

语句及表达式

1. 行

(1) 物理行：程序员编写代码的行。

(2) 逻辑行：python解释器需要执行的指令。

(3) 建议：

一个逻辑行在一个物理行上。

如果一个物理行中使用多个逻辑行，需要使用分号；隔开。

(4) 换行：

如果逻辑行过长，可以使用隐式换行或显式换行。

隐式换行：所有括号里的内容换行，称为隐式换行

括号包括：() [] {} 三种

显式换行：通过折行符\ (反斜杠)换行，必须放在一行的末尾，目的是告诉解释器，下一行也是本行的语句。

```
1  # 4个物理行 4个逻辑行
2  a = 1
3  b = 2
4  c = a + b
5  print(c)
6
7  # 1个物理行 4个逻辑行(不建议)
8  a = 1;b = 2;c = a + b;print(c)
9
10 # 4个物理行 1个逻辑行，显式换行
11 d = 1 + \
12     2 + \
13     3 + \
14     4
15
16 # 4个物理行 1个逻辑行，隐式换行
17 e = (1 +
18     2 +
19     3 +
20     4)
```

2. 条件语句

学习目标

- 条件语句作用
- if语法
- if...else...

- 多重判断
- if嵌套

了解条件语句

举个假设，疫情期间，小红要乘坐火车，但车站规定，体温高于37.2度的不能进站，只有当体温在合格范围内才可以进入车站，那么现在就需要判断小红的体温情况了，如果小红的体温是36度，那么她的体温低于37.2，就可以进站；如果小红的体温是38度，那么她的体温是高于37.2度的，不能够进站乘车。

其实这里所谓的判断就是条件语句，即**条件成立执行某些代码，条件不成立则不执行这些代码。**

2.1 if语法

```
1  if 条件:
2      条件成立执行的代码1
3      条件成立执行的代码2
4      .....
```

快速体验:

```
1  xiaohong = 38.5
2  if xiaohong > 37.2:
3      print("不能够进车站")
```

执行结果如下:

不能够进车站

2.2 if...else...语法

```
1  if 条件:
2      条件成立执行的代码1
3      条件成立执行的代码2
4      .....
5  else:
6      条件不成立执行的代码1
7      条件不成立执行的代码2
8      .....
```

额外加一个小功能，如果体温正常，欢迎乘客进站:

```
1  xiaohong = 36.5
2  if xiaohong > 37.2:
3      print("不能够进车站")
4  else:
5      print("请进站")
```

2.3 多重判断

(1) 作用：

让程序根据条件选择性地执行语句。

(2) 语法：

```
1  if 条件1:
2      语句块1
3  elif 条件2:
4      语句块2
5  else:
6      语句块3
```

(3) 说明：

elif 子句可以有0个或多个。

else 子句可以有0个或1个，且只能放在if语句的最后。

思考：新冠疫苗的注射有条件，必须是年龄大于18岁并且小于60岁的成年人，也就是说当年龄小于18岁或者年龄大于60岁，都不能够接种疫苗

```
1  if 条件1:
2      条件1成立执行的代码1
3      条件1成立执行的代码2
4      .....
5  elif 条件2:
6      条件2成立执行的代码1
7      条件2成立执行的代码2
8      .....
9  .....(别的elif语句)
10 else:
11     以上条件都不成立执行的代码
```

多重判断也可以和else配合使用。一般else放到整个if语句的最后，表示以上条件都不成立的时候执行的代码。

实例：

```
1  age = int(input('请输入您的年龄：'))
2  if 0 < age < 18:
3      print(f'您的年龄是{age},暂时不能接种疫苗')
4  elif (age >= 18) and (age <= 60):
5      print(f'您的年龄是{age},可以正常接种疫苗')
6  elif age > 60:
7      print(f'您的年龄是{age},暂时不能接种疫苗')
```

拓展：age >= 18 and age <= 60 可以化简为 18 <= age <= 60。

练习1：

如果满足 职位是高管 或者 年薪大于500000的 条件 则显示“娶你”

否则显示“继续努力”

练习2:

在终端中输入性别

打印"您好先生" "您好女士" "未知"

练习3:

在终端中输入课程阶段数，显示课程名称

效果:

输入:

输出:

1	Python语言核心编程
2	Python高级软件技术
3	Web 全栈
4	项目实战
5	数据分析、人工智能

练习4:

在终端中录入4个同学身高，打印最高的值

算法:

170 160 180 165

假设第一个就是最大值

使用假设的和第二个进行比较，发现更大的就替换假设的

使用假设的和第三个进行比较，发现更大的就替换假设的

使用假设的和第四个进行比较，发现更大的就替换假设的

最后，假设的就是最大的

效果:

请输入第1个同学身高: 170

请输入第2个同学身高: 160

请输入第3个同学身高: 180

请输入第4个同学身高: 165

最高的同学身高: 180

练习5:

根据心理年龄与实际年龄，打印智商等级

智商IQ = 心理年龄MA 除以 实际年龄CA 乘以 100

天才：140以上（包含）

超常：120-139之间（包含）

聪慧：110-119之间（包含）

正常：90-109之间（包含）

迟钝：80-89之间（包含）

低能：80以下

练习6：

在终端中输入月份，打印相应的天数

1 3 5 7 8 10 12 有 31天

2 有 29天

4 6 9 11 有 30天

超过月份提示月份有误

效果：

请输入月份：10

31天

2.4 if嵌套

语法：

```
1  if 条件1:
2      条件1成立执行的代码
3      条件1成立执行的代码
4
5      if 条件2:
6          条件2成立执行的代码
7          条件2成立执行的代码
8
```

注意：条件2的if也是处于条件1成立执行的代码的缩进关系内部。

实施：

- 判断用不用隔离，做核算检测

```
1  """
2  1. 如果体温超过37.2，则需要隔离，做核算检测
3  2. 核算报告呈阳性，需要接受治疗
4  3. 当地新冠患者超过100，为高风险地区
5  超过50小于100为中风险地区
6  小于50为低风险地区
```

```

7         核酸报告呈阴性，则可以回家
8     如果体温没超过37.2，则不需要隔离
9     """
10    # 假设用 test_report = 1 表示阳性，test_report = 0 表示阴性
11    if temperature >= 37.2:
12        print('发烧，需要进行隔离，做核算检测')
13    else:
14        print('不需要隔离')

```

- 判断核酸报告是否呈阳性

```

1    """
2    1. 如果体温超过37.2，则需要隔离，做核算检测
3        2. 核算报告呈阳性，需要接受治疗
4            3. 当地新冠患者超过100，为高风险地区
5                超过50小于100为中风险地区
6                小于50为低风险地区
7        核酸报告呈阴性，则可以回家
8    如果体温没超过37.2，则不需要隔离
9    """
10   # 假设用 test_report = 1 表示阳性，test_report = 0 表示阴性
11   if temperature >= 37.2:
12       print('发烧，需要进行隔离，做核算检测')
13       if test_report == 1:
14           print('核算报告呈阳性，需要接受治疗')
15       elif test_report == 0:
16           print('核算报告呈阴性，可以回家')
17   else:
18       print('不需要隔离')

```

- 判断当地是否为高风险地区

```

1    """
2    1. 如果体温超过37.2，则需要隔离，做核算检测
3        2. 核算报告呈阳性，需要接受治疗
4            3. 当地新冠患者超过100，为高风险地区
5                超过50小于100为中风险地区
6                小于50为低风险地区
7        核酸报告呈阴性，则可以回家
8    如果体温没超过37.2，则不需要隔离
9    """
10   # 假设用 test_report = 1 表示阳性，test_report = 0 表示阴性
11   if temperature >= 37.2:
12       print('发烧，需要进行隔离，做核算检测')
13       if test_report == 1:
14           print('核算报告呈阳性，需要接受治疗')
15           if count > 100:
16               print('该地区为高风险地区')
17           elif 50 < count < 100:
18               print('该地区为中风险地区')
19           elif 0 < count < 50:
20               print('该地区为低风险地区')
21       else:
22           print('该地区为安全区')

```

```
23     elif test_report == 0:
24         print('核算报告呈阴性，可以回家')
25     else:
26         print('不需要隔离')
```

2.5 三目运算符

三目运算符也叫三元运算符。

语法如下：

```
1  值1 if 条件 else 值2
```

快速体验：

```
1  a = 1
2  b = 2
3
4  c = a if a > b else b
5  print(c)
```

2.6 真值表达式

```
1  if 100:
2      print("真值")
3  # 等同于
4  if bool(100):
5      print("真值")
```

练习：在终端中输入一个整数，如果是奇数为变量state赋值"奇数"，否则赋值"偶数"。

效果：

请输入数字：6

state变量存储的是：偶数

练习：

在终端中输入一个年份，如果是闰年为变量day赋值29，否则赋值28

闰年条件：年份能被4整除但是不能被100整除，或者年份能被400整除

效果：

请输入年份：2020

2020年的2月有29天

2.7 条件表达式

(1) 语法：变量 = 结果1 if 条件 else 结果2

```
1 value = 1 if input("请输入性别：") == "男" else 0
```

(2) 作用：根据条件(True/False) 来决定返回结果1还是结果2。

练习：

将上述两个练习用条件表达式进行书写

总结

- if语句语法

```
1 if 条件：  
2     条件成立执行的代码
```

- if...else...

```
1 if 条件：  
2     条件成立执行的代码  
3 else：  
4     条件不成立执行的代码
```

- 多重判断

```
1 if 条件1：  
2     条件1成立执行的代码  
3 elif 条件2：  
4     条件2成立执行的代码  
5 else：  
6     以上条件都不成立执行的代码
```

- if嵌套

```
1 if 条件1：  
2     条件1成立执行的代码  
3     if 条件2：  
4         条件2成立执行的代码  
5         ....
```


3. 循环语句

3.1 学习目标

- 了解循环
- while语法【重点】
- while应用
- break和continue
- while循环嵌套【重点】
- while循环嵌套应用【难点】
- for循环

3.2 循环简介

1、循环的作用

思考：疫情期间，我们需要提醒所有人做好防疫工作，出门戴好口罩。假如小王出门忘带口罩了，他的妈妈为了让他以后记住，罚他写100遍出门要戴口罩。这个时候程序员会怎么做？

答：100遍 `print('出门戴口罩')`

思考：复制粘贴100次吗？

答：重复执行100次一样的代码，程序中循环即可

循环的作用：让代码更高效的重复执行。

2、循环的分类

在Python中，循环分为 `while` 和 `for` 两种，最终实现效果相同。

3.3 while语句

(1) 作用：

可以让一段代码满足条件，重复执行。

(2) 语法：

```
1 while 条件:
2     # 满足条件执行的语句
3 else:
4     # 不满足条件执行的语句
```

(3) 说明：

`else`子句可以省略。

在循环体内用`break`终止循环时，`else`子句不执行。

```
1 count = 0 # 1. 开始
2 while count < 3: # 2. 结束
3     print(count) # 0 1 2
4     count += 1 # 3. 间隔
```

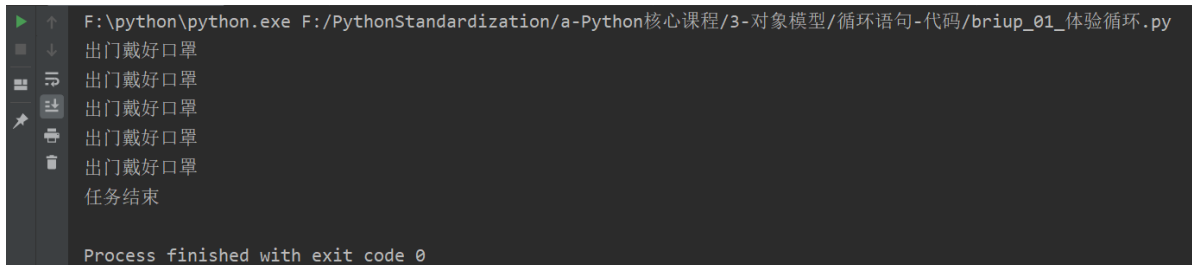
(4) 快速体验

需求：复现重复执行100次 `print('出门戴好口罩')`（输出更简洁一些，我们这里设置5次）。

分析：初始值是0次，终点是5次，重复做的事情输出“出门戴好口罩”。

```
1 # 循环的计数器
2 i = 0
3 while i < 5:
4     print('出门戴好口罩')
5     i += 1
6
7 print('任务结束')
```

执行结果：



```
F:\python\python.exe F:/PythonStandardization/a-Python核心课程/3-对象模型/循环语句-代码/briup_01_体验循环.py
出门戴好口罩
出门戴好口罩
出门戴好口罩
出门戴好口罩
出门戴好口罩
任务结束
Process finished with exit code 0
```

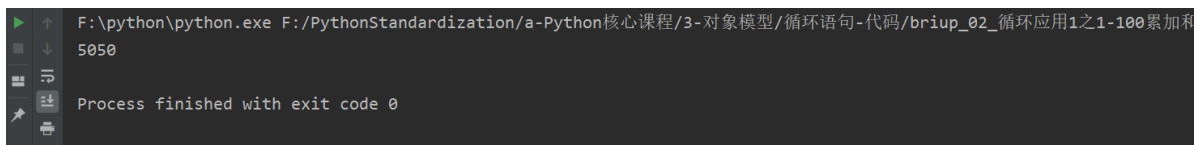
3.4 while的应用

应用一：计算1-100累加和

分析：1-100的累加和，即 $1 + 2 + 3 + 4 + \dots$ ，即前两个数字的相加结果 + 下一个数字(前一个数字 + 1)。

```
1 i = 1
2 result = 0
3 while i <= 100:
4     result += i
5     i += 1
6
7 # 输出5050
8 print(result)
```

执行结果：



```
F:\python\python.exe F:/PythonStandardization/a-Python核心课程/3-对象模型/循环语句-代码/briup_02_循环应用1之1-100累加和
5050
Process finished with exit code 0
```

注意：为了验证程序的准确性，可以先改小数值，验证结果正确后，再改成1-100做累加。

应用二：计算1-100偶数累加和

分析：1-100的偶数和，即 $2 + 4 + 6 + 8 + \dots$ ，得到偶数的方法如下：

方法一：条件判断 和2取余数为0则累加计算

- 偶数即是和2取余结果为0的数字，可以加入条件语句判断是否为偶数，为偶数则累加

```

1  # 方法一：条件判断和2取余数为0则累加计算
2  i = 1
3  result = 0
4  while i <= 100:
5      if i % 2 == 0:
6          result += i
7      i += 1
8
9  # 输出2550
10 print(result)

```

执行结果：

```

Run: briup_03_循环应用2之1-100偶数累加和
F:\python\python.exe F:/PythonStandardization/a-Python核心课程/3-对象模型/循环语句-代码/briup_03_循环应用2之1-100偶数累加和.py
2550
Process finished with exit code 0

```

方法二：计数器控制

- 初始值为0, 计数器每次累加2

```

1  # 方法二：计数器控制增量为2
2  i = 0
3  result = 0
4  while i <= 100:
5      result += i
6      i += 2
7
8  # 输出2550
9  print(result)

```

执行结果：

```

Run: briup_04_循环应用2之1-100偶数累加和
F:\python\python.exe F:/PythonStandardization/a-Python核心课程/3-对象模型/循环语句-代码/briup_04_循环应用2之1-100偶数累加和.py
2550
Process finished with exit code 0

```

练习1：

让下列代码重复执行，输入y继续(不输入y则退出)

```

1  gender = input("请输入性别：")
2  if gender == "男":
3      print("您好先生")
4  elif gender == "女":
5      print("您好女士")
6  else:
7      print("未知")

```

练习2：

在终端中显示0 1 2 3

在终端中显示2 3 4 5 6

在终端中显示1 3 5 7

在终端中显示8 7 6 5 4

在终端中显示-1 -2 -3 -4 -5

练习3:

在终端中循环录入5个成绩

最后打印平均成绩(总成绩除以人数)

效果:

请输入成绩: 98

请输入成绩: 83

请输入成绩: 90

请输入成绩: 99

请输入成绩: 78

平均分: 89.6

练习4:

一张纸的厚度是0.01毫米

请计算, 对折多少次超过珠穆朗玛峰(8844.43米)

思路:

数据: 厚度、高度、次数

算法: 厚度*=2 次数+=1

练习5:

程序产生1个, 1到100之间的随机数。

让玩家重复猜测, 直到猜对为止。

每次提示: 大了、小了、恭喜猜对了, 总共猜了多少次。

效果:

请输入要猜的数字: 50

大了

请输入要猜的数字: 25

小了

请输入要猜的数字：35

大了

请输入要猜的数字：30

小了

请输入要猜的数字：32

恭喜猜对啦，总共猜了5次

3.5 break和continue

break 语句

- (1) 跳出循环体，后面的代码不再执行。
- (2) 可以让while语句的else部分不执行。

continue 语句

- (1) 跳过本次，继续下次循环。

```
1 # 需求：累加1-100之间能被3整除的数字
2 # 思想：不满足条件跳过，否则累加。
3 sum_value = 0
4 for item in range(1, 101):
5     if item % 3 != 0:
6         continue
7     sum_value += item
8 print(sum_value)
```

语法和区别

```
1 #一般形式
2 while <test>:
3     <statements1>
4     if <test1>: break
5     if <test2>: continue
6 else:
7     <statements2>
```

区别：

break：跳出当前循环

continue：跳过continue之后的语句到 循环的起始处

break/continue 只能用在循环中，除此以外不能单独使用

break/continue 在嵌套循环中，只对最近的一层循环起作用

快速体验

举例：一共吃5个苹果，吃完第一个，吃第二个...，这里"吃苹果"的动作是不是重复执行？

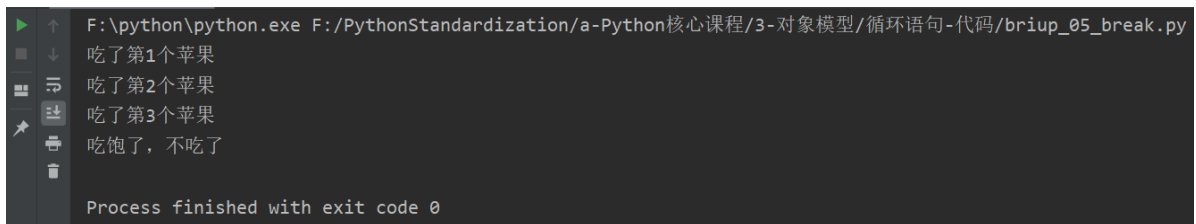
情况一：如果吃的过程中，吃完第三个吃饱了，则不需要再吃第4个和第五个苹果，即是吃苹果的动作停止，这里就是break控制循环流程，即**终止此循环**

情况二：如果吃的过程中，吃到第三个吃出一个大虫子...是不是这个苹果就不吃了，开始吃第四个苹果，这里就是continue控制循环流程，即退出当前一次循环继而执行下一次循环代码。

情况一：break

```
1 i = 1
2 while i <= 5:
3     if i == 4:
4         print('吃饱了不吃了')
5         break
6     print('吃了第%d个苹果' % i)
7     i += 1
```

执行结果：

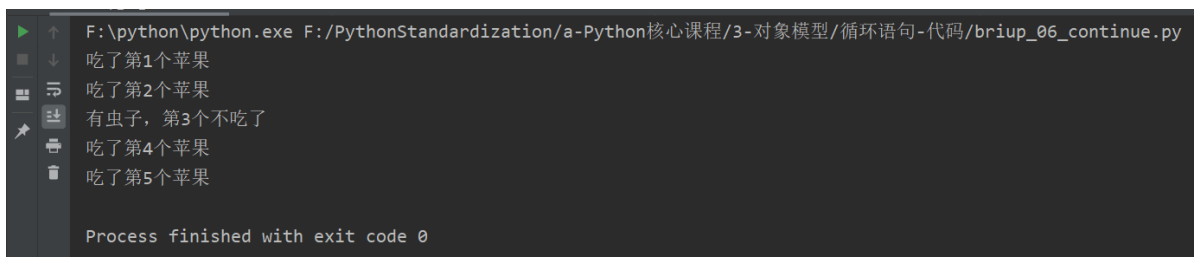


```
F:\python\python.exe F:/PythonStandardization/a-Python核心课程/3-对象模型/循环语句-代码/briup_05_break.py
吃了第1个苹果
吃了第2个苹果
吃了第3个苹果
吃饱了, 不吃了
Process finished with exit code 0
```

情况二：continue

```
1 i = 1
2 while i <= 5:
3     if i == 3:
4         print('有虫子, 第%d个不吃了' % i)
5         # 在continue之前一定要修改计数器, 否则会陷入死循环
6         i += 1
7         continue
8     print('吃了第%d个苹果' % i)
9     i += 1
```

执行结果：



```
F:\python\python.exe F:/PythonStandardization/a-Python核心课程/3-对象模型/循环语句-代码/briup_06_continue.py
吃了第1个苹果
吃了第2个苹果
有虫子, 第3个不吃了
吃了第4个苹果
吃了第5个苹果
Process finished with exit code 0
```

练习：累加10 -- 60之间，个位不是3/5/8的整数和。

3.6 while循环嵌套

语法

```

1 while 条件1:
2     条件1成立执行的代码
3     .....
4     while 条件2:
5         条件2成立执行的代码
6         .....

```

总结：所谓while循环嵌套，就是一个while里面嵌套一个while的写法，每个while和之前的基础语法是相同的。

快速体验

举例：假如小王写了3遍出门要戴口罩了，妈妈告诉他，还要写上要注意个人卫生。要连续写3天

代码：

```

1 j = 0
2 while j < 3:
3     i = 0
4     while i < 3:
5         print('出门要戴口罩')
6         i += 1
7     print('要注意个人卫生')
8     print('-----')
9     j += 1

```

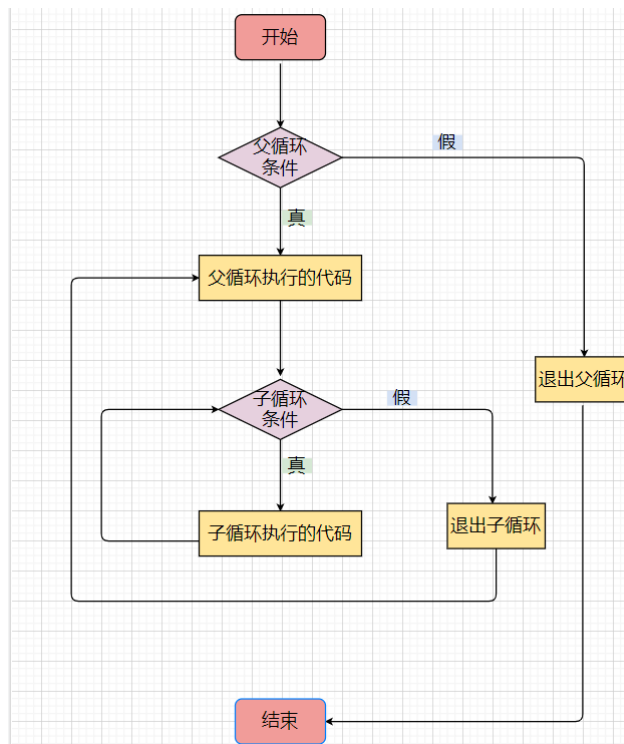
执行结果：

```

F:\python\python.exe F:/PythonStandardization/a-Python核心课程/3-对象模型/循环语句-代码/briup_07_while循环嵌套快速体验.py
出门要戴口罩
出门要戴口罩
出门要戴口罩
要注意个人卫生
-----
出门要戴口罩
出门要戴口罩
出门要戴口罩
要注意个人卫生
-----
出门要戴口罩
出门要戴口罩
出门要戴口罩
要注意个人卫生
-----
Process finished with exit code 0

```

理解执行流程：



3.7 while循环嵌套应用

应用一：打印星号(正方形)

需求：

```
1 *****
2 *****
3 *****
4 *****
5 *****
```

代码：

分析：一行输出5个星号，重复打印5行

```
1 # 重复打印5行星星
2 j = 0
3 while j <= 4:
4     # 一行星星的打印
5     i = 0
6     while i <= 4:
7         # 一行内的星星不能换行，取消print默认结束符\n
8         print('*', end='')
9         i += 1
10    # 每行结束要换行，这里借助一个空的print，利用print默认结束符换行
11    print()
12    j += 1
```

应用二：打印星号(三角形)

需求：


```
1 *
2 **
3 ***
4 ****
5 *****
```

代码:

分析: 一行输出星星的个数和行号是相等的, 每行: 重复打印行号数字个星号, 将打印行星号的命令重复执行5次实现打印5行。

```
1 # 重复打印5行星星
2 # j表示行号
3 j = 0
4 while j <= 4:
5     # 一行星星的打印
6     i = 0
7     # i表示每行里面星星的个数, 这个数字要和行号相等所以i要和j联动
8     while i <= j:
9         print('*', end='')
10        i += 1
11    print()
12    j += 1
```

九九乘法表

代码:

```
1 # 重复打印9行表达式
2 j = 1
3 while j <= 9:
4     # 打印一行里面的表达式 a * b = a*b
5     i = 1
6     while i <= j:
7         print(f'{i}*{j}={j*i}', end='\t')
8         i += 1
9     print()
10    j += 1
```

执行结果:

```
F:\python\python.exe F:/PythonStandardization/a-Python核心课程/3-对象模型/循环语句-代码/briup_10_while循环嵌套应用之九九乘法表.py
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

Process finished with exit code 0
```

3.8. for循环

(1) 作用:

用来遍历可迭代对象的数据元素。

可迭代对象是指能依次获取数据元素的对象，例如：容器类型。

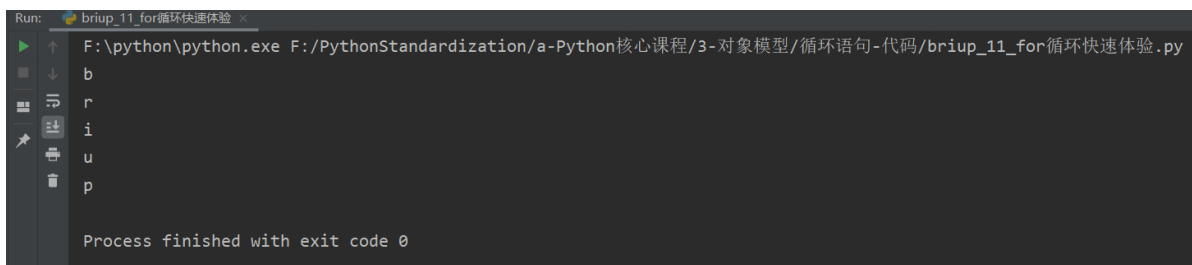
(2) 语法

```
1 for 临时变量 in 序列:
2     重复执行的代码1
3     重复执行的代码2
4     .....
```

(3) 快速体验

```
1 str1 = 'briup'
2 for i in str1:
3     print(i)
```

执行结果:

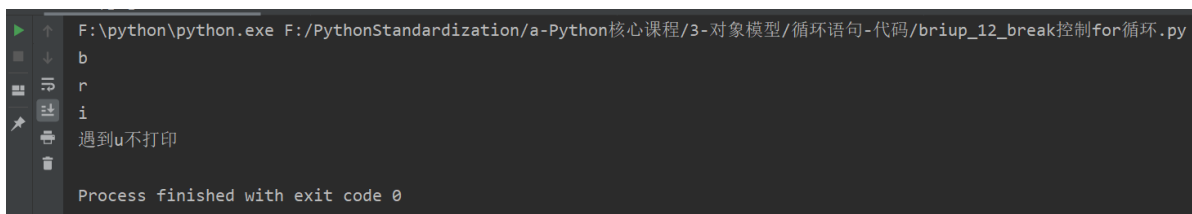


```
Run: briup_11_for循环快速体验 x
F:\python\python.exe F:/PythonStandardization/a-Python核心课程/3-对象模型/循环语句-代码/briup_11_for循环快速体验.py
b
r
i
u
p
Process finished with exit code 0
```

(4) break应用

```
1 str1 = 'briup'
2 for i in str1:
3     if i == 'u':
4         print('遇到u不打印')
5         break
6     print(i)
```

执行结果:



```
Run: F:\python\python.exe F:/PythonStandardization/a-Python核心课程/3-对象模型/循环语句-代码/briup_12_break控制for循环.py
b
r
i
遇到u不打印
Process finished with exit code 0
```

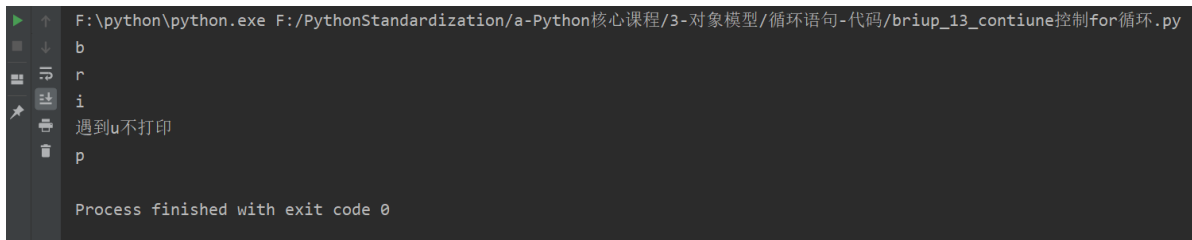
(5) continue应用

```

1  str1 = 'briup'
2  for i in str1:
3      if i == 'u':
4          print('遇到u不打印')
5          continue
6      print(i)

```

执行结果：



```

F:\python\python.exe F:/PythonStandardization/a-Python核心课程/3-对象模型/循环语句-代码/briup_13_contiune控制for循环.py
b
r
i
u
p
遇到u不打印
Process finished with exit code 0

```

练习：

在终端中输入任意整数，计算累加和。

"1234" -> "1" -> 累加 1

效果：

请输入一个整数：12345

累加和是 15

3.9. else

循环可以和else配合使用，else下方缩进的代码指的是-----当循环正常结束之后要执行的代码。

while...else

```

1  i = 1
2  while i <= 5:
3      print('出门要戴口罩')
4      i += 1
5  print('小王是个听话的孩子...')

```

思考：这个print是不是没有循环也能执行？

语法：

```

1  while 条件:
2      条件成立重复执行的代码
3  else:
4      循环正常结束之后要执行的代码

```

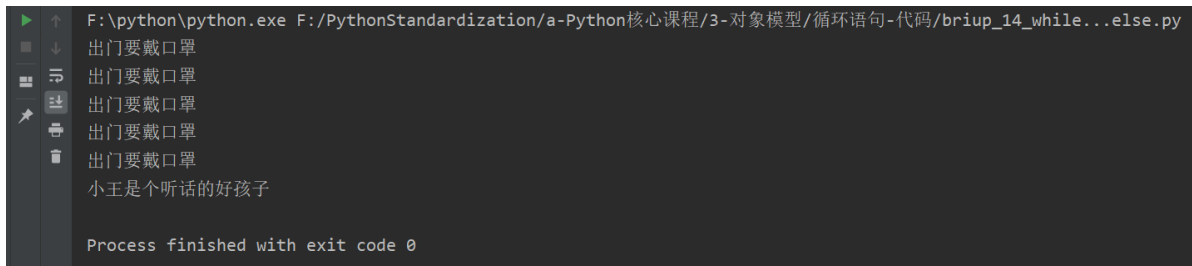
示例：

```

1 i = 1
2 while i <= 5:
3     print('出门要戴口罩')
4     i += 1
5 else:
6     print('小王是个听话的好孩子')

```

执行结果：



```

F:\python\python.exe F:/PythonStandardization/a-Python核心课程/3-对象模型/循环语句-代码/briup_14_while...else.py
出门要戴口罩
出门要戴口罩
出门要戴口罩
出门要戴口罩
出门要戴口罩
小王是个听话的好孩子
Process finished with exit code 0

```

退出循环的方式：

需求：妈妈要求说5遍：出门要戴口罩。说到第三遍的时候，妈妈说这一遍说的不认真，是不是就是要退出循环了？这个退出有两种可能性：

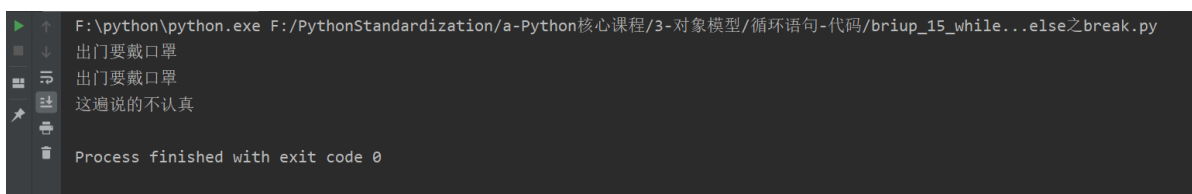
- 觉得小王在敷衍，不让他说了。程序如何书写？
- 只一遍不认真，可以忍受，继续下一遍，程序如何书写？

1.break

```

1 i = 1
2 while i <= 5:
3     if i == 3:
4         print('这遍说的不认真')
5         break
6     print('出门要戴口罩')
7     i += 1
8 else:
9     print('小王是个听话的好孩子! ')

```



```

F:\python\python.exe F:/PythonStandardization/a-Python核心课程/3-对象模型/循环语句-代码/briup_15_while...else之break.py
出门要戴口罩
出门要戴口罩
这遍说的不认真
小王是个听话的好孩子!
Process finished with exit code 0

```

2.continue

```

1 i = 1
2 while i <= 5:
3     if i == 3:
4         print('这遍说的不真诚')
5         i += 1
6         continue
7     print('出门要戴口罩')
8     i += 1
9 else:
10    print('小王是个听话的好孩子! ')

```

```
F:\python\python.exe F:/PythonStandardization/a-Python核心课程/3-对象模型/循环语句-代码/briup_16_while...else之continue.py
出门要戴口罩
出门要戴口罩
这遍说的不真诚
出门要戴口罩
出门要戴口罩
小王是个听话的好孩子！
Process finished with exit code 0
```

因为continue是退出当前一次循环，继续下一次循环，所以该循环在continue控制下是可以正常结束的，当循环结束后，则执行了else缩进的代码。

**** for...else****

语法:

```
1  for 临时变量 in 序列:
2      重复执行的代码
3      ...
4  else:
5      循环正常结束之后要执行的代码
```

所谓else指的是循环正常结束之后要执行的代码，即如果是break终止循环的情况，else下方缩进的代码将不执行。

示例:

```
1  str1 = 'briup'
2  for i in str1:
3      print(i)
4  else:
5      print('循环正常结束执行的else的代码')
```

执行结果:

```
F:\python\python.exe F:/PythonStandardization/a-Python核心课程/3-对象模型/循环语句-代码/briup_17_for...else(1).py
b
r
i
u
p
循环正常结束执行的else的代码
Process finished with exit code 0
```

退出循环的方式:

1.break终止循环

```
1  str1 = 'briup'
2  for i in str1:
3      if i == 'u':
4          print('遇到u不打印')
5          break
6      print(i)
7  else:
8      print('循环正常结束之后执行的代码')
```

```
F:\python\python.exe F:/PythonStandardization/a-Python核心课程/3-对象模型/循环语句-代码/briup_18_for...else之break和continue.py
b
r
i
遇到u不打印
Process finished with exit code 0
```

没有执行else缩进的代码。

2.continue控制循环

```
1 str1 = 'briup'
2 for i in str1:
3     if i == 'u':
4         print('遇到u不打印')
5         continue
6     print(i)
7 else:
8     print('循环正常结束之后执行的代码')
```

```
F:\python\python.exe F:/PythonStandardization/a-Python核心课程/3-对象模型/循环语句-代码/briup_18_for...else之break和continue.py
b
r
i
遇到u不打印
p
循环正常结束执行的else的代码
Process finished with exit code 0
```

因为continue是退出当前一次循环，继续下一次循环，所以该循环在continue控制下是可以正常结束的，当循环结束后，则执行了else缩进的代码。

3.10. range 函数

(1) 作用：

用来创建一个生成一系列整数的可迭代对象(也叫整数序列生成器)。

(2) 语法：

range(开始点, 结束点, 间隔)

(3) 说明：

函数返回的可迭代对象可以用for取出其中的元素。

返回的数字不包含结束点。

开始点默认为0。

间隔默认为1。

```
1 # 写法1: range(开始,结束,间隔)
2 # 注意: 不包含结束值
3 for item in range(1, 3, 1):
4     print(item)
5
6 # 写法2: range(开始,结束)
7 # 注意: 间隔默认为1
```

```

8  for item in range(1, 3):
9      print(item)
10
11 # 写法3: range(结束)
12 # 注意: 开始默认为0
13 for item in range(3):
14     print(item)

```

练习:

在终端中累加 0 1 2 3

在终端中累加 2 3 4 5 6

在终端中累加 1 3 5 7

在终端中累加 8 7 6 5 4

在终端中累加 -1 -2 -3 -4 -5

3.11 for循环的应用

应用一：累加10 -- 60之间，个位不是3/5/8的整数和。

应用二：在终端中录入3个人的身高,打印平均身高

应用三：

在终端中获取一个整数，作为边长，打印矩形。

效果：

请输入整数：5

```

1  $$$$
2  $  $
3  $  $
4  $  $
5  $$$$

```

4. 总结

- 循环的作用：控制代码重复执行
- while语法

```

1  while 条件:
2      条件成立重复执行的代码1
3      条件成立重复执行的代码2
4      .....

```

- while循环嵌套语法

```
1 while 条件1:
2     条件1成立执行的代码
3     .....
4 while 条件2:
5     条件2成立执行的代码
6     .....
```

- for循环语法

```
1 for 临时变量 in 序列:
2     重复执行的代码1
3     重复执行的代码2
4     .....
```

- break退出整个循环
- continue退出本次循环，继续执行下一次重复执行的代码
- else
 - while和for都可以配合else使用
 - else下方缩进的代码含义：当循环正常结束后执行的代码
 - break终止循环不会执行else下方缩进的代码
 - continue退出循环的会执行else下方缩进的代码