分支和循环

1.多向条件分支

```
if 条件1:
    【代码段1】
elif 条件2:
    【代码段2】
.............
elif 条件n:
    【代码段n】
else:
   【else语句块】
【后续代码】
```

执行流程：多选一，如果满足条件1，执行【代码段1】，然后跳出if-elif语句，执行【后续代码】，否则判断是否满足条件2，如果满足执行【代码段2】，然后跳出if-elif语句，执行【后续代码】....。如果所有条件都不满足，执行【else语句块】，然后再执行【后续代码】。

注意：else是可选的，可以写也可以不写。

2 循环

> 广义：一个周期现象或者重复出现的情况，这种状态被称为循环
>
> 狭义：在满足条件的情况下，反复执行某一段代码，在编程语言中出现的这种现象被称为循环。被反复执行的这段代码被称为循环体
>
> 当反复执行某段代码时，需要在合适的时机将循环停止下来，否则会产生死循环
>
> Python中提供的循环语句：while语句，for-in语句

2.1 使用

1 while语法

```
while 表达式:
    循环体
```

> 工作原理：遇到while语句时，首先计算表达式的值，如果表达式的值为假，则跳过整个while语句，继续执行下面的代码；如果表达式的值为真，则执行循环体

```
# 计算1-100的和
sum1 = 0
i = 1
while i <= 100:
     sum1 += i
     i += 1
print(sum1,i)
```

2 while-else

```
while 表达式:
    循环体
else:
    【else语句块】
```

```
i = 0
while i < 100:
    print("hello world")
    if i > 50:
        break
    i += 1
else:
    print("循环正常结束")
```

说明：当while语句执行完成之后，执行【else语句块】，如果用break跳出循环则不执行else

3 死循环

> 在循环语句中，表达式永远为真的循环

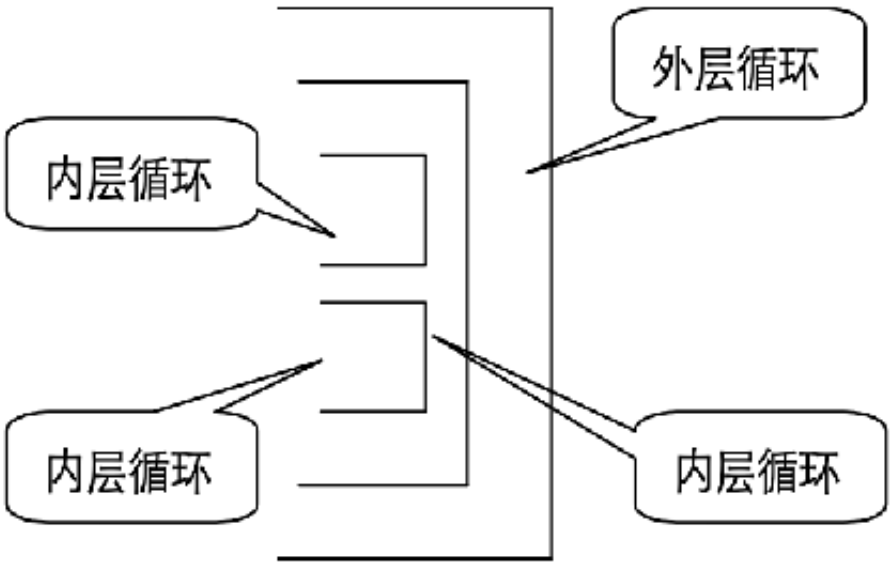
```
while True:
    #循环体

while 1:
    #循环体
```

4 while语句的嵌套

```
while 条件1:      #外层循环
    【语句块A】
    while 条件2:    #内存循环
        【语句B】
```

执行流程：首先判断外层循环的条件1，如果为真，则执行循环体中的【语句块A】，执行内层循环，判断条件2是否成立，如果成立，执行内层循环的【语句块B】，执行完内层循环后，重新判断外层循环条件....

- 特点：外层循环走一步，内层循环执行一遍
- 外层循环和内层循环的循环变量必须不同



合法的嵌套循环

- 内重循环必须完全嵌套到外重循环里面

演示： 打印九九乘法表

```
#行数
i = 1
while i <= 9:
    # 打印每行的内容
    j = 1    # 内循环必须完全嵌入到外重循环里
    while j <= 9:
        print("%d * %d = %2d  "%(i,j,i*j),end='')
        j += 1
    print()
    i += 1
```

1.3. break和continue、exit()、pass

continue 语句用来跳过当前循环的剩余语句，然后重新判断循环条件，开启下一轮循环。continue只能出现在while和for循环中

break用于结束当前循环。只对当前这一重循环起作用。break只能出现在循环中

exit()结束程序

pass是占位符，只是为了保证语法的完整性，本身没有什么实际意义。