以下总结Python函数传递参数的5种形式

1. 普通按函数申明的顺序一次传递

例如：

def Test(a,b):

print(a,b)

Test(1,2)

1. 函数定义时默认参数

例如：

def Test(a,b,c=2)

print(a,b,c)

Test(1,2) #此时函数中a=1, b=2, c为默认值2

Test(1,2,6)#此时函数中a=1,b=2, c=6

Note: 规定默认参数必须在最后申明

1. 函数调用时不按函数申明顺序传递，即使直接指定所传递的形参

例如：

def Test(a,b):

print(a,b)

Test(b=2,a=2)

1. \*args 传任意***非命名参数***，放入args所形成的元组中

例如：

def Test(a,b,\*args):

print(a,b,args)

Test(1,2,3,4,5) #传入函数后a=1,b=2,args = (3,4,5)

Note: 函数调用时，再带\*则会形成list再一个个元素放入tuple args中。如果不带\*，则列表会放入函数中当做元组的一个元素

parameter = [1,2,3,4]

Test(1,2,parameter) # args = ([1,2,3,4],)

Test(1,2,\*parameter)# arge = (1,2,3,4)

1. \*\*args传任意***命名参数***，放入args所形成的dict中（函数调用时，再带\*\*则会形成dict再一个个以键值对方式放入dict args中）

例如：

def Test(a,b,\*\*kargs):

print(a,b,kargs)

Test(1,2,name=”Tom”,age=12) # kargs = {“name”:”Tom,”age”:12}

Note: 函数调用时，再带\*\*则会形成dict再一个个元素放入dict kargs中。如果不带\*，则字典会放入函数中当做kargs的一个key

parameter = {“name”:”Tom”,”age”:12}

Test(1,2,parameter = parameter) # kargs = {“parameter”:{“name”:”Tom”,”age”:12”}}

Test(1,2,\*\*parameter) # kargs = {“name”:”Tom”,”age”:12”}

1. \*args,\*\*kargs万能参数，能接受所有命名，非命名参数（先\*args,再\*\*kargs）

例如:

def Test(\*args,\*\*kargs):

print(args,kargs)

Test(1,2,3,p=1,p2=2) # args = (1,2,3) kargs = {“p”:1,”p2”:2}

Note:

1. 在调用函数为函数中的形参kargs赋值时，不可以使用数字作为键，并且键也不能加双引号。虽然一般的字典可以使用数字作为键。

例如:

def Test(a,b,\*\*kargs):

print(a,b,kargs)

Test(1,2,3=”12”) #非法，不可以使用数字作为键

Test(1,2,”p” =”12”) #非法，键也不能加双引号

Test(1,2,p=”12”) # 合法, kargs = {“p” : “12”}

1. 不可以定义函数的时候有两个\*args,\*args1或者两个\*\*kargs,\*\*kargs2的形参。
2. 调用函数的时候，如果有命名实参，命名实参必须在其他位置实参的后面。
3. 调用函数的时候，各命名参数顺序没有关系。
4. 如果函数申明时候，有形参没有默认值，但是该形参前面有\*args或者\*\*kargs吸收参数，调用函数时必须使用命名指定参数。

例如：

def Test(\*args,a,b,c=2):

print(args,a,b,c)

Test(1,2,3,4,5) #非法，1,2,3,4,5都被\*args所吸收，导致后面非缺省参数a,b没有赋值，报错

Test(1,2,a=3,c=5,5) #非法，命名实参必须在其他位置实参的后面

Test(1,2,5, a=3,b=5)#合法，首先命名参数在所有位置实参前面，非缺省参数a,b,被赋值。另外由于缺省参数c在\*args之后，所以形参c永远无法被按位置赋值，除非通过命名参数复制。

1. Python3总str类的format函数，查看format的内置函数定义为：

def format(self,\*args,\*\*kargs):

所以即使万能参数的函数定义。从而

“{0}+{1}+{2}”.format(1,2,3) #将为输出”1+2+3”

“{A}+{B}+{C}”.format(A=”Hello”,B=”World”,C=”!”) #将为输出”Hello World !”

“{0}+{1}+{C}”.format(”Hello”,”World”,C=”!”) #将为输出”Hello World !”