```
#ShowDeepCopy.py
""" Illustrates the difference between copy an deepcopy.
"""

from copy import copy,deepcopy

class MyColor:
    """
    Attributes:
        rgb:  a length-3 list of floats, each from the inteval [0,1]
        name: string that encodes the name of the color
    """
    def __init__(self,rgb,name):
        """ Creates a color.
        PreC: rgb is a length-3 list of floats, name is a string
        """
        self.rgb = rgb
        self.name = name

    def __str__(self):
        """ Pretty prints a MyColor object.

        To apply this function to a MyColor object P, write
            print P
        """
        return '%10s : [%4.2f, %4.2f, %4.2f]' %(self.name,self.rgb[0],self.rgb[1],self.rgb[2])

if __name__ == '__main__':
    # In this sequence, the idea was for C2 to reference the original
    # object referenced by C1. It does not.:
    C1 = MyColor([1,0,0],'red')
    print C1
    C2 = copy(C1)
    print C2
    C1.rgb[0]=0
    C1.name = 'black'
    print C1
    print C2

    print '\n'
     # Same sequence of instructions only with deepcopy
    C1 = MyColor([1,0,0],'red')
    print C1
    C2 = deepcopy(C1)
    print C2
    C1.rgb[0]=0
    C1.name = 'black'
    print C1
    print C2

    print '\n'
     # Same sequence of instructions but C1 is updated by creating
     # a new list object
    C1 = MyColor([1,0,0],'red')
    print C1
    C2 = deepcopy(C1)
    print C2
    C1.rgb = [0,0,0]
```

```
C1.name = 'black'
print C1
print C2
```