Python程序设计

第六讲 函数与模块化

变量的作用域

张 华
WHU

变量的作用域

- 作用域的概念
- 局部变量
- 全局变量
- global
- globals()和locals()
- 函数的嵌套定义
- nonlocal

变量的作用域

变量的作用域

- 变量起作用的代码范围称为变量的作用域，即可以引用该变量的区域。
- 不同作用域内变量名可以相同，互不影响。

局部变量

- 在函数内部定义的普通变量（包括形参）只在函数内部起作用，称为局部变量。
- 当函数执行结束后，局部变量自动删除，不再可以使用。
- 局部变量的引用比全局变量速度快，应优先考虑使用。

局部变量

局部变量

- 局部变量的作用域只是该函数内部，所以不同的函数中可以有相同名称的变量，它们在各自的函数中互不干扰。

```
def f1():
    x=5
    y=6                    #f1()中的y和f2()中的y互不干扰
    print(x+y)

def f2():
    y=1
    print(x+y)             #出错！不能引用f1()中的x

f1()
f2()
```

全局变量

全局变量

- 在一个源代码文件中，在函数和类定义之外声明的变量。
- 全局变量的作用域为其定义所在的模块，从定义的位置起，直到文件结束位置。
- 如果在函数中定义的局部变量与全局变量同名，则局部变量屏蔽全局变量。

```
x = 'outside'
y = 'global'
def f():
      x = 'inside'
      print(x)
      print(y)
f()
print(x)
```

```
inside
global
outside
```

全局变量

全局变量

- 全局变量可以通过关键字global来定义。这分为两种情况：
  - 一个变量已在函数外定义，如果在函数内需要为这个变量赋值，并要将这个赋值结果反映到函数外，可以在函数内使用global将其声明为全局变量。
  - 如果一个变量在函数外没有定义，在函数内部也可以用global直接将一个变量定义为全局变量，该函数执行后，将增加一个新的全局变量。

```
>>> def demo():
    global x
    x = 3
    y = 4
    print(x,y)

>>> x = 5
>>> demo()
3  4
>>> x
3
```

全局变量

全局变量

- 在函数体中，可以引用全局变量，但是，如果函数内部任意位置只要有为变量赋值的操作，该变量在这个作用域内就是局部变量，除非使用global进行了声明。

```
>>> x = 3
>>> def f():
...     print(x)
...     x = 5
...     print(x)
...
>>> f()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "<stdin>", line 2, in f
UnboundLocalError: local variable 'x' referenced before assignment
```

全局变量和局部变量

- **输出全局变量和局部变量**
  - 使用内置函数**globals()**和**locals()**，可以查看并输出局部变量和全局变量列表。

函数的嵌套定义

- Python允许函数的嵌套定义，在函数内部可以再定义另外一个函数。

```
def my_map(iterable, op, value):          #自定义函数
    if op not in '+-*/':
        return 'Error operator'
    def nested(item):                     #嵌套定义函数
        return eval(repr(item)+op+repr(value))
    return map(nested, iterable)          #使用在函数内部定义的函数

print(list(my_map(range(5), '+', 5)))#调用外部函数，不需要关心其内部实现
print(list(my_map(range(5), '-', 5)))
```

```
[5, 6, 7, 8, 9]
[-5, -4, -3, -2, -1]
```

函数的嵌套定义

非局部语句nonlocal

- 在嵌套函数中，如果要为定义在上级函数体的局部变量赋值，可以使用**nonlocal**语句，表明变量不是所在块的局部变量，而是在上级函数体中定义的局部变量。
- **nonlocal**语句可以指定多个非局部变量。
  - 例如，**nonlocal x, y, z**

函数的嵌套定义

非局部语句nonlocal

示例

```
def outer_func():
    tax_rate = 0.17
    print('outer func tax rate =', tax_rate)
    def inner_func():
        nonlocal tax_rate
        tax_rate = 0.05
        print('inner func tax_rate =', tax_rate)
    inner_func()
    print('outer func tax_rate =', tax_rate)

outer_func()
```

```
outer func tax rate = 0.17
inner func tax_rate = 0.05
outer func tax_rate = 0.05
```