类成员包括字段，方法和属性。所以成员修饰符号，可以用于修饰这三个。

1. 在成员前加入\_\_，那么这个成员便是私有的。不可以在外部访问只可以使用类中定义的方法去访问。并且继承的子类都不允许访问，只允许本类的方法访问。

特殊的类成员: （特殊的成员是前后都有\_\_,但是虽然前面有\_\_但是不是私有变量，可以外部调用）

|  |  |
| --- | --- |
| \_\_init\_\_ | 构造方法，当建立实例对象的时候将执行的函数 |
| \_\_del\_\_ | 析构方法，在对象被垃圾回收之前将执行的函数。 |
| \_\_doc\_\_ | 类的文档说明 |
| \_\_module\_\_ | 表示当前操作的对象在哪个模块中 |
| \_\_class\_\_ | 表示当前操作的对象在哪个类名中 |
| \_\_call\_\_ | 当对一个对象进行调用时，将执行的方法。例如:  Obj = Example()  Obj() #此时调用类中定义的\_\_call\_\_方法 |
| \_\_str\_\_ | 当对象进行print(obj)操作，或者str(obj)操作的时候将输出的字符串（默认情况下将会直接输出对象的内存地址） |
| \_\_dict\_\_ | 返回对象中的所有字段并且以字典的形式返回  或者返回类中的所有定义的方法并以字典形式返回 |
| \_\_getitem\_\_ | 当对象执行obj[key] 或者obj[index:index] 的时候，即是以中括号对对象进行取值的时候，将执行类中\_\_getitem\_\_的方法。如果我要以key或者[index:index]都能进行取值，那么我们在\_\_getitem\_\_中要进行判断而已。但是调用的都是用\_\_getitem\_\_方法。  (例如Python中的dict便是Dict类的对象，其中便定义了\_\_getitem\_\_方法，\_\_setitem\_\_,\_\_delitem\_\_方法) |
| \_\_setitem\_\_ | 当对象执行obj[key] = value 或者obj[index:index] = list的时候，即是以中括号对对象进行赋值的时候，将执行类中\_\_setitem\_\_的方法。如果我要以key或者[index:index]都能进行赋值，那么我们在\_\_setitem\_\_中要进行判断而已。但是调用的都是用\_\_setitem\_\_方法。  (例如Python中的dict便是Dict类的对象，其中便定义了\_\_getitem\_\_方法，\_\_setitem\_\_,\_\_delitem\_\_方法) |
| \_\_delitem\_\_ | \_\_delitem\_\_和\_\_setitem\_\_和\_\_getitem\_\_ 的原理一样。 |
| \_\_iter\_\_ | 如果我们想要对一个对象进行循环，例如for i in obj, 那么便要求这个obj需要有返回生成器的能力。如果没有则报错。当对一个对象进行循环时，变会寻找类中的\_\_iter\_\_方法去寻找生成器。（Python自带的可循环对象，list, dict,set类中便有定义\_\_iter\_\_方法） |