**In the Python programs, we assume students will import Matplotlib, Pandas, Numpy and Scipy as the following:**

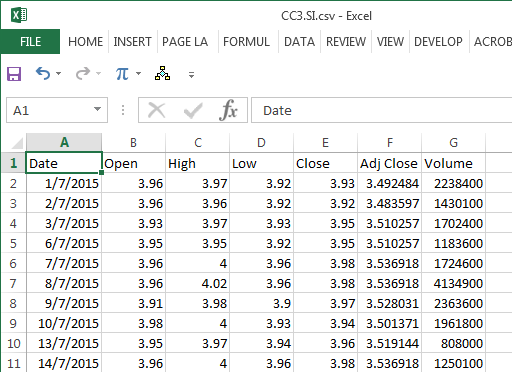
**import matplotlib.pyplot as plt**

**import pandas as pd**

**import numpy as np**

**Write a Python program according to the following instructions.**

1. **Use one command with** **pandas.read\_csv** to load data from the CSV file **CC3.SI.csv** (see the figure) to a DataFrame, using the **first column** **as the row labels**, the **first row as column names** and **parse the row index as dates**. Name this DataFrame as **data**.

****

**data = pd.read\_csv('CC3.SI.csv', index\_col=0, parse\_dates=True)**

1. **Use one command** **with** **pandas.DataFrame.drop** to drop the row in **data** whose “Volume” is zero.

**data.drop(data.index[data['Volume']==0],inplace=True)**

1. **Use one command** to add a new column with the name “**15d**” to **data** which is the rolling mean of 15 (rounded to 3 decimal places) of the data in the column “**Adj Close**” computed by the library function **pandas.Series.rolling** (not **pandas.DataFrame.rolling**).

**data['15d']=np.round(data['Adj Close'].rolling(15).mean(), 3)**

1. **Use one command** to add a new column with the name “**50d**” to **data** which is the rolling mean of 50 (rounded to 3 decimal places) of the data in the column “**Adj Close**” computed by the library function **pandas.Series.rolling** (not **pandas.DataFrame.rolling**).

**data['50d']=np.round(data['Adj Close'].rolling(50).mean(), 3)**

1. **Use one command** **with the operator** **-** to compute the difference (stored as a Series with the same labels as those of **data**) between the data in the column “**15d**” and the data in the column “**50d**” (i.e. “**15d**”“**50d**”) within the same row. Name this the difference as **x**.

**x=data['15d']-data['50d']**

1. **Use one command** to update the Series **x** by changing all the positive numbers to **1**.

**x[x>0]=1**

1. **Use one command** to update the Series **x** by changing all the negative numbers and zero to **0**.

**x[x<=0]=0**

1. **Use one command** to compute and store the first discrete difference of the elements in the Series **x** as a Series with name **y**.

**y=x.diff()**

1. **Use one command** to find and store the labels of those elements in the Series **y** whose values are negative as **idxSell**.

**idxSell=y.index[y<0]**

1. **Use one command** to find and store the labels of those elements in the Series **y** whose values are positive as **idxBuy**.

**idxBuy=y.index[y>0]**

1. **Use one command** to add a new column with the name “**crossSell**” to **data** and make every value equal to the special number **NaN**.

**data['crossSell']=np.nan**

1. **Use one command** to update the column “**crossSell**” in **data** in the rows where row labels are equal to **idxSell** using the values in the column “**Adj Close**” within the same row.

**data.loc[idxSell,'crossSell']=data['Adj Close'][idxSell]**

1. **Use one command** to add a new column with the name “**crossBuy**” to **data** and make every value equal to the special number **NaN**.

**data['crossBuy']=np.nan**

1. **Use one command** to update the column “**crossBuy**” in **data** in the rows where row labels are equal to **idxBuy** using the values in the column “**Adj Close**” within the same row.

**data.loc[idxBuy,'crossBuy']=data['Adj Close'][idxBuy]**

1. **Use one command with pandas.DataFrame.plot** to plot the 5 columns of **data** (“**Adj Close**”, “**15d**”, “**50d**”, “**crossSell**” and “**crossBuy**”) using 5 different line styles (black solid line, blue solid line, cyan solid line, red circle and yellow circle) and linewidth **1**.

**data[['Adj Close', '15d', '50d','crossSell','crossBuy']].plot(**

**ax=ax, style=['k-','b-','c-','ro','yo'], linewidth=1)**

1. **Use one command** to display the figure.

**fig, ax=plt.subplots()**