```

---

代码举例 (range-ex.py):

---

```
1 #[range-ex.py] range 函数: range(start,stop,step)
2
3 print("range(3) 输出:")
4 for n in range(3): # 只有一个参数时, 默认 start=0, step=1
5     print(n)
6
7 print("range(5,10) 输出:")
8 for n in range(5,10): # 两个参数, 给定 start,stop, 范围:[start,stop), step
9     print(n)
10
11 print("range(0,10,2) 输出:")
12 for n in range(0,10,2): # 三个参数, 分表表示 start,stop,step
13     print(n)
```

# 遍历循环: 遍历字符串

## 遍历字符串

```
1 for <字符> in <字符串>:  
2     <语句块>
```

## 代码举例 (for-str.py):

```
1 # 从字符串中依次取一个字符, 放入循环变量 c 中  
2  
3 #c: 字符串, 在 Python 中, 单个字符也是字符串  
4 for c in "hello,world":  
5     print(c)
```

# 遍历循环: 遍历列表

## 列表遍历循环<sup>1</sup>

```
1 for <item> in <列表>:  
2     <语句块>
```

遍历列表中的每个元素, 放入 item, 产生循环

代码举例 (for-list.py):

```
1 # 从列表中依次取一个元素, 放入循环变量 item  
2  
3 for item in [1234, "python", 56.78]:  
4     print(item)
```

<sup>1</sup>后续课程会探讨如何遍历字典及元组等



## 1 循环结构

- 循环结构
- **条件循环**
- 循环控制保留字
- 循环相关其他内容
- 实例: 打印九九乘法表

# 条件循环

由条件控制的循环运行方式

```
1 while <条件>:  
2     <语句块>
```

反复执行语句块, 直到条件不满足时结束

代码举例 (while-ex.py):

```
1 # 条件循环示例  
2  
3 x = 0 # 给变量 x 赋初值  
4  
5 ''' 条件是 x<=5, 只要满足这个条件, 就执行下面的代码块 (8,9 行);  
6     否则, 退出循环, 转第 11 行 '''  
7 while x <= 5:  
8     print(x)  
9     x+=1 # 等价与 x=x+1  
10  
11 print("执行结束")
```

# 条件循环: 补充

操作符	意义	描述
<code>+=</code>	加法赋值运算符	<code>c += a</code> 等效于 <code>c = c + a</code>
<code>-=</code>	减法赋值运算符	<code>c -= a</code> 等效于 <code>c = c - a</code>
<code>*=</code>	乘法赋值运算符	<code>c *= a</code> 等效于 <code>c = c * a</code>
<code>/=</code>	除法赋值运算符	<code>c /= a</code> 等效于 <code>c = c / a</code>
<code>%=</code>	取模赋值运算符	<code>c %= a</code> 等效于 <code>c = c % a</code>
<code>**=</code>	幂赋值运算符	<code>c **= a</code> 等效于 <code>c = c ** a</code>
<code>//=</code>	取整除赋值运算符	<code>c //= a</code> 等效于 <code>c = c // a</code>

# 条件循环: 无限循环<sup>2</sup>

代码举例 (dead-loop.py):

```
1 # 无限循环示例
2
3 x = 3 # 给变量 x 赋初值 3
4
5 while x > 0: # 因为后面的操作是对 x 增 1, 因此判断条件永远满足
6     print(x)
7     x+=1    # 等价与 x=x+1
8
9 print("执行结束")
```

按 CTRL+C 退出执行

- 一般来说, 要避免无限循环的发生!
- 有时候, 需要无限循环的效果。举例: 消息泵

<sup>2</sup>开发中常称“死循环”


## 1 循环结构

- 循环结构
- 条件循环
- **循环控制保留字**
- 循环相关其他内容
- 实例: 打印九九乘法表


# 循环控制保留字

- `break` 跳出并结束当前整个循环, 执行循环后的语句
- `continue` 结束当次循环, 继续执行后续循环


```
for var in sequence:
    # codes inside for loop
    if condition:
        break
    # codes inside for loop
# codes outside for loop
```




```
while test expression:
    # codes inside while loop
    if condition:
        break
    # codes inside while loop
# codes outside while loop
```



```
for var in sequence:
    # codes inside for loop
    if condition:
        continue
    # codes inside for loop
# codes outside for loop
```



```
while test expression:
    # codes inside while loop
    if condition:
        continue
    # codes inside while loop
# codes outside while loop
```



# 循环控制保留字:break

代码举例 (break-ex.py):

```
1 #break 跳出 while 循环范例
2 n = 1 # 给变量 n 赋初值 1
3 while True: # 无限循环
4     print (n)
5     n += 1 # 等价于 n=n+1
6     if n== 5: # 条件判断, 如果 n 等于 5, 执行 break, 跳出循环
7         break
8
9 print("while 循环执行完毕")
10
11 #break 跳出 for 循环范例
12 for s in "Python": # 遍历字符串 Python, 将每个字符放入循环变量 s
13     if s == "t": # 条件判断, 如果字符 (串) 是 "t", 执行 break, 跳出循环
14         break
15     print(s)
16
17 print("for 循环执行完毕")
```

# 循环控制保留字:continue

代码举例 (continue-ex.py):

```
1 #continue 在 while 循环中的示例
2 n = 0 # 给循环控制变量赋初值为
3
4 while n < 5: # 循环, 当 n<5 时
5     n += 1 # 等价于 n=n+1
6     if n == 3: # 条件判断, 如果 n 为 3, 执行 continue, 短路后续部分
7         continue
8     print (n) # 如果未执行 continue, 打印输出 n
9
10 print("while 循环执行完毕")
11
12 #continue 在 while 循环中的示例
13 for n in range(5): # 遍历 [0,5), 依次取值放入 n 中
14     n += 1
15     if n == 3:
16         continue
17     print (n)
18
19 print("for 循环执行完毕")
```



## 1 循环结构

- 循环结构
- 条件循环
- 循环控制保留字
- 循环相关其他内容
- 实例: 打印九九乘法表

# 循环的扩展

## 循环与 else

```
1 while condition:
2     statement_1(s)
3 else:
4     statement_2(s)
```

```
1 for <变量> in <遍历结构>:
2     <语句块1>
3 else:
4     <语句块2>
```

当循环正常执行结束时, 执行 else 块

- 没有被 break 语句退出
- 没有触发异常<sup>3</sup>

<sup>3</sup>后续章节介绍

# 循环的扩展

示例代码 (loop-else.py):

```
1 # 循环结构中的 else 示例
2 x = int(input("请输入 1 个整数: ")) # 接受用户键盘输入
3
4 while x <= 10: # 条件循环
5     print (x)
6     x = x+1
7     '''只有当 x 等于 7 这个条件不满足, break 不执行时,
8     才会执行第 12 行 else 部分'''
9     if x == 7: # 如果 x 等于 7, 执行 break, 跳出循环
10        break;
11 else:
12     print("执行 else 部分")
13
14 print("循环执行完毕")
```

# 补充:pass 语句

pass：不做任何事情。

一般用作占位符，保证程序结构的完整性，以免报错。

---

```
1 #while 循环
2 while 条件判断:
3     pass
4
5 #for 循环
6 for 变量名 in 序列:
7     pass
8
9 # 分支判断
10 if 条件:
11     pass
```

---

用处：编代码时写框架思路，具体实现可暂用 pass 进行占位，避免报错。

## 1 循环结构

- 循环结构
- 条件循环
- 循环控制保留字
- 循环相关其他内容
- 实例: 打印九九乘法表

# 综合举例

循环结构嵌套: 打印九九乘法表 (multi-table.py)

```
1 # 打印九九乘法表范例
2
3 print("打印下三角")
4 for i in range(1, 10): # 循环变量 i 取 [1,10)
5     for j in range(1, i+1): # 循环变量 j 取 [1,i+1)
6         # 打印, 不换行, 注意 {:2d} 的意义
7         print("{}*{}={:2d}".format(i,j,i*j),end=" ")
8     print("") # 只有 i 的值更换时才换行
9
10 print("打印上三角")
11 for i in range(1, 10): # 循环变量 i 取 [1,10)
12     for j in range(1, i): # 循环变量 j 取 [1,i)
13         print(" ",end=" ") # 打印前面的空格, 不换行
14     for j in range(i, 10): # 循环变量 j 取 [i,10), 打印真正有效信息
15         print("{}*{}={:2d}".format(i,j,i*j),end=" ")
16     print("") # 只有 i 的值更换时才换行
```