哈希表本质是一个空间换时间的存储结构。

Python中可哈希的对象是不可变对象，包括数字类型(int, float, bool)，字符串str，元组tuple，不可变集合frozenset，自定义类的实例。

不可变对象之所以能可哈希，原因是：对于不可变对象而言，不同的值存储在不同的内存，相同的值存储在相同的内存，用不可变对象作为key时，不同的key对应的是不同的值，满足哈希表的特性。

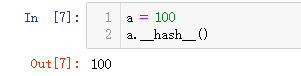
同理，可变对象(字典dict，列表list，集合set，字节数组bytearray)之所以不可哈希，原因是：更改了可变对象的值，例如修改list某个元素的值，list对象本身的地址并未改变，这意味着不同的key对应到了相同的value上，显然不满足哈希表的特性。

在Python中，更本质的解释是看对象是否有\_\_hash\_\_和\_\_cmp\_\_或\_\_eq\_\_，并且这些方法能够调用。

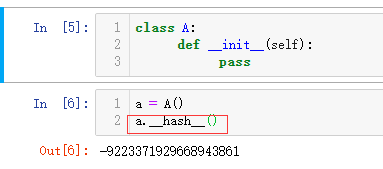
可以通过dir()函数，来查看对象包含的方法。

如果用dir()方法进行查看，会发现不可变对象和可变对象都包含\_\_hash\_\_和\_\_eq\_\_，但区别是可变对象的\_\_hash\_\_方法无法调用。

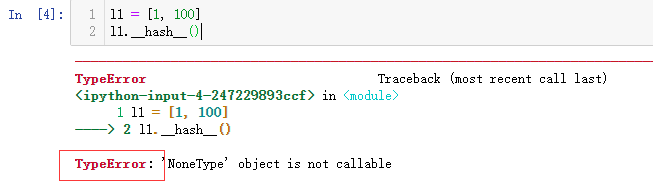
例：程序object\_type\_test\ hash\_unhash.ipynb



自定义类的实例：



可变对象list：



字典dict的key值必须是不可变对象，即可哈希对象。原因是：Python中字典为哈希表(这一点与C++不同，C++字典类似的类型map为红黑树)，如果把可变对象作为key创建在字典中，例如：

d = {[1, 2] : ‘A’}

当访问这个key时d[[1, 2]]，字典哈希表是根据地址(对象id)来查找的，两个可变对象[1, 2]的id是不同，尽管其值是相同的，这样也无法查询到创建字典时[1, 2]对应的值。