Python中有两种判断相同的操作：

1. ==操作符：测试两个被引用的对象是否具有相同的值
2. is操作符：检查对象的同一性，两个变量名是否指向同一个对象

例：程序equal\_test/equal\_test1.py

a = 3  
b = 3  
  
print("a == b:" + str(a == b))  
print("a is b:" + str(a is b))  
print()

tuple\_a = 1, 2 # 元组可以省略()  
tuple\_b = (1, 2)  
  
print("tuple\_a == tuple\_b:" + str(tuple\_a == tuple\_b)) # True  
print("tuple\_a is tuple\_b " + str(tuple\_a is tuple\_b)) # False  
print()

L = [1, 2, 3]  
M = [1, 2, 3]  
print("L == M:" + str(L == M))  
print("L is M:" + str(L is M))  
print()  
  
L1 = [1, 2, 3]  
M1 = L1  
print("L1 == M1:" + str(L1 == M1))  
print("L1 is M1:" + str(L1 is M1))

输出为：

a == b:True

a is b:True

L == M:True

L is M:False

L1 == M1:True

L1 is M1:True

从输出结果可以看出，如果给两个变量分别赋值了一个相同的不可变对象，is操作符会认为这两个变量指向了同一个不可变对象。这是由于Python中对小的整数和字符串的缓存复用机制。对于tuple不可变对象，不存在缓存复用。对于可变对象，则不会出现这种情况。