初识函数

1

初识函数

—— 函数基础

函数基本概念

- 函数是对程序逻辑进行结构化或过程化的一种编程方法
- 将整块代码巧妙地隔离成易于管理的小块
- 把重复代码放到函数中而不是进行大量的拷贝，这样既能节省空间，也有助于保持一致性
- 通常函数都是用于实现某一种功能

创建函数

- 函数是用def语句来创建的，语法如下：

```
def function_name(arguments):
    "function_documentation_string"
    function_body_suite
```

- 标题行由def关键字，函数的名字，以及参数的集合（如果有的话）组成
- def子句的剩余部分包括了一个虽然可选但是强烈推荐的文档字串，和必需的函数体

调用函数

- 同大多数语言相同，python用一对圆括号调用函数
- 如果没有加圆括号，只是对函数的引用

```
>>> def foo():
...     print('hello')
...
>>> foo()
hello
>>> foo
<function foo at 0x7ff2328967d0>
```

函数的返回值

- 多数情况下，函数并不直接输出数据，而是向调用者返回值
- 函数的返回值使用return关键字
- 没有return的话，函数默认返回None

```
>>> def foo():
...         res = 3 + 4
>>> i = foo()
>>> print i
None
```

2

初识函数

—— 函数参数

定义参数

- 形式参数
  - 函数定义时，紧跟在函数名后（圆括号内）的参数被称为形式参数，简称形参。由于它不是实际存在变量，所以又称虚拟变量
- 实际参数
  - 在主调函数中调用一个函数时，函数名后面括弧中的参数（可以是一个表达式）称为“实际参数”，简称实参

传递参数

- 调用函数时，实参的个数需要与形参个数一致
- 实参将依次传递给形参

```
>>> def foo(x, y):
...         print('x=%d, y=%d' % (x, y))
>>> foo()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: foo() takes exactly 2 arguments (0 given)
>>> foo(3)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: foo() takes exactly 2 arguments (1 given)
>>> foo(3, 4)
x=3, y=4
```

位置参数

- 与shell脚本类似，程序名以及参数都以位置参数的方式传递给python程序
- 使用sys模块的argv列表接收

```
[root@zzghost1 day02]# vim args.py
#!/usr/bin/env python3
import sys
print sys.argv

[root@zzghost1 day02]# ./args.py hello world
['./args.py', 'hello', 'world']
```

默认参数

- 默认参数就是声明了默认值的参数
- 因为给参数赋予了默认值，所以在函数调用时，不向该参数传入值也是允许的

```
>>> def pstar(num = 30):
...         print '*' * num
...
>>> pstar()
******************************
>>> pstar(40)
****************************************
```

更多精彩...

http://bj.linux.tedu.cn/
企业QQ： 86198501