**变量的存储和赋值**

在Python中，变量和数据是分开存储的，数据保存在内存中的一个位置，变量中保存着数据在内存中的地址，称为地址引用，因此变量所需的存储空间大小一致。变量的每一次初始化，都开辟了一个新的空间，将新内容的地址赋值给变量。我们把不同的值赋给变量时候，变量地址不同，把相同的值赋给变量时，变量地址相同，可用id()函数获取变量在内存中的地址

a = 1

b = 1

print(id(a),id(b),id(a)==id(b))

>>> 140722619623216 140722619623216 True

id表示具体的存储地址的位数

lst = [1,2,3]

print(id(lst[0]),id(lst[1]),id(lst[2]))

>>>

140722619623216 140722619623248 140722619623280

简单数据类型(如字符串)变量，将一个变量给另一个变量，两个变量同时指向了一个地址，重新赋值其中一个，该变量会指向新的地址，另一个变量的id不会变

str1 = 'abc'

str2 = str1

print(id(str1),id(str2),id(str1)==id(str2))

print(id(str1[0]),id(str2[0]),id(str1[0])==id(str2[0]))

str1 = 'abcd'

print(id(str1),id(str2),id(str1)==id(str2))

>>>

2071721524464 2071721524464 True

2071721430448 2071721430448 True

2070887144880 2071721524464 False

复杂的数据类型(如列表)变量，将一个列表赋值给另一个列表，两个列表都指向了一个地址。对于一个列表重新初始化(赋值)的时候，就给这个变量重新赋予了一个新的地址，该列表的内存id就发生了改变，而另一个列表的id不会改变，其值也不会受影响。但当对列表中的元素进行一些增删改的操作的时候，是不会影响作为整体的列表的地址的，只会改变其内部元素的id，此时另一个列表相应元素的值和id也会改变

lst1 = [1,2,3,4]

lst2 = lst1

print(lst1,id(lst1),id(lst1[1]))

print(lst2,id(lst2),id(lst2[1]))

lst1[1] = 1

print(lst1,id(lst1),id(lst1[1]))

print(lst2,id(lst2),id(lst2[1]))

lst1 = [1,2]

print(lst1,id(lst1),id(lst1[1]))

print(lst2,id(lst2),id(lst2[1]))

>>>

[1, 2, 3, 4] 2070887119552 140722619623248

[1, 2, 3, 4] 2070887119552 140722619623248

[1, 1, 3, 4] 2070887119552 140722619623216

[1, 1, 3, 4] 2070887119552 140722619623216

[1, 2] 2070887078144 140722619623248

[1, 1, 3, 4] 2070887119552 140722619623216