copy与mutableCopy

- 本质区别
  - 1：是否开启新的内存地址
  - 2：是否影响内存地址的引用计数
- 浅拷贝
  - a:浅拷贝就是对内存地址的复制，让目标对象指针和源对象指向同一片内存空间，
  - b:当内存销毁的时候，指向这片内存的几个指针需要重新定义才可以使用，要不然会成为野指针。
- 深拷贝
  - a:两个对象虽然存的值是相同的，但是内存地址不一样，两个对象也互不影响，互不干涉。
  - b:两个对象虽然存的值是相同的，但是内存地址不一样，两个对象也互不影响，互不干涉。
- 涉及
  - 涉及到容器与非容器、可变与不可变对象的copy与mutableCopy
- 准则
  - No1：可变对象的copy和mutableCopy方法都是深拷贝（区别完全深拷贝与单层深拷贝）。
  - No2：不可变对象的copy方法是浅拷贝，mutableCopy方法是深拷贝。
  - No3：copy方法返回的对象都是不可变对象。
  - 特别注意的是：对于集合类的可变对象来说，深拷贝并非严格意义上的深复制，只能算是单层深复制，即虽然新开辟了内存地址，但是存放在内存上的值（也就是数组里的元素仍然之乡员数组元素值，并没有另外复制一份），这就叫做单层深复制。
- 集合对象的完全深拷贝
  - (1) 归档解档大法
    - a: NSData *data = [NSKeyedArchiver archivedDataWithRootObject: marry1];
    - B:NSArray *marray2 = [NSKeyedUnarchiver unarchiveTopLevelObjectWithData:data error:nil];
  - (2) - (instancetype)initWithArray: (NSArray<ObjectType> *)array copyItems:(BOOL)flag;
    - NSArray *marray2 = [[NSArray alloc] initWithArray:marry1 copyItems:YES];
- strong与copy的异同点
  - 使用场景
    - a:在声明字符串属性时尽量使用copy，如果使用strong声明属性，那么当源字符串发生改变时，对应的属性值也会发生改变，
    - b:因为他们指向同一个地址，而使用copy就能杜绝这种情况，因为对于可变字符串进行copy是深拷贝，而strong只表示持有对象，引用计数加一。
  - A:当原字符串是NSString时，由于是不可变字符串，所以，不管使用strong还是copy修饰，都是指向原来的对象，copy操作只是做了一次浅拷贝
  - B:而当源字符串是NSMutableString时，strong只是将源字符串的引用计数加1，而copy则是对原字符串做了次深拷贝，从而生成了一个新的对象，并且copy的对象指向这个新对象。
  - C:另外需要注意的是，这个copy属性对象的类型始终是NSString，而不是NSMutableString，如果想让拷贝过来的对象是可变的，就要使用mutableCopy。
  - D: 所以，如果源字符串是NSMutableString的时候，使用strong只会增加引用计数。
  - E:但是copy会执行一次深拷贝，会造成不必要的内存浪费。而如果原字符串是NSString时，strong和copy效果一样，就不会有这个问题