python 基础 07：函数

函数最重要的目的是方便我们重复使用相同的一段程序，将一些操作隶属于一个函数，以后你想实现相同的操作的时候，只用调用函数名就可以，而不需要重复敲所有的语句。

1、函数的定义

首先，我们要定义一个函数, 以说明这个函数的功能。

```
def square_sum(a,b):        c = a**2 + b**2        return c
```

这个函数的功能是求两个数的平方和。

首先，def，这个关键字通知 python：我在定义一个函数。square_sum 是函数名。

括号中的 a, b 是函数的参数，是对函数的输入。参数可以有多个，也可以完全没有（但括号要保留）。

我们已经在循环和选择中见过冒号和缩进来表示的隶属关系。

```
c = a**2 + b**2        # 这一句是函数内部进行的运算
return c               # 返回 c 的值，也就是输出的功能。Python 的函数允许不返回值，也就是不用 return。
```

return 可以返回多个值，以逗号分隔。相当于返回一个 tuple(定值表)。

```
return a,b,c          # 相当于 return (a,b,c)
```

在 Python 中，当程序执行到 return 的时候，程序将停止执行函数内余下的语句。return 并不是必须的，当没有 return, 或者 return 后面没有返回值时，函数将自动返回 None。None 是 Python 中的一个特别的数据类型，用来表示什么都没有，相当于 C 中的 NULL。None 多用于关键字参数传递的默认值。

函数调用和参数传递

定义过函数后，就可以在后面程序中使用这一函数

print square_sum(3,4)

Python 通过位置，知道 3 对应的是函数定义中的第一个参数 a， 4 对应第二个参数 b，然后把参数传递给函数 square_sum。

（Python 有丰富的参数传递方式，还有关键字传递、表传递、字典传递等，基础教程将只涉及位置传递）

函数经过运算，返回值 25, 这个 25 被 print 打印出来。

我们再看下面两个例子

a = 1 def change_integer(a): a = a + 1 return a print change_integer(a)print a #===(Python 中 "#" 后面跟的内容是注释，不执

行） b = [1,2,3] def change_list(b): b[0] = b[0] + 1 return b print change_list(b)print b

第一个例子，我们将一个整数变量传递给函数，函数对它进行操作，但原整数变量 a 不发生变化。

第二个例子，我们将一个表传递给函数，函数进行操作，原来的表 b 发生变化。

对于基本数据类型的变量，变量传递给函数后，函数会在内存中复制一个新的变量，从而不影响原来的变量。（我们称此为值传递）

但是对于表来说，表传递给函数的是一个指针，指针指向序列在内存中的位置，在函数中对表的操作将在原有内存中进行，从而影响原有变量。（我们称此为指针传递）

2、总结

def function_name(a,b,c): statement return something # return 不是必须的

函数的目的： 提高程序的重复可用性。

return None

通过位置，传递参数。

基本数据类型的参数：值传递

表作为参数：指针传递

练习:

写一个判断闰年的函数，参数为年、月、日。若是是闰年，返回 True。