PYTHON
编程基础

循环语句概述和for循环

概述

通过循环，可以使得某些语句重复执行多次。

例如

我们要计算从1到n的和，可以使用一个变量sum=0保存求和结果，并设置一个变量i、让其遍历1到n这n个整数；
对于i的每一个取值，执行sum+=i的运算；遍历结束后，sum中即保存了求和结果。

概述

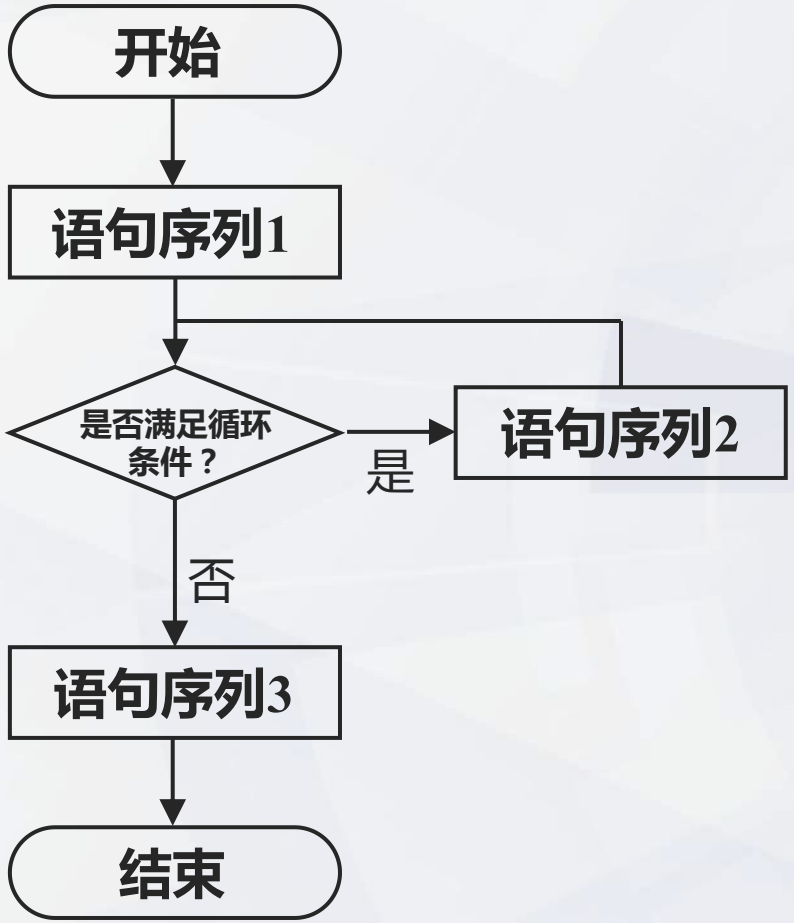
提示

“遍历”这个词在计算机程序设计中经常会用到，其表示对某一个数据中的数据元素按照某种顺序进行访问，使得每个数据元素访问且仅访问一次。

例如

对于列表ls=[1, 'Python', True]中的3个元素，如果按照某种规则（如从前向后或从后向前）依次访问了1、'Python'、True这3个元素，且每个元素仅访问了一次，则可以说对列表ls完成了一次遍历。

循环语句执行过程



提示

循环条件判断和语句序列2构成了循环语句：只要满足循环条件，就会执行语句序列2；执行语句序列2后，会再次判断是否满足循环条件。

for循环

用于遍历可迭代对象中的每一个元素，并根据当前访问的元素做数据处理，其语法格式为：

```
for 变量名 in 可迭代对象:
    语句序列
```

例如

```
1  ls=['Python','C++','Java']
2  for k in ls:
3      print(k)
```

```
Python
C++
Java
```

for循环

再如

```
1  d={'Python':1,'C++':2,'Java':3}
2  for k in d: #注意for后要写上“:”
3      print('%s:%d'%(k,d[k]))
```

```
Python:1
C++:2
Java:3
```

提示

使用for遍历字典中的元素时，每次获取到的是元素的键，通过键可以再获取到元素的值。

for循环

使用for循环时，如果需要遍历一个数列中的所有数字，则通常利用range函数生成一个可迭代对象。

range函数的语法格式如下：

```
range([beg, ]end[, step])
```

for循环

例如

```
1  print(list(range(1,5,2))) #输出“[1, 3]”
2  print(list(range(5,-1,-2))) #输出“[5, 3, 1]”
3  print(list(range(1,5))) #输出“[1, 2, 3, 4]”
4  print(list(range(5))) #输出“[0, 1, 2, 3, 4]”
```

提示

range函数返回的是一个可迭代对象，通过list函数可将该对象转换为列表。

for循环

例 ▶ 使用for循环实现1到n的求和。

```
1  n=eval(input('请输入一个大于0的整数：'))
2  sum=0
3  for i in range(1,n+1): #range函数将生成由1到n这n个整数组成的可迭代对象
4      sum+=i
5  print(sum) #输出求和结果
```

for循环

例 ▶ 使用for循环实现1到n之间所有奇数的和。

```
1  n=eval(input('请输入一个大于0的整数：'))
2  sum=0
3  for i in range(1,n+1,2): #步长2，因此会生成1、3、5、…等奇数
4      sum+=i
5  print(sum) #输出求和结果
```