MODE-GL实现细节

1. 问题概述

N个asset，选择其中K个组成一组投资组合，K个asset中，每个asset获得一个投资比例后都能计算出它的预估收益以及预估风险，K个asset的总投资比例为1，该投资组合问题要求最小化风险，最大化收益。

在该问题中，存在6种约束：

1. 基数约束：每种组合最多选择K个asset。
2. 上下限约束：在被选择的K个asset中，每一个asset的投资比例有一个下限与上限。
3. 预分配约束：有|Z|个asset已经被要求一定要选择。
4. 批约束：投资组合中的任何一个asset的投资比例都必须是vi的精确倍数，该vi是市场上能够交易的最小批数（或者用最小交易单元来理解）。
5. 类约束：N个asset，这些asset属于不同的类型（有些相同，有些不同），假设这N个asset一共有M种类型，则要求这M种类型中，每一种都满足至少一个属于该类的asset被选择在K中。
6. 类限约束：每一种类的总投资比例有上下限。
7. 输入输出描述

输入为94个asset的1000天实际收益记录，该组文件在当前目录下的MODED-GL-DataSets文件夹中。

输出为用A集合，该集合在下面的内容中会有描述。

1. MODE-GL算法描述

该算法在DEMO算法的基础上进行改进。

1. 相比于DEMO，MODE-GL添加了两个外部档案，A archive和D archive，用来存储在迭代过程中找到的易于扩散的非支配解。
2. 提出了一种学习机制用于在迭代过程中提取有效解的重要特征。
3. 提出两种 DE算法的拓展突变方案。
4. 利用学习机制，问题特定启发以及有效的突变方案提出的高效解生成方案，指引搜索方向到最可能有解的区域。

DEMO算法：

1. 初始化种群并评估个体
2. 当不满足停止条件时:
   1. 对于种群的每一个个体Pi：
      1. 从Pi中生成新个体P’
      2. 对P’进行评估
      3. 如果P’优于Pi则用P’取代Pi
      4. 否则如果Pi优于P’则删除P’
      5. 否则将P’添加至种群
   2. 种群大小超过限定值则对它消减至限定值内

MODE-GL算法

输入：archive A,D的大小A\_max 与D\_max以及Population的大小NP

1. 随机生成种群P
2. 当不满足停止条件时：
   1. 利用P中的非支配解维护A集
   2. 使A集大小维护在A\_max以内
   3. 利用P中前D\_max个拥挤度最小的解维护D
   4. 利用A集计算每个asset的浓度值
   5. 对于P中的每个个体pi（i = 1，... ，N）
      1. 利用生成机制生成一个新的个体p’
      2. 如果p’不满足约束则修复
      3. 对p’进行评估
      4. 如果p’支配pi则用p’替换pi
      5. 否则如果pi支配p’则丢弃p’
      6. 否则将p’加入P

f） 如果P的大小超过NP，将P中的解通过非支配性以及拥挤距离排序，留下前NP个个体

输出：A集

Individual 生成算法

输入：每个asset的浓度值以及p` P

1. 选择|Z|个预分配asset后随机通过S1-S4选择方案选择K-|Z|个asset，并且使其满足类约束
2. 随机从{P \ p`}中选择三个不同的解：p1,p2,p3
3. 从K个asset中随机选择一个并让其下标复制给i，之后令j = = I
4. 对于每一个选择的asset：
   1. 如果 r(0,1) < CR 或者 j == 则通过W1或者W2方案给该asset分配比例
   2. 否则 将p`的wi赋值给当前asset
   3. 从K个asset中随机选择一个并让其下标复制给j

输出：新的解 p`

S1-S4方案：

S1：以asset浓度ci为基础使用轮盘赌策略选择asset

S2：选择ci值最高的asset

S3：选择平均收益最高的asset

S4：选择收益标准差最小的asset

W1-W2方案：

W1: wi` = best\_i + r[0,1] \* (w1i – w2i)

W2: wi` = wi + F \* (best\_i – wi) + F \* (w1i – w2i)

其中w1,w2来自D集，随机抽取；best\_i随机挑选自A集的前10%；F是缩放因子。

Individual 修复机制  
1.被选择的全部asset的投资比例通过以下公式调整：

wi` = +

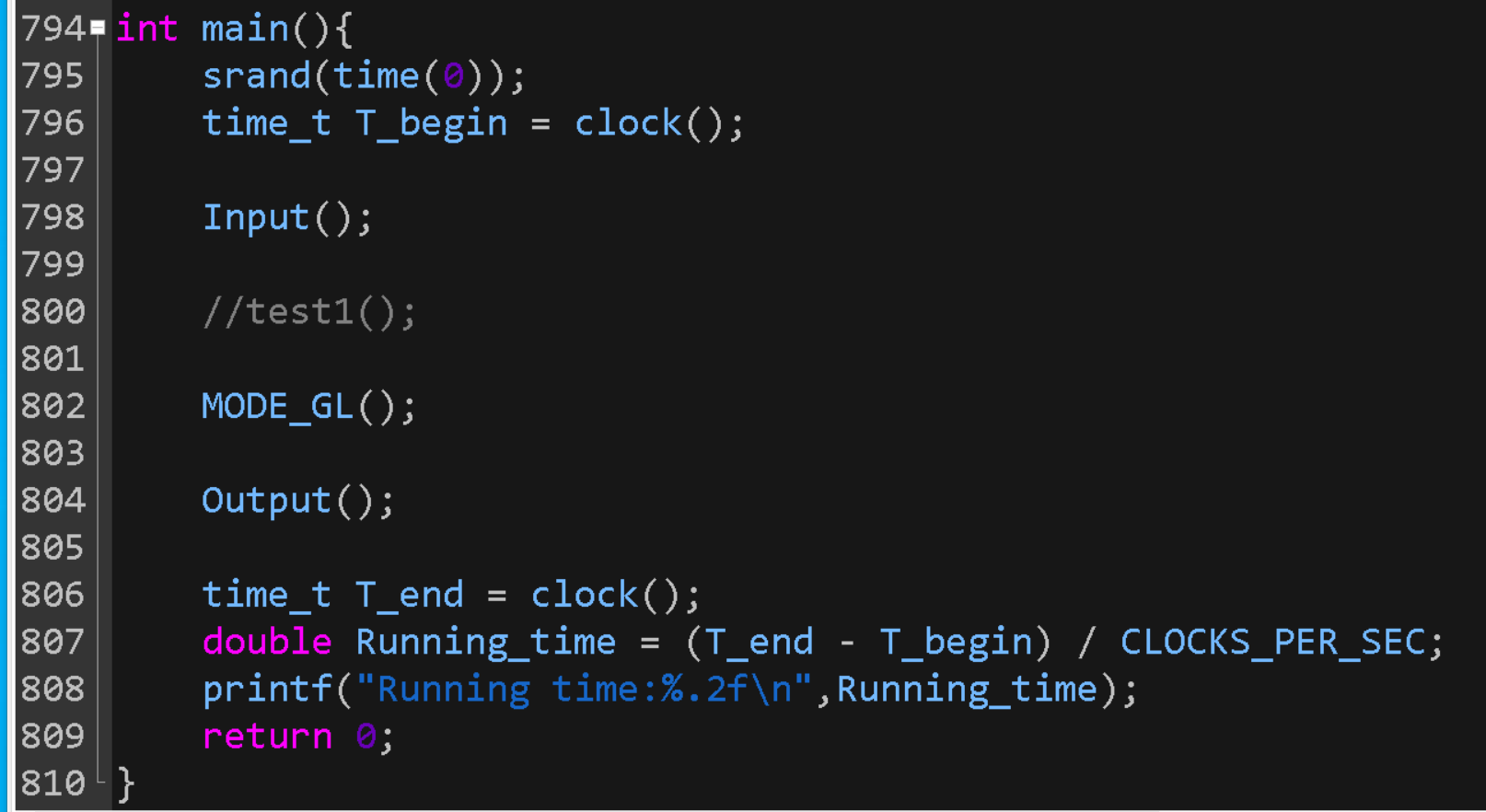
其中

2.令 wi` = wi` - (wi` mod vi)，然后将 加入到最大的(wi` mod vi)下标asset的比例中。

1. 如果比例违反类限约束，则wi需要做以下调整。
   1. 如果，则从满足条件的类中抽取vi插入到类m中，直到 ，其中。
   2. 如果，则用与a)相似的方法调整。

四、代码细节

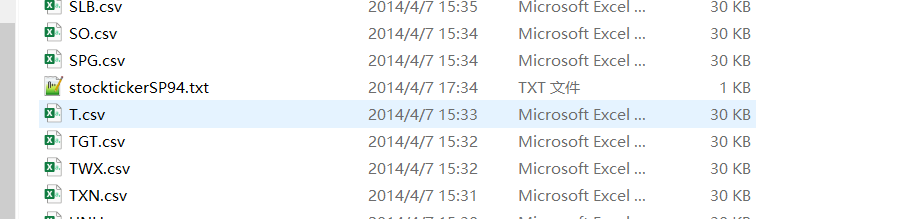
main函数：



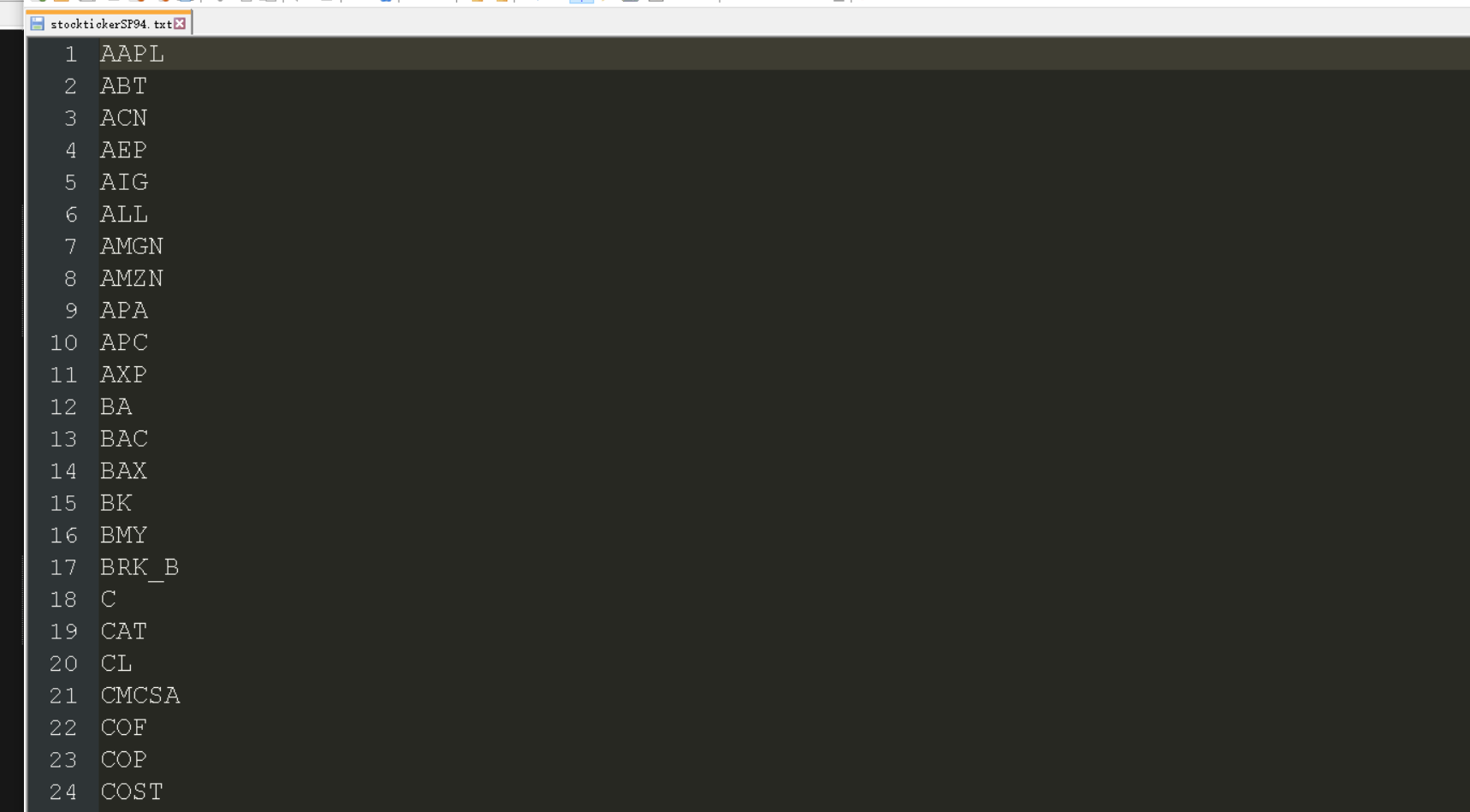
main函数主要由3个函数组成，分别是输入函数，输出函数以及处理数据的函数

srand(time(0))语句初始化随机数种子，其它内容用于计算程序运行时间

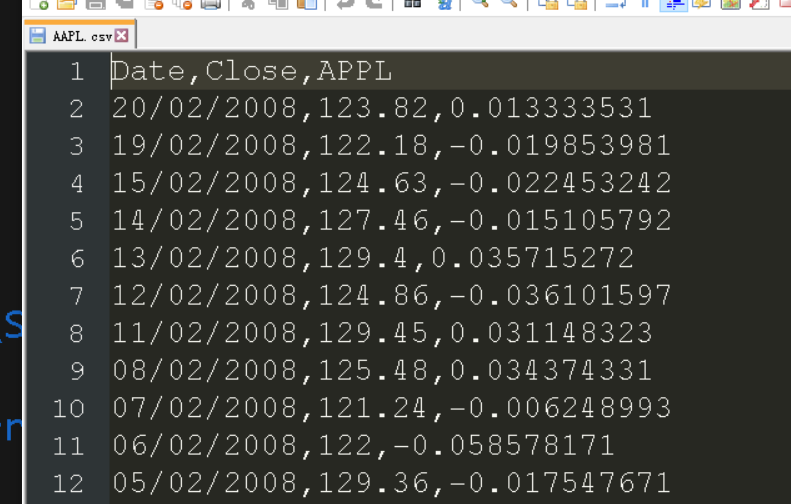
Input()函数用于输入当前目录下MODED-GL-DataSets/S&P/中S&P100文件，这些文件由数据文件以及数据文件名称文件。如下图：



.csv后缀的文件是数据文件，其中stocktickerSP94.txt文件是这些数据文件的名称。



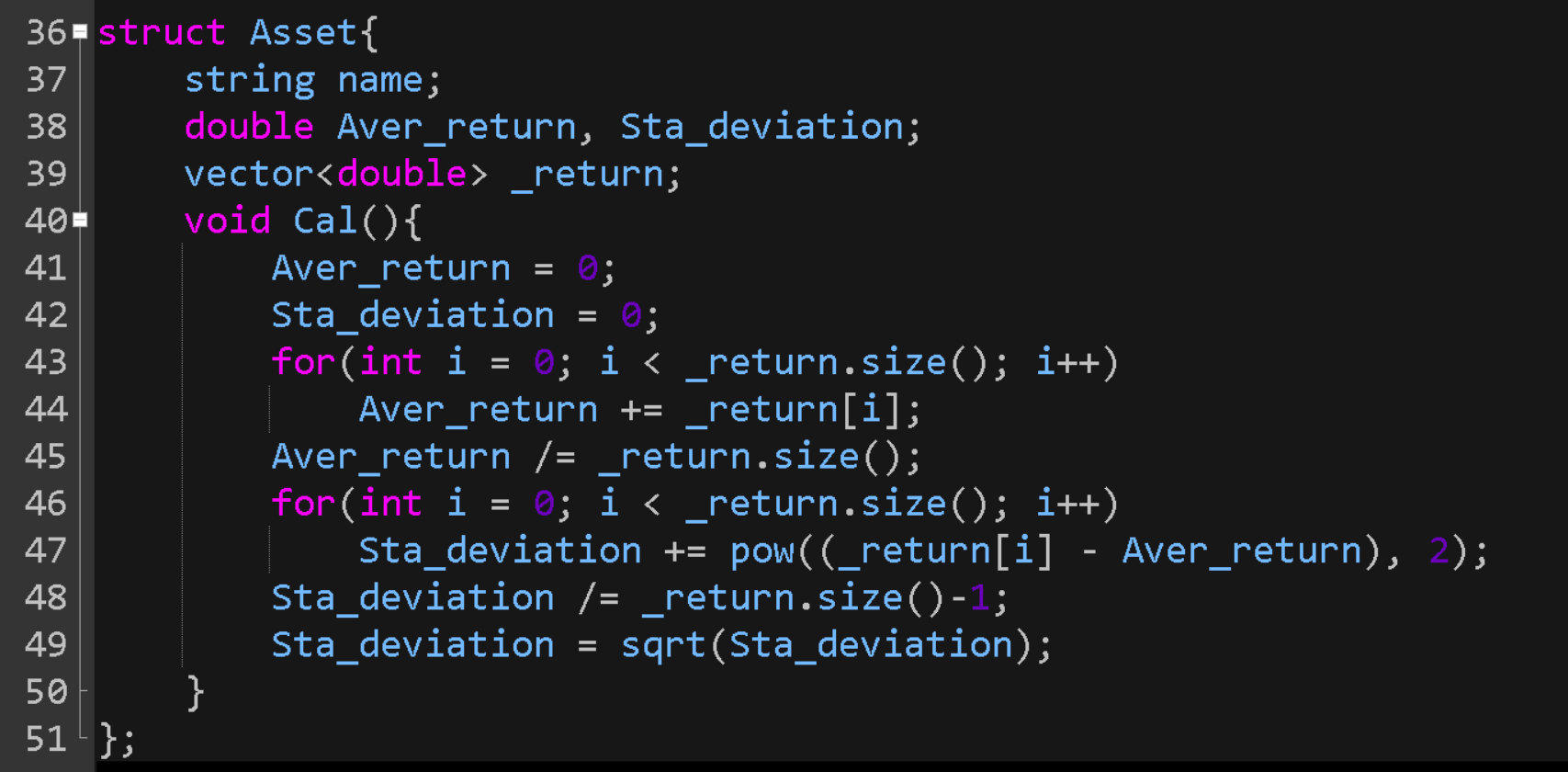
.csv数据文件的内容如下：



除了第一行，其他每行存在两个“,”将数据分为三列，第一列是时间，第二列是当天asset的收盘价，第三列是增长率（即（今天收盘价-昨天收盘价）/ 昨天收盘价）。我们只需要第二列的

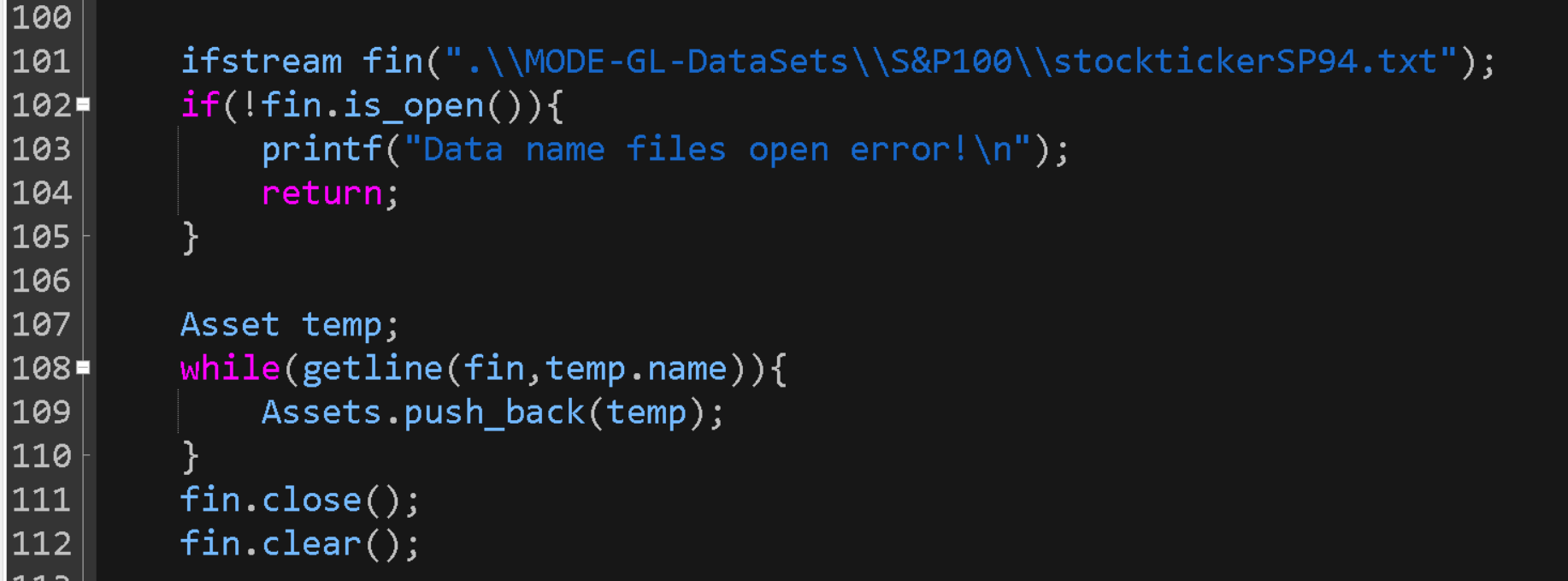
由这些数据，对输入函数的编写思路是：

1. 建立asset的结构体，该结构体里的属性有asset的名称，每日相对于前一天的收益数组，平均收益以及收益标准差。



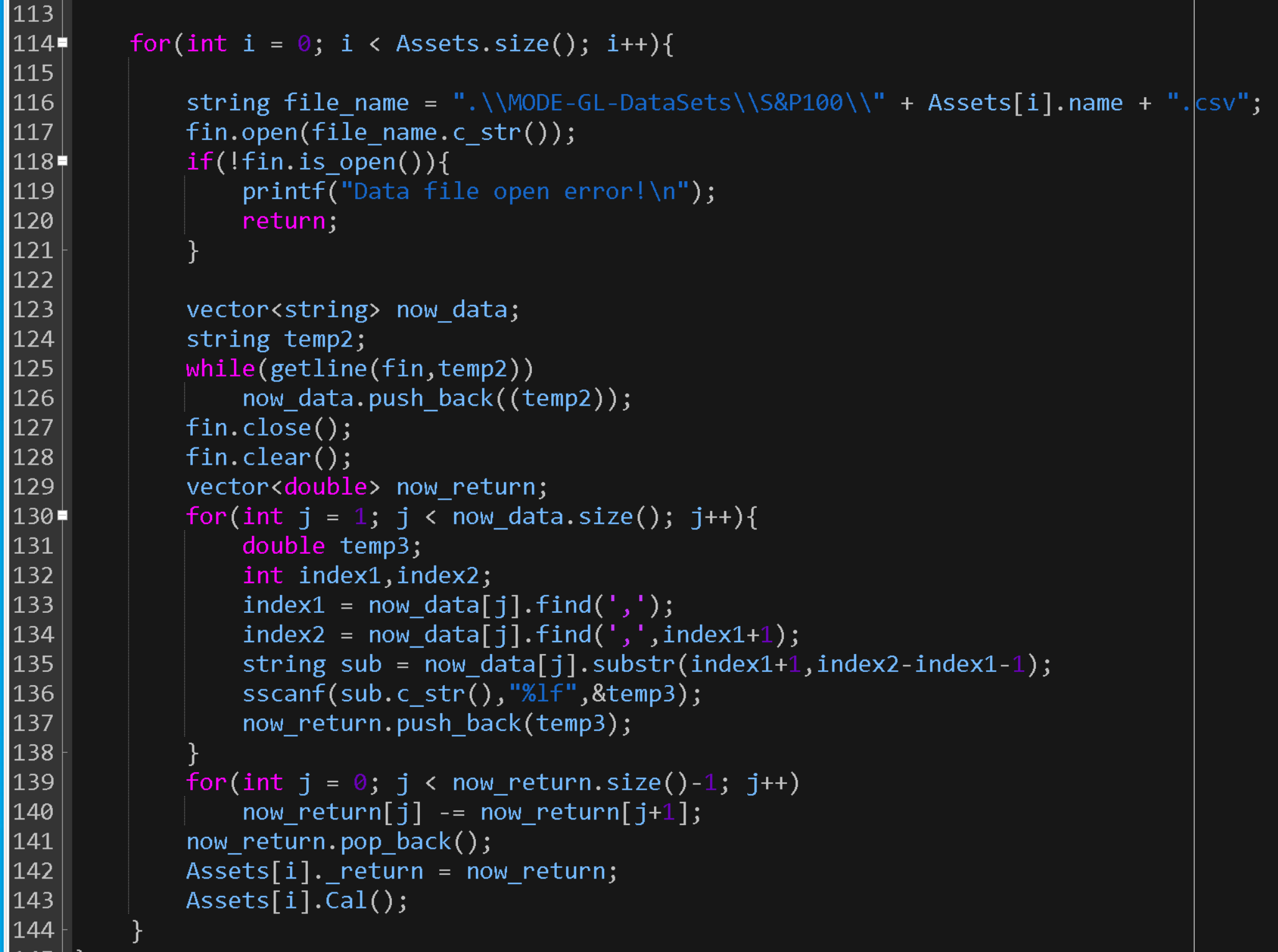
Cal()函数是该结构体的内置函数，用来计算一个asset的平均收益与标准差。

1. 通过读入stocktickerSP94.txt获得每个asset的名称。



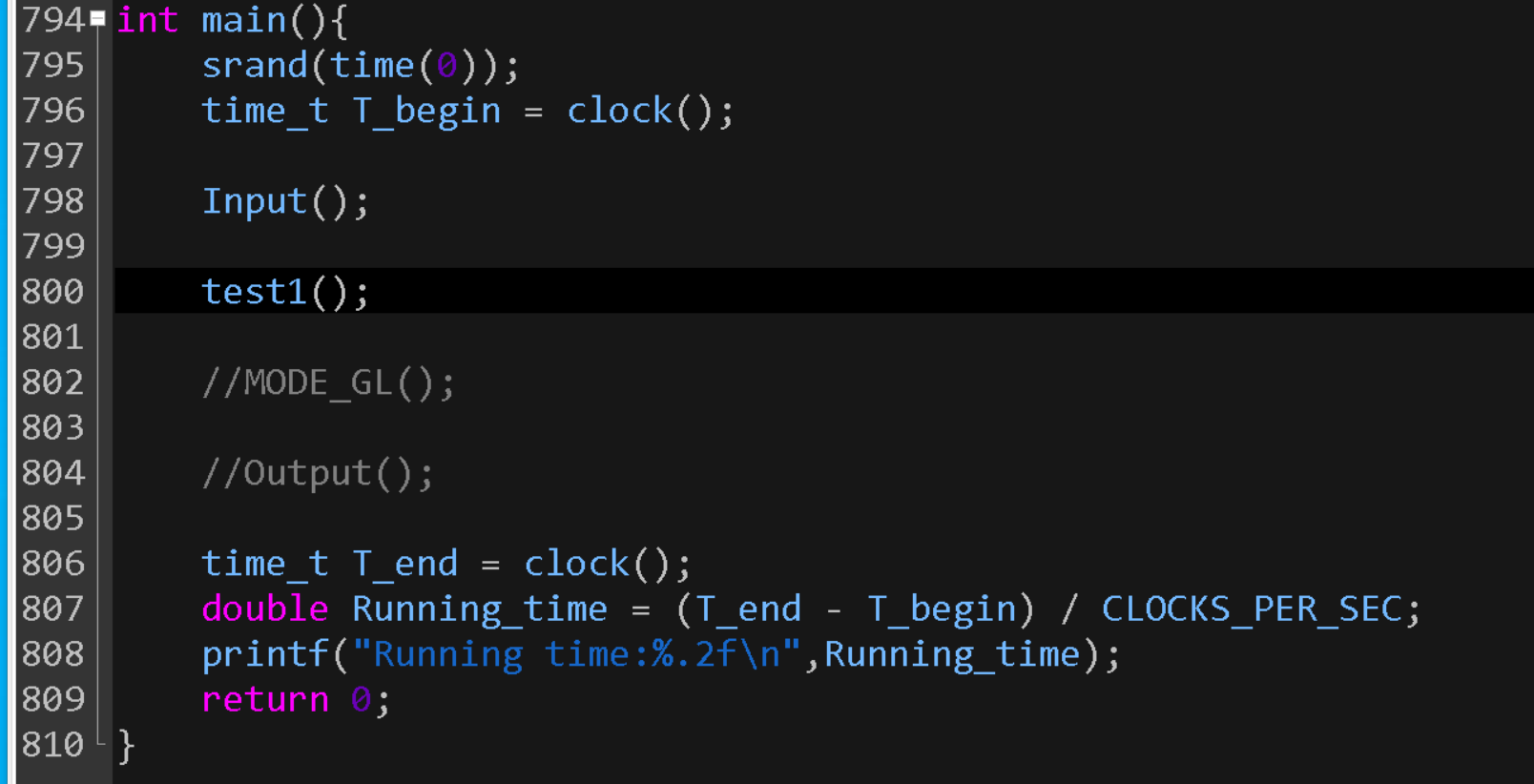
101-105打开数据名称文件，107-112读取该文件每行内容并且存入暂存入Asset类型变量中。109行出现的Assets是一个Asset类型的全局动态数组。

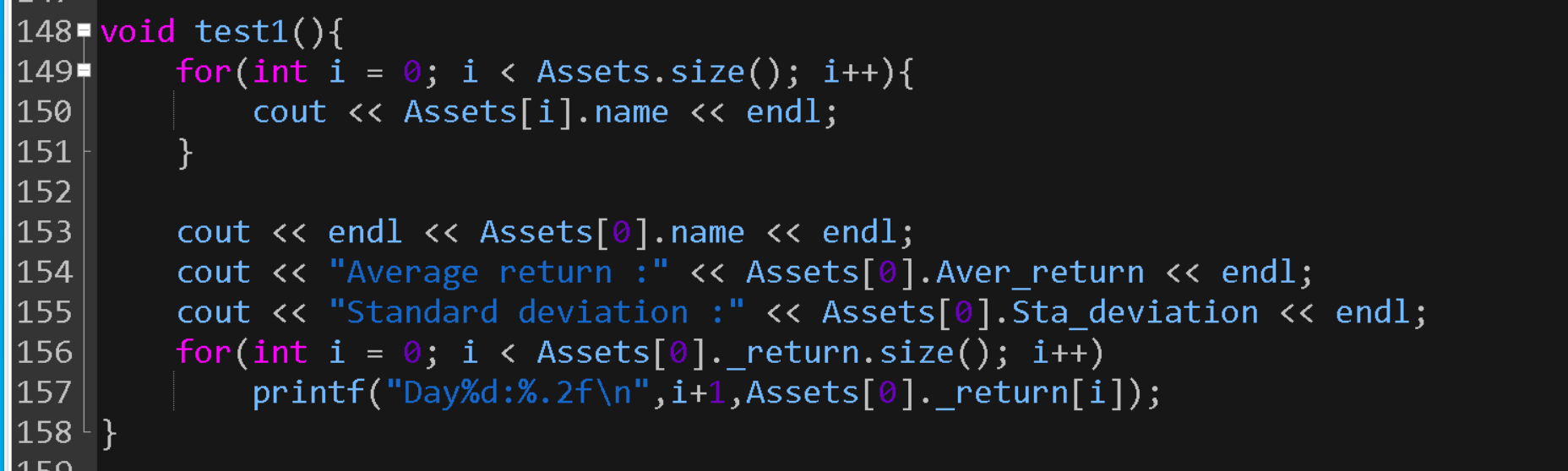
1. 通过上一步读入的数据文件名称打开每个asset数据文件，对于每个asset文件，使用一个string类型的动态数组存储该数据文件全部数据，然后对于第二行开始的数据，提取第二列的收盘价，存入当前asset结构体中的\_return数组，然后计算每一天的收益（今天收盘价-昨日收盘价）。因为日期是从后往前读取（即第j行的时间更晚，j+1行的时间更早），所以用\_return[j] - \_return[j+1]计算每一天的收益，计算完成之后，删除\_reutrn数组中的最后一个元素，并调用Asset结构体中的Cal()函数计算平均收益以及其标准差。



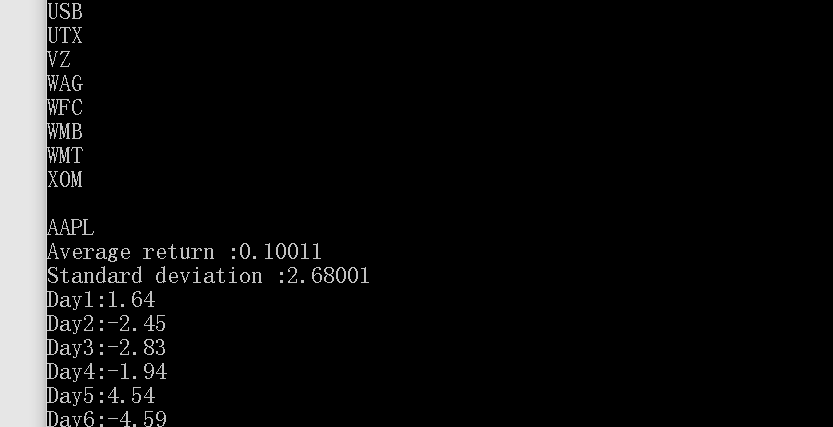
116-121打开文件，123-128读取内容，129-138提取从第二行开始的第二列数据，存入now\_return数组中，139-140计算每天收益以及删除最后一个元素，142行将当前提取的数据保存在Assets数组中，143调用Cal()函数。

现在回到主函数，测试是否能够正确读取数据，这里做了一个test1()函数用作测试数据是否读入成功。

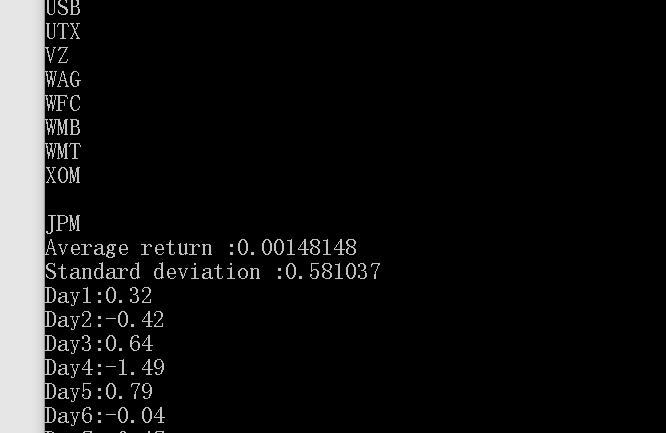




在主函数中将其他两个函数注释后运行test1函数，测试内容是每个asset的名称以及第1个asset的全部数据。

 (只截取部分)

修改成第51个asset输出也是正确的。

（只截取部分）