函数

万永权

目录

CONTENTS

1. 函数
2. 递归函数

什么是函数?

函数是组织好的，可重复使用的，用来实现单一或相关联功能的代码段，它能够提高应用的模块性和代码的重复利用率。

```
print(“  *  ”)
print(“ *** ”)
print("*****")
```

如果在一个程序的不同地方需要输出这个图形，每次使用print语言输出的做法显然不可取。

函数的定义和调用

Python定义函数使用def关键字，格式如下：

```
def 函数名（参数列表）：
    函数体
```

函数定义

❖函数定义语法:

```
def 函数名([参数列表]):
    '''注释'''
    函数体
```

1. 函数名可以是任何有效的标识符；
2. 参数可以是零个或者多个；
3. 函数定义中的参数，叫做形参

❖用户进行自定义函数时,需要遵循如下规则:

- 用def关键字定义函数,后接函数名称;
- 即使该函数不需要接收任何参数，也必须保留一对空的圆括号;
- 括号里可以用于定义参数;
- 括号后面的冒号必不可少;
- 并且函数内部语句有缩进;
- 函数必须要先定义,后调用;
- return表示结束函数并返回一个值给调用方,如果函数体内没有return,则相当于返回一个None。
- 函数形参不需要声明类型，也不需要指定函数返回值类型。

函数的定义和调用

这是一个自定义的函数：

```
def printInfo():
    print('------------------------------------')
    print('          生命苦短，我用Python          ')
    print('------------------------------------')
```

函数的定义和调用

定义了函数之后，想要让这些代码能够执行，需要调用函数。通过“函数名()” 即可完成调用。

```
# 调用刚才定义的函数
printInfo()
```

函数的参数

先看一段代码：

```
def add2num():
    c = 11+22
    print(c)
```

这个函数计算的只是固定的两个数，没有什么意义。

函数的参数

如果希望定义的函数，可以计算任何两个数和，我们在定义函数的时候，让函数接收数据，这就是函数的参数。

```
def add2num(a, b):
    c = a+b
    print(c)
```

a和b就是函数的参数，调用的时候，可以传入任何两个数。

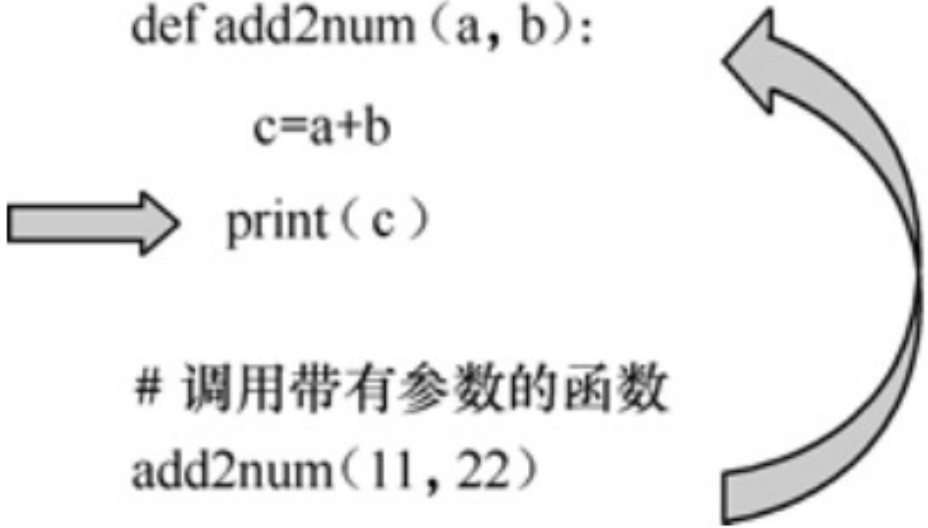
```
add2num(11, 22)
```

函数的参数

参数调用过程

```
# 定义接收 2 个参数的函数
def add2num（a, b）:
    c=a+b
    print（c）

# 调用带有参数的函数
add2num（11, 22）
```



此时：
a=11
b=22
c=33

形参和实参

- 定义：

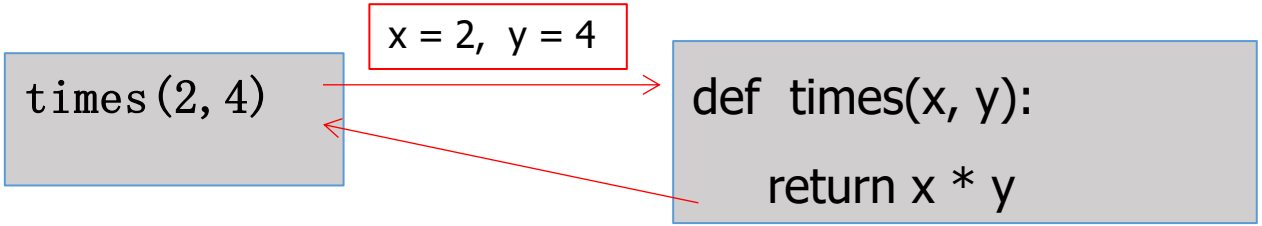
```
def  times(x, y):
        return x * y
```

- 调用：

```
times(2, 4)
```

也可以把函数调用结果赋值给变量

```
result = times(2, 4)
```



在调用过程中， 把实参2和4分别赋值给形参x和y；

函数的返回值

函数的返回值是使用return语句来完成的。

```
def add2num(a, b):
    c = a+b
    return c
```

函数add2num中包含return，
意味着这个函数有一个返回值，
其返回值就是a和b相加的结果。

常用内置函数

dict()	用于创建一个字典
help()	用于查看函数或模块用途的详细说明
dir()	不带参数时，返回当前范围内的变量、方法和定义的类型列表；带参数时，返回参数的属性、方法列表。
hex()	用于将10进制整数转换成16进制，以字符串形式表示
next()	返回迭代器的下一个项目
divmod()	把除数和余数运算结果结合起来，返回一个包含商和余数的元组(a // b，a % b)
id()	用于获取对象的内存地址
sorted()	对所有可迭代的对象进行排序操作
ascii()	返回一个表示对象的字符串，但是对于字符串中的非 ASCII 字符则返回通过 repr() 函数使用 \x, \u 或 \U 编码的字符
oct()	将一个整数转换成8进制字符串
bin()	返回一个整数 int 或者长整数 long int 的二进制表示

常用内置函数（2）

open()	用于打开一个文件
str()	将对象转化为适于人阅读的形式
sum()	对序列进行求和计算
filter()	用于过滤序列，过滤掉不符合条件的元素，返回由符合条件元素组成的新列表
format()	格式化字符串
len()	返回对象（字符、列表、元组等）长度或项目个数
list()	用于将元组转换为列表
range()	返回的是一个可迭代对象（类型是对象）
zip()	用于将可迭代的对象作为参数，将对象中对应的元素打包成一个个元组，然后返回由这些元组组成的对象
compile()	将一个字符串编译为字节代码
map()	根据提供的函数对指定序列做映射
reversed()	返回一个反转的迭代器
round()	返回浮点数x的四舍五入值

案例

- 温度转换
- 定义一个函数，输入一个华氏温度，转换成摄氏度。

```
def f_to_c(f):
    '''
    这是一个工具函数，把华氏度转换成摄氏度
    '''
    c = (f - 32) / 1.8
    return c
```

根据华氏和摄氏温度定义，利用转换公式如下：

$$C = (F - 32) / 1.8$$

$$F = C * 1.8+32$$

其中， C表示摄氏温度， F表示华氏温度

练习

- 定义一个函数，输入一个摄氏温度，转换成华氏度。

递归的概念

- 在数理逻辑和计算机科学中，一种计算过程，如果其中每一步都要用到前一步或前几步的结果，称为递归。
- 用递归过程定义的函数，称为递归函数。

递归的两种形式

```
def  func() :
      ...
      func()
      ...
```

```
def  funcA() :
      ...
      funcB()
      ...
```

```
def  funcB() :
      ...
      funcA()
      ...
```

直接递归调用

间接递归调用

递归的例子-阶乘

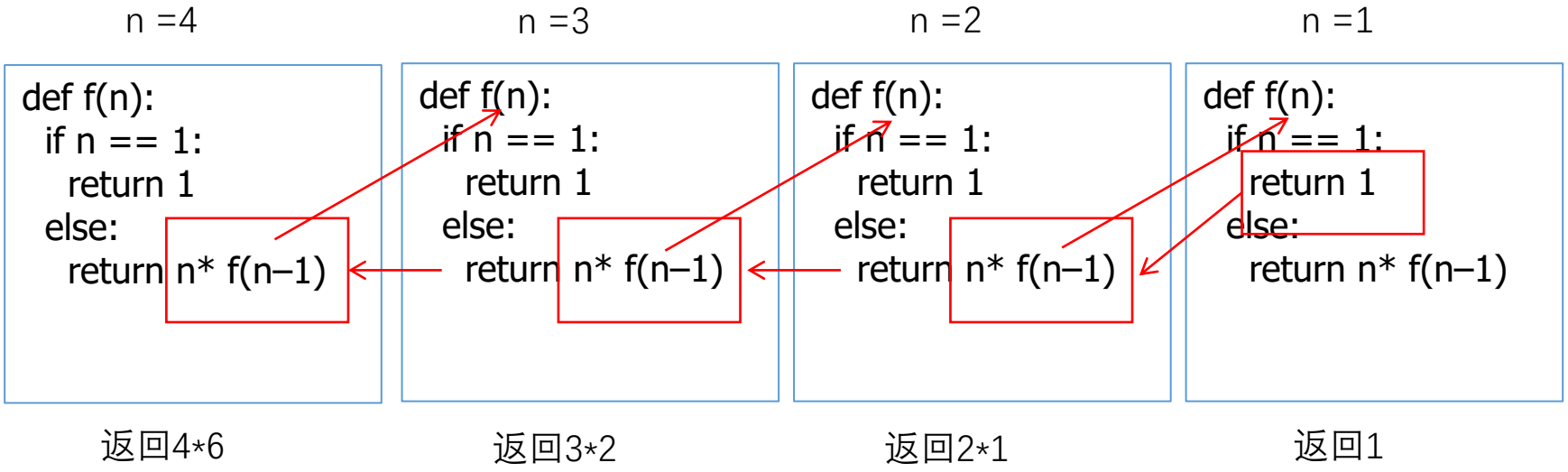
- 阶乘的定义：　　$n! = n * (n-1) * (n-2) * \cdots * 1$
- 阶乘的递归定义：

$$n! = \begin{cases} n * (n-1)! & n > 1 \\ 1 & n = 1 \end{cases}$$

- 递归函数：

```
def factorial(num):
    if num == 1:
        return 1
    else:
        return num * factorial(num – 1)
```

执行过程分析



https://pythontutor.com/

思考

```
def factorial(num):
    if num == 1:
        return 1
    else:
        return num * factorial(num – 1)
```

如果缺少了if num==1:
return 1,会怎么样?

程序会无限递归

递归函数的条件

一个直接或间接调用本函数语句的函数，称为递归函数，它必须满足以下两个条件：

1. 每一次调用自己时，必须是更接近问题的解；
2. 必须有一个终止（退出）处理的条件；

递归VS循环

求阶乘递归算法：

```
def f(n):
  if n == 1:
    return 1
  else:
    return n* f(n–1)
```

求阶乘循环算法：

```
def f(n):
  r = 1
  for i in range(1, n + 1):
    r = r * i
   return r
```

递归：n*(n-1)!, n*(n-1)*(n-2)!,…

自顶向下的计算；

效率低；占用内存多；

符合人的思维过程；

循环：1!, 2!, 3!,…

自底向上的计算；

效率高，符合计算机计算方式；

递归设计的过程

- 问题求解：一个大型复杂的问题能不能分解为一个与原问题相似的规模较小的问题来求解？
- 构建递归模式：对问题分解进行抽象，确定递归的两个条件：
  - 用递归的形式表示，且每一次调用都更接近问题的解；
  - 存在终止递归的条件及终止时的值；
- 递归算法实现：

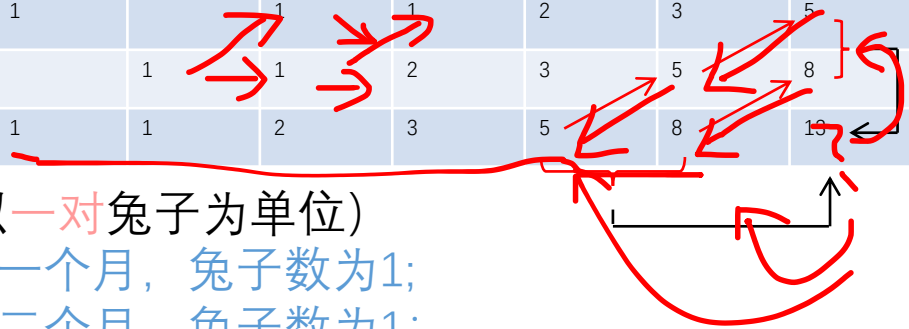
  递归函数的一般格式：

```
def function(参数)
    if 参数满足终止条件 :
        返回直接结果
    else:
        递归调用
```

递归的应用案例

- 斐波那契数列（兔子数列，黄金分割数列）
- 意大利数学家斐波那契提出一个“兔子问题”：假定小兔子一个月可以长成大兔子，而一对大兔子每个月会生出一对小兔子。如果年初养了一对小兔子，问到年底时将有多少对兔子？

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
小兔子	1		1	1	2	3	5	8	13	21	34	55
大兔子		1	1	2	3	5	8	13	21	34	55	89
总数	1	1	2	3	5	8	13	21	34	55	89	144



分析(以一对兔子为单位)

第一个月，兔子数为1;

第二个月，兔子数为1;

第N月 = (N-1)月总数 + N-1月大兔子数

= (N-1)月总数+(N-2)月总数