面向对象编程练习与简单数据结构

面向对象的设计思路

黑箱理论与封装性
黑箱大家应该都听说过，使用者不需要知道黑箱的内部内容，只要使用其外部接口，输入内容，由黑箱处理，再获得输出数据就行了。生活中有许多黑箱，尤其是在计算机中。计算机本身就是一个黑箱，我们不需要知道其硬件原理，不需要知道其操作系统和软件是怎么编写的，只要会使用PS就能得出想要修改的图片。编程道理也是这样，我们使用他人编写好的类库，并不需要知道这些类库是如何编写的，只需要知道如何调用就行。
封装是对于一个黑箱来说非常重要的东西。假如黑箱没有封装好，有电线露在外面，就有可能被不明真相的熊孩子剪断，影响使用。因此，将不想让人用的内容封装起来是非常重要的。

面向对象的设计思路，本质上就是在分解你的问题，然后针对每一类问题设计工具

面向对象更适合多人项目

（不仅仅是面向对象，程序设计的大致思路都是如此）

1. 明确你的需求

   举例：一个投资组合优化问题的小软件

   我手里选了几只股票，我想写一个软件，输入是我几只股票的股票代码，输出是我对每只股票的权重

2. 分解你的问题，分解成多个非常简单的小问题

   1. 一次只做一件事
   2. keep every thing simple

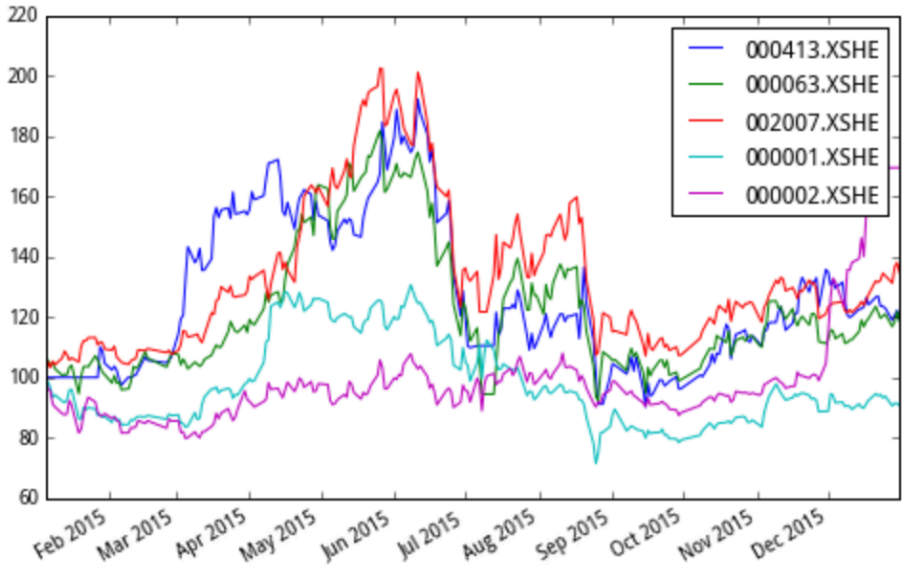
   投资组合优化问题

   1. 得到数据
      1. 什么样的数据：时间序列数据
      2. 从哪里得到数据：爱哪哪，比如说网上
         1. 查询数据 Yahoo API example
         2. 返回数据的文本
   2. 预处理数据
      1. 文本结构数据的处理

         从文本到结构化数据的解析

         返回时间序列数据

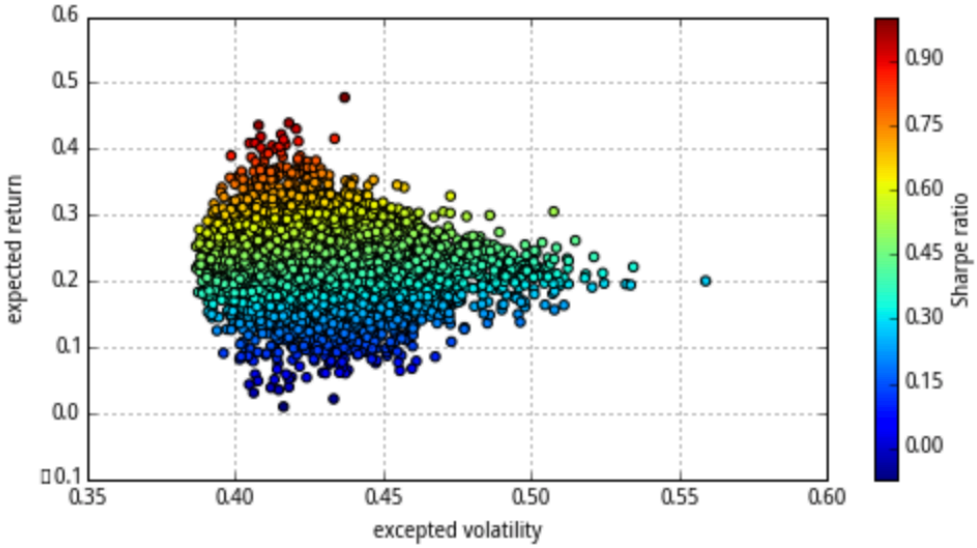
2. 时间序列的处理



    1. 均值
    2. 方差
    3. 协方差矩阵

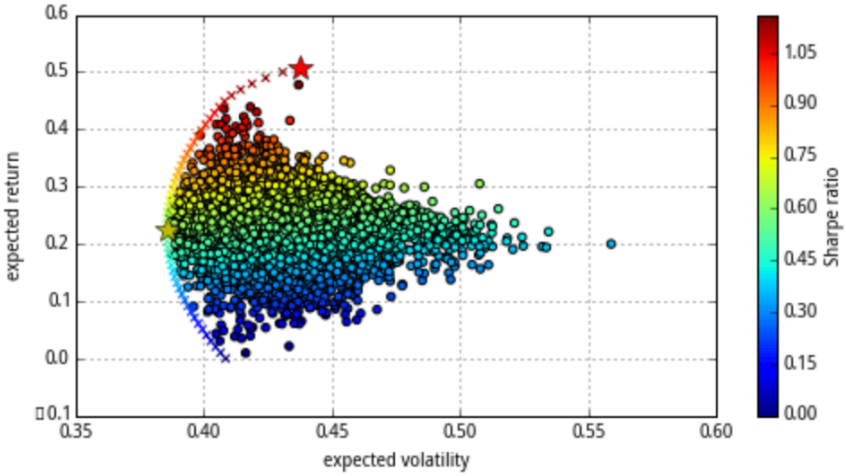
3. 分析数据

    1. 预期组合年华收益，组合方差和组合标准差
    2. 随机组合生成器



    3. 带约束组合优化问题的求解

        1. sharpe最大
        2. 方差最小
        3. 组合前沿



4. 用户交互，给出解

3. 将问题分类，抽象

   对每个问题进行分析，抽象，提取共同的内容（数据成员和操作方法）

   1. 同一类的数据结构
   2. 同一类的操作方法

4. 设计你的类，和类中的方法

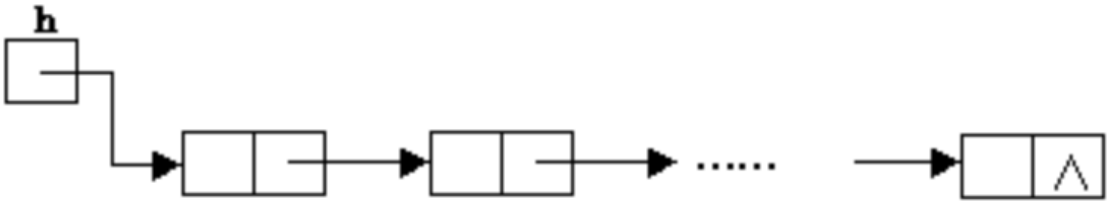
数据结构

1. 队列
2. 堆栈
3. 链表

练习：链表的面向对象实现

1. 链表

   链表是一种物理存储单元上非连续、非顺序的存储结构，数据元素的逻辑顺序是通过链表中的指针链接次序实现的。链表由一系列结点（链表中每一个元素称为结点）组成，结点可以在运行时动态生成。每个结点包括两个部分：一个是存储数据元素的数据域，另一个是存储下一个结点地址的指针域。



2. 实现：

   1. Node 类

      是每一个节点的类，节点需要存储本身的值和下一个节点的对象

      1. 属性

         1. 本身的值

        2. 指向的下一个节点，如果是队未就为空
    2. 方法
        1. 返回本身值的方法
        2. 返回此节点指向的下一个节点的方法
        3. 改变自身值得方法
        4. 改变自身指向的下一个节点的方法
2. LinkedList 类

    是单项链表的类

    1. 属性
        1. 链表的头指向的节点
        2. 链表的长度
    2. 方法
        1. 向队首增加节点
        2. 向队未增加节点
        3. 返回第某个节点的值
        4. 返回链表长度
        5. 打印全表
        6. 在第几个位置处插入一个节点
        7. 删除某个节点
3. 作业练习一

    用类似的方法实现一个双向链表