化工应用数学 第二章 python 编程基础 讲义

（02.Python 高级用法）

if 条件语句：

```
if condition:
    expressions
```

If 条件语句的基本使用方法如上，如果 condition 的值为 True,将会执行 expressions 语句的内容，否则将跳过该语句往下执行。

注意点：

在 python 语言中等号的判断使用==而不是=，后一种是赋值语句。

```
if x=y:
    print('x is equal to y')
```
错误

```
if x==y:
    print('x is equal to y')
```
正确

当需要同时处理相等和不等的时候，可以使用 if else 语句，基本语法：

```
if condition:
    true-expressions
else:
    false-expressions
```

当 if 判断条件为 True,执行 true_expressions 语句; 否则将执行 else 的内部的 false_expressions。

```
if condition1:
    expression1
elif condition2:
    expression2
elif condition3:
    expression3
elif...
    ...
else:
    expression-else
```

三目操作符：condition? value1: value2 （其他语言）

Python： value1 if condition else value2

如果 condition 的值为真，那么上式返回值为 value1，否则上式的返回值为 value2。

如果有多个判断条件，那可以通过 elif 语句添加多个判断条件，一旦某个条件为 True，那么将执行对应的 expression。 并在之代码执行完毕后跳出该 if-elif-else 语句块，往下执行。

while 循环语句：

python 语言中循环语句的语法有两个，分别是 while 和 for 语句，首先介绍 while 循环语句。

```
while condition:
    expressions
```

如果 condition 判断为真，那么程序将执行 expressions 语句；否则的话将跳过这部分继续执行下面的语句。

Condition 的判断：

1.如果 condition 部分是数字，那么如果这个数字等于 0 或者 0.0 则判断为假，否则为真；

2.如果 condition 部分为 None，则判断为假；

3.如果 condition 部分是集合类型，如果集合中元素数量为 0，则判断为假，否则为真。

for 循环语句：

```
for item in sequence:
    expressions
```

For 循环的基本语法如上，与 c 语言中的 for 语句不同，python 中的 for 循环会一个个循环可迭代对象（sequence）中的每一个对象（item）。

range 函数：

1.range(start, stop)

其中 start 将会是序列的起始值，stop 为结束值，但是不包括该值，类似 数学中的表达 [start, stop),左边为闭区间，右边为开区间。

2.range(stop)

如果省略了 start 那么将从 0 开始，相当于 range(0, stop)

3.range(start, stop, step)

step 代表的为步长，即相隔的两个值得差值。从 start 开始，依次增加 step 的值，直至等于或者大于 stop

break 语句：

与 c 语言相同，Python 中 break 语句，是打破最小封闭 for 或 while 循环。

即使循环条件没有 False 条件或者序列还没被完全递归完，也会停止执行循环语句。在嵌套循环中，break 语句将停止执行最深层的循环，并开始执行下一行代码。

continue 语句：

continue 在 for 或 while 循环中仅用来跳出本次循环步，而不像 break 那样跳出整个循环；continue 会跳过当前循环步的剩余语句，直接进行下一轮循环。

pass 语句：

空语句，仅仅为了保持程序结构的完整性，不产生任何效果，一般只是作为站位语句。

pythonic：

Python 之禅：简练，明确，优雅，绝大部分时候执行效率高

def 函数：

有时候我们为了实现某些功能的代码需要在程序代码中重复使用，这时候我们不能在代码中到处粘贴这些代码，因为这样做违反了软件工程中 DRY（don't repeat yourself）原则。针对这一问题，Python 提供了函数功能，可以将我们这部分功能抽象成一个函数以方便程序调用，或者提供给其他模块使用。

```
def function-name(parameters):
    expressions
```

定义函数的基本语法如上，Python 使用 def 开始函数定义，紧接着是函数名，括号内部为函数的参数，内部为函数的具体功能实现代码，如果想要函数有返回值，在 expressions 中的逻辑代码中用 return 返回。

比如，我们可以使用下面的方式来定义一个函数：

```
def exp(a,b):
        return a**b
```

然后`result=exp(2,3)`来调用它，但是要注意，函数参数的使用有很多的不同，上面的方式是一种简化的方式，而下面这些参数的使用其实都是等价的，它们会输出同样的结果：

```
print(result)
print(exp(2,3))
print((exp(a=2,b=3)))
print((exp(b=3,a=2)))
```

全局&局部变量：

在 def 函数中，我们可以定义一个局部变量，这个变量 a 只能在这个函数中

有效，出了这个函数，a 这个变量就不是那个局部的 a。

如果要声明一个函数中的变量是外部变量，即在外部也能使用这个变量，这时就需要声明这个变量为全局变量，方式为在变量前加 global，如 global a。

作为一个案例，如果我们运行下面的程序：

```
a=100
b=100

def fun():
        a=0
        global b
        b=0
        print('a in fun is',a)
        print('b in fun is',b)
fun()
print('a is',a)
print('b is',b)
```

会得到这样的结果：

```
a in fun is 0
b in fun is 0
a is 100
b is 0
```

可以看出，函数并不能改变内部局部变量在外部相同名字变量的值；而如果我们声明了全局变量，在函数内就可以对其进行操作。