1.

P3特殊方法支持以下语言架构：

迭代、集合类、属性访问、运算符重载、函数和方法的调用、对象的创建和销毁、字符串表示形式和格式化

【重写一些特殊方法之后，我们就可以使用上述框架了】

2.

#P3 collections的用法，要后补。

collections.namedtuple像它的名字说的，只是个有名字的元组，或者说是**只用少数属性但是没有方法的类**。

3.

#P4关于双循环的列表推导式，后面的先被遍历一圈。简单说，如果是[for f(i,j) for i in List1 for j in List2]。相当于：

for i in List1 :

for j in List2 :

XXXX

4.

如果重写了\_\_getitem\_\_,就可以用[]了，return逻辑上对应那个就好了。

5.

in 可以用在可迭代的类上。

大多时候特殊方法的调用是隐式的，书上例：

for i in x : 的背后是iter(x), iter(x)的背后是x.\_\_iter\_\_

7.

让内置的函数调用特殊方法是最好的方法，这样快。

8.

随意添加特殊方法名称不是明智的

9.

repr方法，字符串表示形式，把一个对象用字符串的形式表示出来以便辨认。repr方法实际就是调用\_\_repr\_\_作为替代。

对比之下，\_\_str\_\_对应str方法，还是不一样的。如果\_\_str\_\_没被实现但是str()被调用，解释器会用\_\_repr\_\_

从程序习惯的角度来说，\_\_repr\_\_应该是准确无歧义的。

10.

P11特殊方法表，有空粘过来。

11.

作者认为不应该把\_\_的方法叫做魔术方法，而是相反，因为用户自己的类型也可以运用这些方法。