深拷贝、浅拷贝

笔记本：	Python提高-1		
创建时间：	2018/5/6 16:17	**更新时间：**	2018/6/23 17:06
作者：	985601646@qq.com		

1，引用的作用：

```
a=1
b=a
id(a)
id(b)
```

b只引用了a的地址　并不会开辟独立的内存地址

Python数字缓存池　-5 ~ 256
Python解析器在运行时,就会直接把这些数字的在内存地址直接分配

```
a = 1
b = 1
id(a)
id(b)   # 地址一样     在范围以内的数字已经默认分配了固定的地址id(1)
```

2，深拷贝、浅拷贝

```
import copy  导入拷贝模块
```

浅拷贝：　浅拷贝是对于一个对象的顶层拷贝

```
拷贝后的变量名=copy.copy(原始变量名)
```

深拷贝：　深拷贝是对于一个对象所有层次的拷贝(递归)

```
拷贝后的变量名=copy.deepcopy(原始变量名)
```

（1）没有嵌套的单个**普通的数据**类型时　两种拷贝没有区别

- 都具备了独立性　（拷贝的变量不会随着原始变量值的改变而动态改变）

（2）有嵌套的**复杂的数据**类型

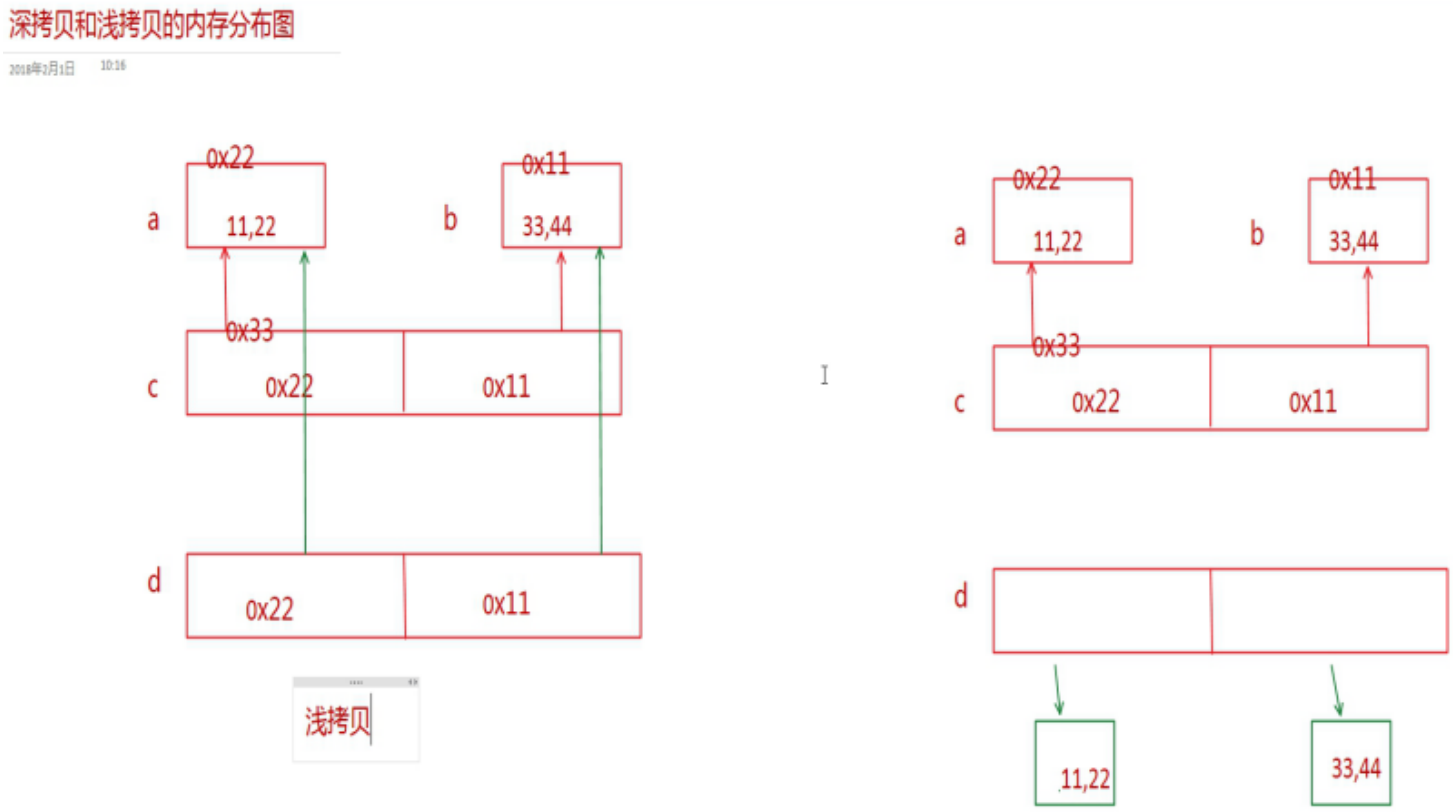
- 浅拷贝 只看最外层

    a. 最外层是可变类型时： 会直接开辟新的空间，拷贝数据
    b. 最外层是不可变类型时：不会开辟新的空间，直接引用

- 深拷贝 由外及内只要有任意一个类型为可变（递归），都会开辟新的空间，拷贝数据 ，保证数据的独立性

注意：两种不同的结果
拷贝数据：开辟新的内存地址（id不同），拷贝变量的值不会随着原始变量的值的改变而动态改变（复制），保证了独立性
引用：不会开辟行的内存地址(id相同），引用变量会随着原始变量的值的改变而动态改变（关联），无法保证数据的独立性

深拷贝和浅拷贝对应，深拷贝拷贝了对象的所有元素，包括多层嵌套的元素。因此，它的时间和空间开销要高。同样的对列表a，如果使用 b = copy.deepcopy(a)，再修改列表b将不会影响到列表a，即使嵌套的列表具有更深的层次，也不会产生任何影响，因为深拷贝拷贝出来的对象根本就是一个全新的对象，不再与原来的对象有任何的关联。



（4）分片表达式可以赋值一个序列

```
d = c[:]   与 d = copy.copy(c)  一样 属于浅拷贝
```

字典的copy方法可以拷贝一个字典 也是浅拷贝

```
d = {"name" : "zhangsan", "age" : 18}
co = d.copy
```