

# Python 循环语句

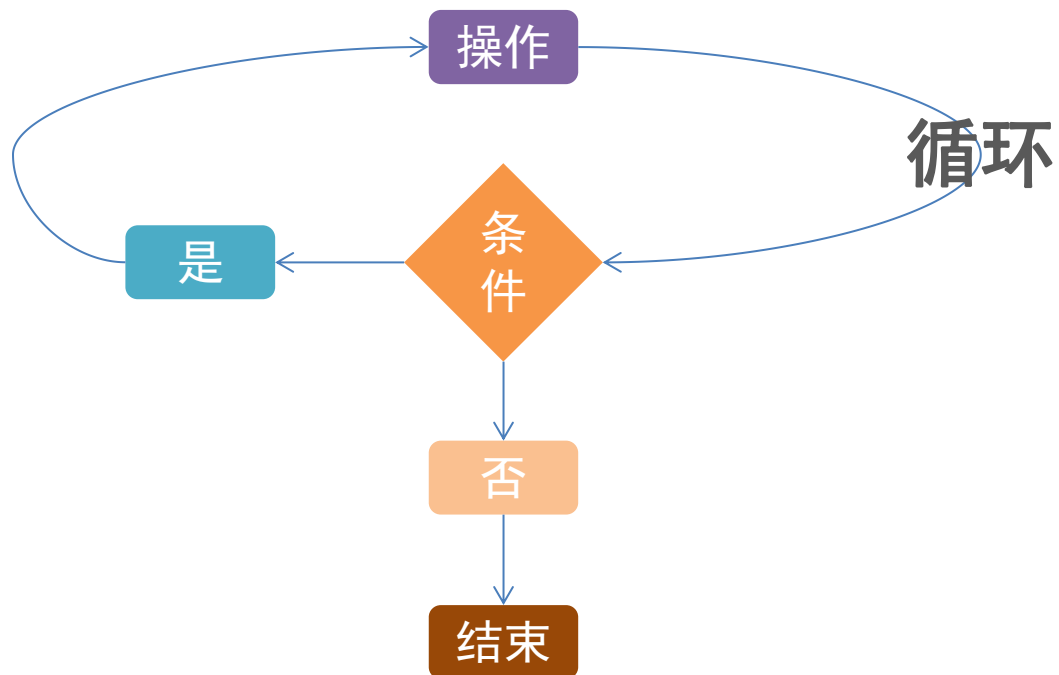


黑马程序员  
[www.itheima.com](http://www.itheima.com)

传智教育旗下  
高端IT教育品牌

## 前言

循环普遍存在于日常生活中，同样，在程序中，循环功能也是至关重要的基础功能。



## 为什么学习循环语句

循环在程序中同判断一样，也是广泛存在的，是非常多功能实现的基础：



Bilibili 循环轮播图

循环广告牌

批量修图

视频轮播

音乐轮播

图片轮播

大喇叭喊话

动态壁纸

视频监控



**循环同样是程序最基础  
最核心的逻辑功能**



# 目录

## Contents



- ◆ while循环的基础语法
- ◆ while循环的基础案例
- ◆ while循环的嵌套应用
- ◆ while循环的嵌套案例
- ◆ for循环的基础语法
- ◆ for循环的嵌套应用
- ◆ 循环中断 : break和continue
- ◆ 综合案例



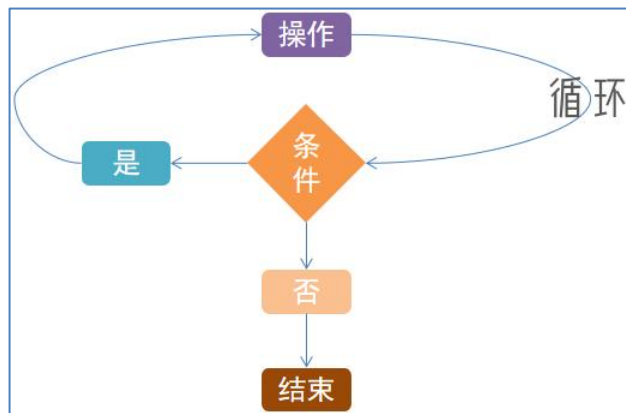
# 学习目标

Learning Objectives

掌握使用while循环的基础应用

## while循环语句

生活中的循环



程序中的循环

while 条件:

条件满足时，做的事情1

条件满足时，做的事情2

条件满足时，做的事情3

...(省略)...



每天都去向小美表白  
直到成功为止

**只要条件满足  
会无限循环执行**



## while循环语句

小美心软，只要表白100次，就会成功

```
print("小美，我喜欢你")  
print("小美，我喜欢你")  
print("小美，我喜欢你")  
... (还有97次)...
```



累



## 程序中的while循环

使用循环语句简单搞定

```
i = 0
while i < 100:
    print("小美，我喜欢你")
    i += 1
```





## while循环注意点

```
i = 0
while i < 100:
    print("小美，我喜欢你")
    i += 1
```

1. while的条件需得到布尔类型，True表示继续循环，False表示结束循环
2. 需要设置循环终止的条件，如i += 1配合 i < 100，就能确保100次后停止，否则将无限循环
3. 空格缩进和if判断一样，都需要设置





# 总结

## 1. while循环的语法格式

```
while 条件:  
    条件满足时，做的事情1  
    条件满足时，做的事情2  
    条件满足时，做的事情3  
    ... (省略) ...
```

## 2. while循环的注意事项

- 条件需提供布尔类型结果，True继续，False停止
- 空格缩进不能忘
- 请规划好循环终止条件，否则将无限循环

练习

## 练习案例：求1-100的和

需求：通过while循环，计算从1累加到100的和

提示：

1. 终止条件不要忘记，设置为确保while循环100次
2. 确保累加的数字，从1开始，到100结束





# 目录

## Contents

- ◆ while循环的基础语法
- ➔ ◆ while循环的基础案例
- ◆ while循环的嵌套应用
- ◆ while循环嵌套案例
- ◆ for循环的基础语法
- ◆ for循环的嵌套应用
- ◆ 循环中断：break和continue
- ◆ 综合案例



# 学习目标

Learning Objectives

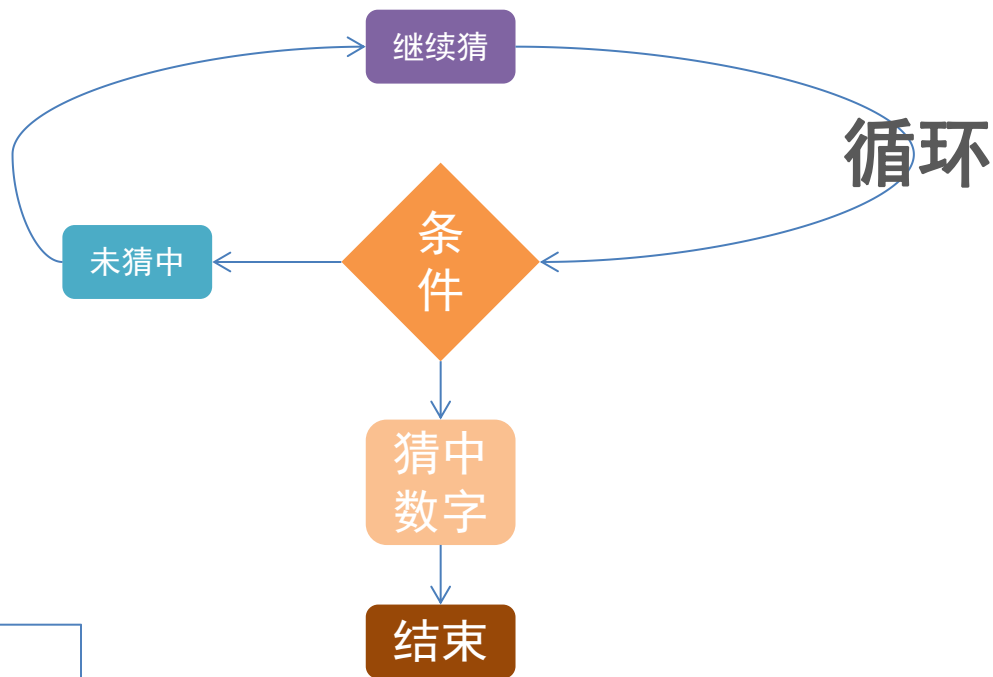
能够使用while循环，完成猜数字案例

## 猜数字案例

设置一个范围1-100的随机整数变量，通过while循环，配合input语句，判断输入的数字是否等于随机数

- 无限次机会，直到猜中为止
- 每一次猜不中，会提示大了或小了
- 猜完数字后，提示猜了几次
- 提示：
- 无限次机会，终止条件不适合用数字累加来判断
  - 可以考虑布尔类型本身 (True or False)
- 需要提示几次猜中，就需要提供数字累加功能
- 随机数可以使用：

```
import random  
num = random.randint(1, 100)
```





# 目录

## Contents



- ◆ while循环的基础语法
- ◆ while循环的基础案例
- ◆ while循环的嵌套应用
- ◆ while循环的嵌套案例
- ◆ for循环的基础语法
- ◆ for循环的嵌套应用
- ◆ 循环中断 : break和continue
- ◆ 综合案例



# 学习目标

Learning Objectives

1. 掌握while嵌套循环基本格式
2. 完成while嵌套循环的应用



## while循环的嵌套

生活中的嵌套循环



每天都去向小美表白  
直到成功为止

每次表白的流程是：  
送10朵玫瑰然后表白

程序中的循环

```
while 条件1:
```

条件1满足时，做的事情1

条件1满足时，做的事情2

条件1满足时，做的事情3

...(省略)...

```
while 条件2:
```

条件2满足时，做的事情1

条件2满足时，做的事情2

条件2满足时，做的事情3

...(省略)...

**循环内  
套循环**



## while循环的嵌套

```
i = 1
while i <= 100:
    print(f"今天是第{i}天, 准备表白.....")
    j = 1
    while j <= 10:
        print(f"送给小美第{j}只玫瑰花")
        j += 1
    print("小美, 我喜欢你")
    i += 1

print(f"坚持到第{i - 1}天, 表白成功")
```

**表白100天**  
**每一天都会送10朵玫瑰花**



## while循环的嵌套-注意点

```
i = 1
while i <= 100:
    print(f"今天是第{i}天, 准备表白.....")
    j = 1
    while j <= 10:
        print(f"送给小美第{j}只玫瑰花")
        j += 1
    print("小美, 我喜欢你")
    i += 1

print(f"坚持到第{i - 1}天, 表白成功")
```

- 同判断语句的嵌套一样，循环语句的嵌套，要注意空格缩进。
  - **基于空格缩进来决定层次关系**
- 注意条件的设置，避免出现无限循环（除非真的需要无限循环）





# 总结

## 1. 嵌套循环的语法格式:

- 见右图

## 2. 嵌套循环需要注意的地方:

- 注意条件的控制，避免无限循环
- 多层嵌套，主要空格缩进来确定层次关系

## 3. 嵌套循环的使用难点:

- 循环条件的控制，层次越多越复杂，需要细心+耐心

```
while 条件1:
```

```
    条件1满足时，做的事情1
```

```
    条件1满足时，做的事情2
```

```
    条件1满足时，做的事情3
```

```
    ... (省略) ...
```

```
while 条件2:
```

```
    条件2满足时，做的事情1
```

```
    条件2满足时，做的事情2
```

```
    条件2满足时，做的事情3
```

```
    ... (省略) ...
```



# 目录

Contents



- ◆ while循环的基础语法
- ◆ while循环的基础案例
- ◆ while循环的嵌套应用
- ◆ while循环的嵌套案例
- ◆ for循环的基础语法
- ◆ for循环的嵌套应用
- ◆ 循环中断 : break和continue
- ◆ 综合案例



# 学习目标

Learning Objectives

1. 掌握使用while嵌套循环，打印九九乘法表

## 补充知识-print输出不换行

默认print语句输出内容会自动换行，如下图：

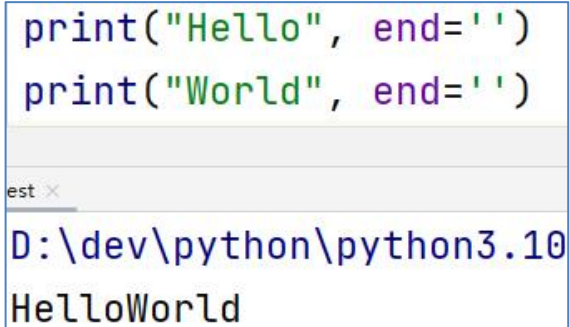
```
print("Hello")
print("World")
```



est x  
D:\dev\python\p  
Hello  
World

在即将完成的案例中，我们需要使用print语句，**输出不换行的功能**，非常简单，实现方式如下：

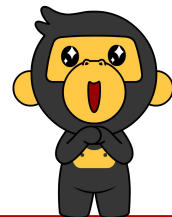
```
print("Hello", end='')
print("World", end='')
```



est x  
D:\dev\python\python3.10  
HelloWorld

如图，在print语句中，加上 `end=''` 即可输出不换行了

ps: `end=''` 是使用的的方法传参功能，我们在后面会详细讲解。



## 补充知识-制表符\t

在字符串中，有一个特殊符号：`\t`，效果等同于在键盘上按下：`tab`键。

它可以让我们多行字符串进行对齐。

比如：

```
print("Hello World")
print("itheima best")

print("Hello\tWorld")
print("itheima\tbest")
```

使用空格  
无法对齐

使用\t后，可以对齐



```
test x
D:\dev\python\python3.10.4\python.exe D:/py
Hello World
itheima best
Hello | World
itheima | best
```



 案例

## 练习案例-打印九九乘法表

通过while循环，输出如下九九乘法表内容

```
1*1=1
1*2=2 2*2=4
1*3=3 2*3=6 3*3=9
1*4=4 2*4=8 3*4=12 4*4=16
1*5=5 2*5=10 3*5=15 4*5=20 5*5=25
1*6=6 2*6=12 3*6=18 4*6=24 5*6=30 6*6=36
1*7=7 2*7=14 3*7=21 4*7=28 5*7=35 6*7=42 7*7=49
1*8=8 2*8=16 3*8=24 4*8=32 5*8=40 6*8=48 7*8=56 8*8=64
1*9=9 2*9=18 3*9=27 4*9=36 5*9=45 6*9=54 7*9=63 8*9=72 9*9=81
```

提示：

- 2层循环，外层控制行，内层控制列
- 外层循环和内存循环的累加数字变量，用以辅助输出乘法表的数值





# 目录

## Contents

- ◆ while循环的基础语法
- ◆ while循环的基础案例
- ◆ while循环的嵌套应用
- ◆ while循环的嵌套案例
- ➔ ◆ for循环的基础语法
- ◆ for循环的嵌套应用
- ◆ 循环中断 : break和continue
- ◆ 综合案例



# 目录

Contents

◆ for循环的基础语法



◆ 基础语法

◆ range语句

◆ 变量作用域

# 学习目标

Learning Objectives

1. 掌握for循环基本语法

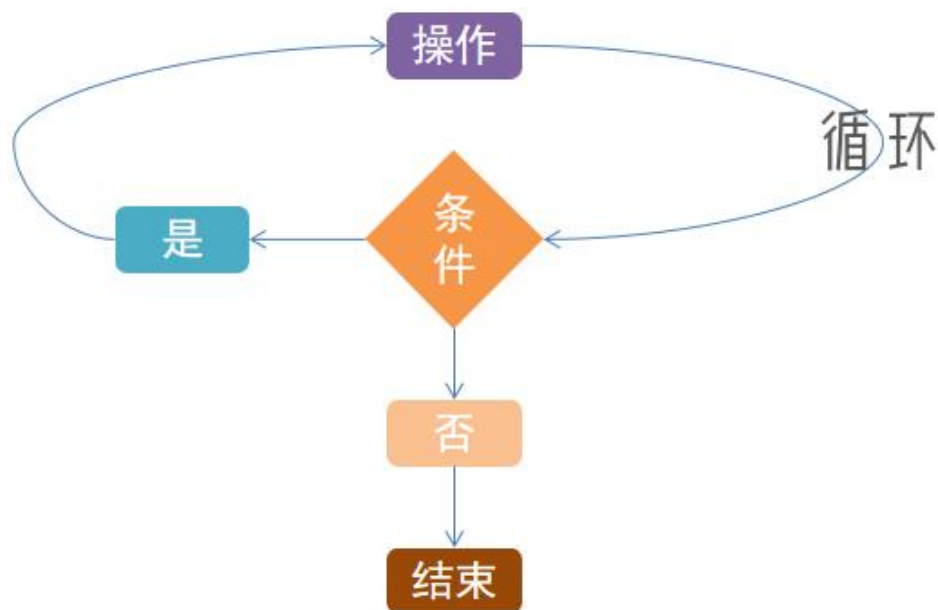
## for循环

除了while循环语句外，Python同样提供了for循环语句。

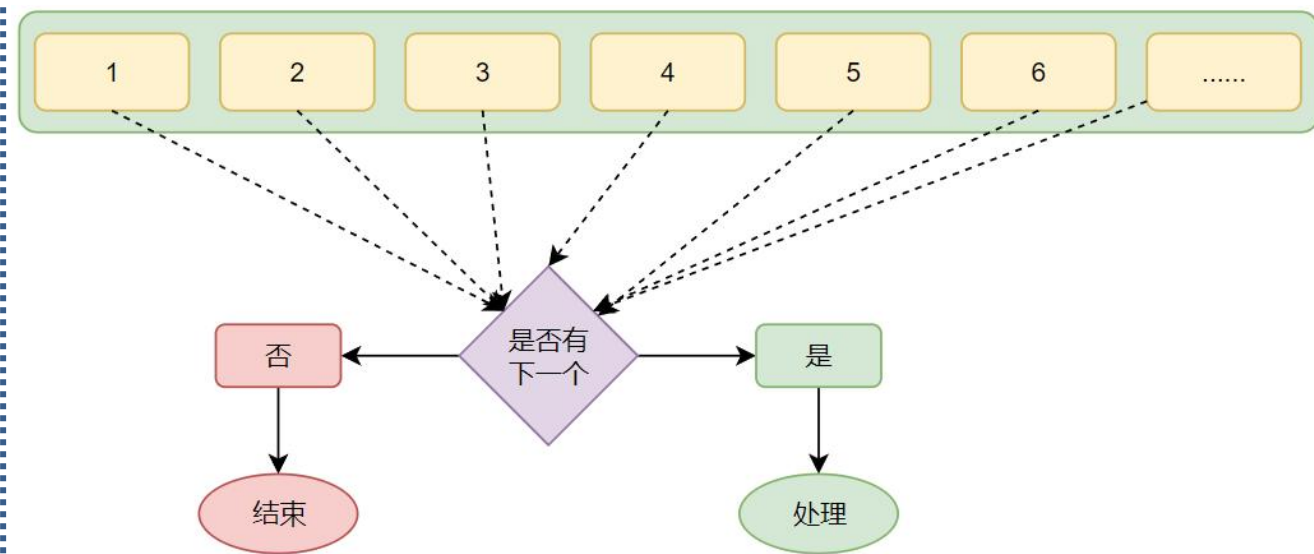
两者能完成的功能基本差不多，但仍有一些区别：

- while循环的循环条件是自定义的，自行控制循环条件
- for循环是一种“轮询”机制，是对一批内容进行“逐个处理”

**for循环就是将“待办事项”  
逐个完成的循环机制**



while循环



for循环

## for循环语句

### 生活中的for循环



- 洗碗，一个个洗干净
- 剥蒜，一个个剥干净
- 送快递，一个个送达
- 吃瓜子，一个个吃下
- ...，一个个完成

### 程序中的for循环

for 临时变量 in 待处理数据集:  
    循环满足条件时执行的代码

**从待处理数据集中：逐个取出数据  
赋值给临时变量**



for循环语句

遍历字符串

```
# 定义字符串name  
name = "itheima"  
  
# for循环处理字符串  
for x in name:  
    print(x)
```

运行结果如下：

```
i  
t  
h  
e  
i  
m  
a
```

可以看出，for循环是将字符串的内容：**依次取出**

所以，for循环也被称之为：**遍历循环**



## for循环注意点

```
# 定义字符串name  
name = "itheima"  
  
# for循环处理字符串  
for x in name:  
    print(x)
```

同while循环不同，for循环是无法定义循环条件的。

只能从被处理的数据集中，依次取出内容进行处理。

所以，理论上讲，Python的for循环无法构建无限循环（被处理的数据集不可能无限大）







# 总结

1. for循环的语法格式是:

```
for 临时变量 in 待处理数据集:  
    循环满足条件时执行的代码
```

2. for循环的注意点
  - 无法定义循环条件，只能被动取出数据处理
  - 要注意，循环内的语句，需要有空格缩进

**练习****练习案例：数一数有几个a**

定义字符串变量name，内容为：“itheima is a brand of itcast”

通过for循环，遍历此字符串，统计有多少个英文字母：“a”

```
D:\dev\python\python3.10.4\python.exe D:/py  
itheima is a brand of itcast中共含有：4个字母a
```

提示：

1. 计数可以在循环外定义一个整数类型变量用来做累加计数
2. 判断是否为字母“a”，可以通过if语句结合比较运算符来完成



# 目录

Contents

◆ for循环的基础语法

◆ 基础语法

➔ ◆ range语句

◆ 变量作用域

## range语句

**for** 临时变量 **in** 待处理数据集(可迭代对象):  
    循环满足条件时执行的代码

语法中的：待处理数据集，严格来说，称之为：**可迭代类型**

可迭代类型指，**其内容可以一个个依次取出的一种类型**，包括：

- 字符串
- 列表
- 元组
- 等

目前我们只学习了字符串类型，其余类型在后续章节会详细学习它们



## range语句

for循环语句，本质上是遍历：可迭代对象。

尽管除字符串外，其它可迭代类型目前没学习到，但不妨碍我们通过学习range语句，获得一个简单的数字序列（可迭代类型的一种）。

语法1:

```
range(num)
```

获取一个从0开始，到num结束的数字序列（不含num本身）

如range(5)取得的数据是：[0, 1, 2, 3, 4]



## range语句

语法2:

```
range(num1,
```

*num2*)  
获得一个从num1开始，到num2结束的数字序列（不含num2本身）

如，range(5, 10)取得的数据是：[5, 6, 7, 8, 9]

语法3:

```
range(num1, num2, step)
```

获得一个从num1开始，到num2结束的数字序列（不含num2本身）

数字之间的步长，以step为准（step默认为1）

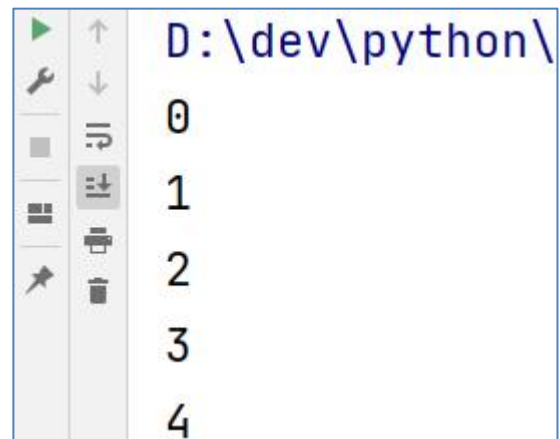
如，range(5, 10, 2)取得的数据是：[5, 7, 9]



for循环遍历range序列

```
# for循环处理字符串  
for i in range(5):  
    print(i)
```

运行结果如下：



```
D:\dev\python\  
0  
1  
2  
3  
4
```





# 总结

1. range语句的功能是：

获得一个数字序列（可迭代类型的一种）

2. range语句的语法格式：

语法1: `range(num)`

语法2: `range(num1, num2)`

语法3: `range(num1, num2, step)`

3. range语句的注意事项：

- 语法1从0开始，到num结束（不含num本身）
- 语法2从num1开始，到num2结束（不含num2本身）
- 语法3从num1开始，到num2结束（不含num2本身），步长以step值为准

range的用途很多，多数用在for循环场景



 **练习****练习案例：有几个偶数**

定义一个数字变量num，内容随意

并使用range()语句，获取从1到num的序列，使用for循环遍历它。

在遍历的过程中，统计有多少偶数出现



```
Run: test x
D:\dev\python\python3.10.4\python
1到100(不含100本身)范围内，有49个偶数。
```

提示：

1. 序列可以使用：range(1, num)得到
2. 偶数通过if来判断，判断数字余2是否为0即可



# 目录

Contents

◆ for循环的基础语法

◆ 基础语法

◆ range语句

➔ ◆ 变量作用域

思考

```
for i in range(5):  
    print(i)  
  
print(i)
```

如图代码，思考一下：

红框中的print语句，能否访问到变量 i ？

**规范上：不允许**  
**实际上：可以**

## for循环的变量作用域

**for** 临时变量 **in** 待处理数据集:  
循环满足条件时执行的代码

回看for循环的语法，我们会发现，将从数据集（序列）中取出的数据赋值给：临时变量  
为什么是临时的呢？

临时变量，在**编程规范**上，作用范围（作用域），只限定在for循环内部

如果在for循环外部访问临时变量：

- 实际上是可以访问到的
- 在编程规范上，是不允许、不建议这么做的

## for循环的变量作用域

如果实在需要在循环外访问循环内的临时变量，可以在循环外预先定义

```
i = 0
for i in range(5):
    print(i)

print(i)
```

如图，每一次循环的时候，都会将取出的值赋予i变量。

- 由于i变量是在循环之前（外）定义的
- 在循环外访问i变量是合理的、允许的



# 总结

1. for循环中的临时变量，其作用域限定为：

循环内

2. 这种限定：

- 是编程规范的限定，而非强制定限
- 不遵守也能正常运行，但是不建议这样做
- 如需访问临时变量，可以预先在循环外定义它



# 目录

## Contents

- ◆ while循环的基础语法
- ◆ while循环的基础案例
- ◆ while循环的嵌套应用
- ◆ while循环的嵌套案例
- ◆ for循环的基础语法
- ◆  for循环的嵌套应用
- ◆ 循环中断 : break和continue
- ◆ 综合案例



# 学习目标

Learning Objectives

1. 掌握for循环的嵌套使用



## for循环的嵌套

同while一样，for循环也支持嵌套使用

生活中的嵌套循环



每天都去向小美表白  
直到成功为止

每次表白的流程是：  
送10朵玫瑰然后表白

程序中的嵌套for循环

```
for 临时变量 in 待处理数据集(序列):  
    循环满足条件应做的事情 1  
    循环满足条件应做的事情 2  
    循环满足条件应做的事情 N  
    ...  
for 临时变量 in 待处理数据集(序列):  
    循环满足条件应做的事情 1  
    循环满足条件应做的事情 2  
    循环满足条件应做的事情 N
```



## for循环的嵌套

同样以向小美表白的案例为例

- 坚持表白100天
- 每天送花10束

```
i = 1
for i in range(1, 101):
    print(f"今天是向小美表白的第{i}天，坚持.")
    for j in range(1, 11):
        print(f"送给小美的第{j}朵玫瑰花")
    print(f"小美，我喜欢你(第{i}天的表白结束)")

print(f"第{i}天，表白成功")
```

**表白100天**  
**每一天都会送10朵玫瑰花**



## for循环的嵌套注意点

```
i = 1
for i in range(1, 101):
    print(f"今天是向小美表白的第{i}天，坚持.")
    for j in range(1, 11):
        print(f"送给小美的第{j}朵玫瑰花")
        print(f"小美，我喜欢你(第{i}天的表白结束)")

print(f"第{i}天，表白成功")
```

如图，和while循环一样，需要注意缩进  
因为通过缩进，确定层次关系



## for循环的嵌套注意点

我们目前学习了2个循环，while循环和for循环。

这两类循环语句是可以相互嵌套的，如下，小美表白的案例可以改为：

```
i = 1
for i in range(1, 101):
    print(f"今天是向小美表白的第{i}天，坚持.")
    for j in range(1, 11):
        print(f"送给小美的第{j}朵玫瑰花")
    print(f"小美，我喜欢你(第{i}天的表白结束)")

print(f"第{i}天，表白成功")
```



```
i = 1
while i <= 100:
    print(f"今天是向小美表白的第{i}天，坚持.")
    for j in range(1, 11):
        print(f"送给小美的第{j}朵玫瑰花")
    print(f"小美，我喜欢你(第{i}天的表白结束)")
    i += 1

print(f"表白第{i - 1}天，表白成功")
```

# 总结

1. for循环的嵌套语法:

见右图

for 循环 或 while 循环:

```
循环满足条件应做的事情 1  
循环满足条件应做的事情 2  
循环满足条件应做的事情 N
```

...

for 循环 或 while 循环:

```
循环满足条件应做的事情 1  
循环满足条件应做的事情 2  
循环满足条件应做的事情 N
```

2. 注意事项:

- 需要注意缩进，嵌套for循环同样通过缩进确定层次关系
- for循环和while循环可以相互嵌套使用

 案例

## 练习案例-for循环打印九九乘法表

通过for循环，输出如下九九乘法表内容

```
1*1=1
1*2=2 2*2=4
1*3=3 2*3=6 3*3=9
1*4=4 2*4=8 3*4=12 4*4=16
1*5=5 2*5=10 3*5=15 4*5=20 5*5=25
1*6=6 2*6=12 3*6=18 4*6=24 5*6=30 6*6=36
1*7=7 2*7=14 3*7=21 4*7=28 5*7=35 6*7=42 7*7=49
1*8=8 2*8=16 3*8=24 4*8=32 5*8=40 6*8=48 7*8=56 8*8=64
1*9=9 2*9=18 3*9=27 4*9=36 5*9=45 6*9=54 7*9=63 8*9=72 9*9=81
```

提示：

- 2层循环，外层控制行，内层控制列
- 可使用range语句来得到数字序列进行for循环
- 内层for循环的range最大范围，取决于当前外层循环的数字





# 目录

## Contents

- ◆ while循环的基础语法
- ◆ while循环的基础案例
- ◆ while循环的嵌套应用
- ◆ while循环的嵌套案例
- ◆ for循环的基础语法
- ◆ for循环的嵌套应用
- ➔ ◆ 循环中断 : break和continue
- ◆ 综合案例



# 学习目标

Learning Objectives

1. 掌握使用continue和break关键字控制循环





思考：无论是while循环或是for循环，都是重复性的执行特定操作。

在这个重复的过程中，会出现一些其它情况让我们不得不：

- 暂时跳过某次循环，直接进行下一次
- 提前退出循环，不在继续

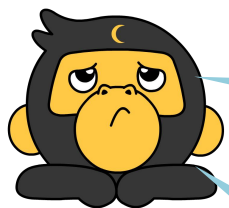
对于这种场景，Python提供continue和break关键字

用以对循环进行临时跳过和直接结束

## continue

continue关键字用于：中断本次循环，直接进入下一次循环

continue可以用于：for循环和while循环，效果一致



每天都去向小美表白  
送一日三餐，直到成功

每天送晚餐的时候看小  
美脸色，今天不高兴就  
不送了，明天继续

continue

让我们通过代码来模拟一下追求小美的过程。（以for循环为例，while循环效果一样）

通过input输入语句，来决定今天小美的心情(0表示好心情，1表示坏心情)

```
for i in range(1, 101):
    print(f"追求小美的第{i}天，坚持.....")
    print("给小美送早饭，比心")
    print("给小美送午饭，比心")
    if input(f"今天小美的心情好像(0好心情，1坏心情)") == '1':
        print("小美心情不好，晚饭不送了，撤退.....")
        print()
        continue
    print("给小美送晚饭，送小美回家并表白")
    print()
```

## continue在嵌套循环中的应用

continue关键字只可以控制：**它所在的循环临时中断**

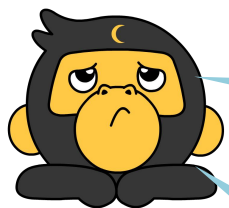
```
for i in range(1, 101):  
    2 print(f"追求小美的第{i}天, 坚持.....")  
    print("给小美送早饭, 比心")  
    print("给小美送午饭, 比心")  
  
    1 for j in range(1, 11):  
        print(f"向小美送今天第{j}朵花")  
        continue  
        print("小美, 我喜欢你")  
  
    print("给小美送晚饭, 送小美回家并表白")  
    print()
```

**continue只能控制左图编号1的for循环**  
**对编号2的for循环, 无影响**

break

break关键字用于：直接结束所在循环

break可以用于：for循环和while循环，效果一致



每天都去向小美表白  
送一日三餐，直到成功

某天小美说，不要再打  
扰我了，以后就不再缠  
着她了

## break

让我们通过代码来模拟一下追求小美的过程。（以for循环为例，while循环效果一样）

通过input输入语句，来决定今天小美的心情(0表示好心情，1表示坏心情)

```
for i in range(1, 101):
    print(f"追求小美的第{i}天，坚持.....")
    print("给小美送早饭，比心")
    print("给小美送午饭，比心")
    if input(f"如果你是小美，请告诉我你是否明确拒绝(0再观察观察，1不合适拒绝)") == '1':
        print("小美拒绝我了，以后不追求小美了。T.T")
        break
    print("给小美送晚饭，送小美回家并表白")
print()
```

## break在嵌套循环中的应用

break关键字同样只可以控制：**它所在的循环永久中断**

```
i = 1
j = 1
while i <= 100:
    2 print(f"追求小美的第{i}天，坚持.....")
    print("给小美送早饭，比心")
    print("给小美送午饭，比心")
    print("给小美送晚饭，送小美回家并表白")

    while j <= 10:
        1 print(f"送给小美第{j}朵花")
        print("小美好像心情不好，今天不送花了，明天继续....")
        break
        j += 1

    i += 1
    print()
```

**break只能控制左图编号1的循环**

**对编号2的循环，无影响**



# 总结

## 1. continue的作用是：

中断所在循环的当次执行，直接进入下一次

## 2. break的作用是：

直接结束所在的循环

## 3. 注意事项：

- continue和break，在for和while循环中作用一致
- 在嵌套循环中，只能作用在所在的循环上，无法对上层循环起作用





# 目录

## Contents

- ◆ while循环的基础语法
- ◆ while循环的基础案例
- ◆ while循环的嵌套应用
- ◆ while循环的嵌套案例
- ◆ for循环的基础语法
- ◆ for循环的嵌套应用
- ◆ 循环中断：break和continue
- ➔ ◆ 综合案例



# 学习目标

Learning Objectives

1. 基于学到的循环知识，完成发工资案例

**案例****练习案例：发工资**

某公司，账户余额有1W元，给20名员工发工资。

- 员工编号从1到20，从编号1开始，依次领取工资，每人可领取1000元
- 领工资时，财务判断员工的绩效分（1-10）（随机生成），如果低于5，不发工资，换下一位
- 如果工资发完了，结束发工资。

```
Run: test
↑
↓
员工12, 绩效分3, 低于5, 不发工资, 下一位。
员工13, 绩效分1, 低于5, 不发工资, 下一位。
员工14, 绩效分4, 低于5, 不发工资, 下一位。
向员工15发放工资1000元, 账户余额还剩余2000元
向员工16发放工资1000元, 账户余额还剩余1000元
员工17, 绩效分2, 低于5, 不发工资, 下一位。
向员工18发放工资1000元, 账户余额还剩余0元
工资发完了, 下个月领取吧。
```

```
money = 10000
for i in range(1, 21):
    import random
    score = random.randint(1, 10)
    if score < 5:
        print(f"员工{i}, 绩效分{score}, 低于5, 不发工资, 下一位。")
        continue
    else:
        money -= 1000
        print(f"向员工{i}发放工资1000元, 账户余额还剩余{money}元")
        if money == 0:
            print("工资发完了, 下个月领取吧。")
            break
```

提示:

- continue用于跳过员工，break直接结束发工资
- if判断余额，不要忘记发完工资后，余额减少1000哦



传智教育旗下高端IT教育品牌