- 深拷贝与浅拷贝
  - 浅拷贝
  - 直接赋值,默认浅拷贝传递对象的引用而已,原始列表改变，被赋值的b也会做相同的改变
  - copy.copy()浅拷贝，没有拷贝子对象，所以原始数据改变，子对象会改变
  - 深拷贝
  - 深拷贝，包含对象里面的自对象的拷贝，所以原始对象的改变不会造成深拷贝里任何子元素的改变
- 装饰器模式
  `针对带参数的decorator： import functools def log(text): def decorator(func): @functools.wraps(func) def wrapper(*args, **kw): print '%s %s():' % (text, func.__name__) return func(*args, **kw) return wrapper return decorator`