1.0阶段：

-从数据库中获取25支股票的数据，从中随机抽出一支，并对其数据进行隔行抽取（如隔5行），模拟现实股票数据显示单位。

-用python进行股票对比搜寻，确定样本属于哪支股票。

我的解决方案：

①将25支股票的数据爬取后存入excel文档（stock.xlsx)，同时选取某支股票（603813.SH）利用OFFSET公式各行抽取，另存为新的文档extract.xlsx；

②选取一定的区间，如隔5行，隔10行，……对25支股票的数据进行重组生成result\_file.xlsx，行列转置后另存为新的文档path\_to\_transposed\_excel\_file.xlsx；

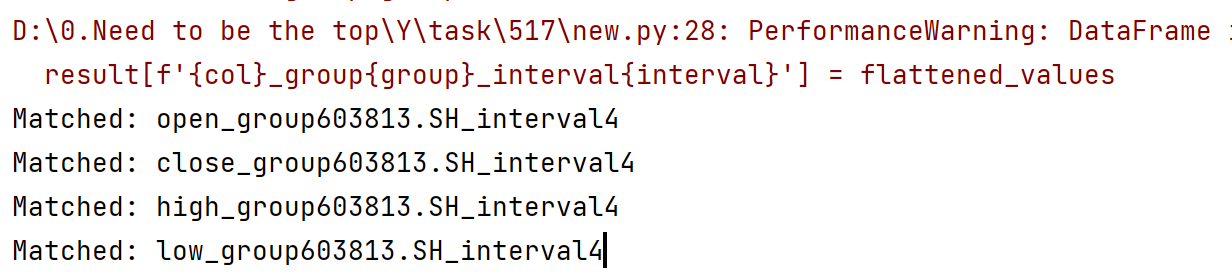
③对于extract每一条以列表形式读取的记录，遍历path\_to\_transposed\_excel\_file.xlsx的每一行，判断是否有匹配。

不足之处与可改进的地方：

①在对比时是将open、close、high、low分开来对比的。原本有试图将其作为一个个以天为单位的元组（open，close，high，low）来进行对比，但在生成上述的result\_file.xlsx后表头变得难以处理，遂放弃……

②逐条遍历、精确对比效率比较低，其实可以设置一个临界值或判断标准，使程序认为只要在一定程度上完成了相对精确的匹配就可以判断“确实是这支股票”。目前没有想到对应的解决办法。

运行结果：



代码如下：

import pandas as pd

# 读取股票数据库并进行预处理

data = pd.read\_excel('D:/0.Need to be the top/Y/task/517/stock.xlsx')

# 指定要操作的列名和分组列名

value\_columns = ['open', 'close', 'high', 'low']

group\_column = 'thscode'

# 获取分组列的唯一值

group\_values = data[group\_column].unique()

# 创建一个空的 DataFrame 用于存储结果

result = pd.DataFrame()

# 对每个列、每个分组和每个间隔进行操作，并将结果添加到结果 DataFrame

for col in value\_columns:

for group in group\_values:

group\_data = data[data[group\_column] == group]

for interval in range(1,21):

values = []

for i in range(0, len(group\_data), interval):

value = group\_data[col].iloc[i] # 获取指定间隔的单元格值

values.append(value)

flattened\_values = values + [float('nan')] \* (len(group\_data) - len(values))

# 将结果填充到结果 DataFrame，并确保填充缺失值为 NaN

result[f'{col}\_group{group}\_interval{interval}'] = flattened\_values

# 将结果保存为新的 Excel 文件

result.to\_excel('D:/0.Need to be the top/Y/task/517/result\_file.xlsx', index=False)

# 整理数据库的格式以便操作

df = pd.read\_excel(r'D:/0.Need to be the top/Y/task/517/path\_to\_transposed\_excel\_file.xlsx')

database = []

for index, row in df.iterrows():

data\_list = list(row)

database.append(data\_list)

# 导入需要查找的股票的数据

df = pd.read\_excel(r'D:/0.Need to be the top/Y/task/517/extract.xlsx')

tolerance = 0.01#浮点数比对，一定的容错率

extract = []

for column in df:

data\_list = list(df[column])

extract.append(data\_list)

# 对比并输出匹配的记录对应的股票名

for i, alist in enumerate(extract):

match = False

for record in database:

if all(abs(extract\_value - db\_value) <= tolerance for extract\_value, db\_value in zip(alist, record[1:])):

print(f"Matched: {record[0]}")

match = True

break

if not match:

print("No match found")