**第二十一章 运算符重载**

**第一部分**

我们都知道，+是进行加法运算的，参与运算的是两个数字，如果要改变+运算法则，让它实现两个矩阵的相加，比如vector1=[1,3],vector2=[3,4],vector1+vector2=[4,7],像这样的，称为运算符重载。

**一、运算符重载的定义**

运算符重载，[就是](https://baike.baidu.com/item/%E5%B0%B1%E6%98%AF)对已有的运算符重新进行定义，[赋予](https://baike.baidu.com/item/%E8%B5%8B%E4%BA%88)其另一种功能，以适应不同的数据类型。

**二、运算符重载的开发实例**

1、重新定义print输出的内容

\_\_str\_\_方法和\_\_init\_\_方法类似，都是一些特殊方法,所以前后都有双下划线,它用来返回对象的[字符串](http://www.iplaypy.com/jichu/str.html)表达式。  
 Python能用print语句输出内建数据类型。有时，程序员希望定义一个类，要求它的对象也能用print语句输出。Python类可定义特殊方法\_\_str\_\_，为类的对象提供一个不正式的字符串表示。如果类的客户程序包含以下语句：print objectOfClass,那么Python会调用对象的\_\_str\_\_方法，并输出那个方法所返回的字符串。

#encoding=utf-8

class stu:

name="czf"

print(stu())

执行结果：

<\_\_main\_\_.stu instance at 0x0000000002DC4C08>

可以看出结果，打印的是这个对象的地址。

如果在类中重新定义\_\_str\_\_,就改变了了print输出内容：

#encoding=utf-8

class stu:

name="czf"

def \_\_str\_\_(self):

return "stu"

print(stu())

结果：

stu

方法\_\_str\_\_是一个特殊方法，它构造并返回stu:类的一个对象的字符串表示。解析器一旦遇到如下语句：print stu()就会执行以下语句：print stu().\_\_str\_\_(),程序如果将stu对象传给内建函数str(如str(stu()),或者为stu对象使用字符串格式化运算符%(例如"%s"%stu()),Python也会调用\_\_str\_\_方法。

2、重载+

#encoding=utf-8

class Vector:

def \_\_init\_\_(self,a,b):

self.a = a

self.b = b

def \_\_str\_\_(self):

return 'Vector (%d, %d)'%(self.a, self.b)

def \_\_add\_\_(self,other):

return Vector(self.a + other.a, self.b + other.b)

v1 = Vector(2,10)

v2 = Vector(5,-2)

print (v1 + v2)

运行结果：

Vector (7, 8)

3、重载-

#encoding=utf-8

class Vector:

def \_\_init\_\_(self,a,b):

self.a = a

self.b = b

def \_\_str\_\_(self):

return 'Vector (%d, %d)'%(self.a, self.b)

def \_\_sub\_\_(self,other):

return Vector(self.a - other.a, self.b - other.b)

v1 = Vector(2,10)

v2 = Vector(5,-2)

print (v1-v2)

**第二部分**

4、重载\*

#encoding=utf-8

class Vector:

def \_\_init\_\_(self,a,b):

self.a = a

self.b = b

def \_\_str\_\_(self):

return 'Vector (%d, %d)'%(self.a, self.b)

def \_\_mul\_\_(self,other):

return Vector(self.a \* other.a, self.b \* other.b)

v1 = Vector(2,10)

v2 = Vector(5,-2)

print (v1\*v2)

5、重载/

#encoding=utf-8

class Vector:

def \_\_init\_\_(self,a,b):

self.a = a

self.b = b

def \_\_str\_\_(self):

return 'Vector (%d, %d)'%(self.a, self.b)

def \_\_div\_\_(self,other):

return Vector(self.a / other.a, self.b / other.b)

v1 = Vector(2,10)

v2 = Vector(5,-2)

print (v1/v2)

6、重载取余

#encoding=utf-8

class Vector:

def \_\_init\_\_(self,a,b):

self.a = a

self.b = b

def \_\_str\_\_(self):

return 'Vector (%d, %d)'%(self.a, self.b)

def \_\_mod\_\_(self,other):

return Vector(self.a % other.a, self.b % other.b)

v1 = Vector(2,10)

v2 = Vector(5,3)

print (v1%v2)

7、重载比较运算符 >,<,==

#encoding=utf-8

class Vector:

def \_\_init\_\_(self,a,b):

self.a = a

self.b = b

def \_\_str\_\_(self):

return 'Vector (%d, %d)'%(self.a, self.b)

def \_\_cmp\_\_(self,other):

if self.a>other.a and self.b>other.b:

return 1

elif self.a<other.a and self.b<other.b:

return -1

else:

return 0

v1 = Vector(5,10)

v2 = Vector(12,10)

print (v1>v2)

8、其他重载

