**Task: Asynchronous Requests 2 (Part 3)**

Overview: Redo Tasks 19 & 20 but use the list created from Task 21 instead of the “listofthousandtickers.txt”

1. Import the following modules at the start of your algorithm

A screen shot of a computer code

Description automatically generated

2. Create “today”, “starttime”, and “endtime” variables:

A black text on a white background

Description automatically generated

A screenshot of a computer code

Description automatically generated

3. Create a list of all the tickers from task 21

A close-up of a computer code

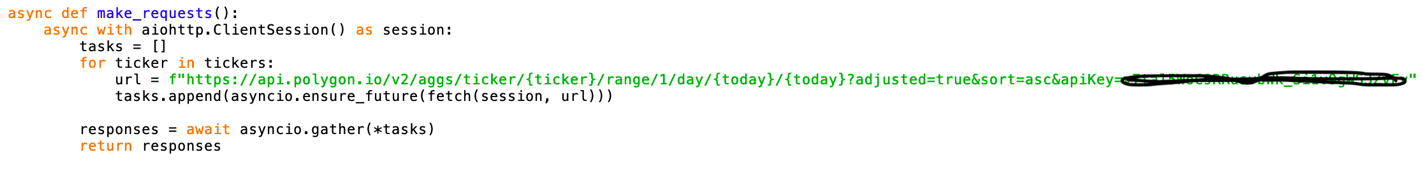
Description automatically generated

4. Create the asynchronous function “fetch”

A close-up of a computer screen

Description automatically generated

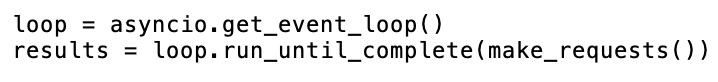
5. Create the asynchronous function “make\_requests” that makes requests to the Polygon.io API



A screenshot of a computer

Description automatically generated

6. Create the event loop and run it using the following code:



6. For each ticker in “tickers” create an object containing the stock’s ticker, its opening price, and its last trading price

Hint: The last trading price can be found parsing through the returned JSON object by using “c”. Before 4pm, the closing price is equivalent to the last trading price

7. Rank the stock objects from highest to lowest difference

8. Create an empty list called “stocks” and append the 15 stocks with the highest difference to the list. You can also print out the 15 stocks with the highest difference.

9. Obtain the minute by minute data for each stock in “stocks” by making a request to the Polygon.io (<https://polygon.io/docs/stocks/get_v2_aggs_ticker__stocksticker__range__multiplier___timespan___from___to>) and you will get a JSON object as a return value.

A screenshot of a computer

Description automatically generated

Hint: Use an f-string with the two variables from step 3 for the “from” and “to” parameters

10. For each stock create list of it’s minute by minute open price, close price, high price, low price, and volume.

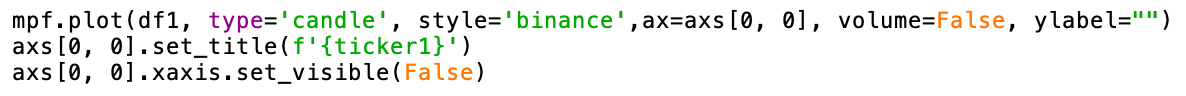
11. For each stock create an empty dataframe using pandas

12. For each stock manipulate the first column in the dataframe so that it displays the time, starting at 9:30am and increasing by one minute each row

13. Create subplots



14. Plot each subplot with mplfinance using the specified parameters



15. Show the subplot



16. Save your algorithm as a “.py” file in your documents folder:

A screenshot of a computer

Description automatically generated

17. Run your algorithm from the terminal (change directory to documents before running code)

A screenshot of a computer

Description automatically generated