In [1]:

```python
# install pandas and pyalgotrading
!pip install pandas
!pip install pyalgotrading
```

```
Requirement already satisfied: pandas in c:\users\jyoti\appdata\local\programs\python\python310\lib\site-packages (2.0.2)
Requirement already satisfied: tzdata>=2022.1 in c:\users\jyoti\appdata\local\programs\python\python310\lib\site-packages (from pandas) (2023.3)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\jyoti\appdata\local\programs\python\python310\lib\site-packages (from pandas) (2.8.2)
Requirement already satisfied: numpy>=1.21.0 in c:\users\jyoti\appdata\local\programs\python\python310\lib\site-packages (from pandas) (1.23.5)
Requirement already satisfied: pytz>=2020.1 in c:\users\jyoti\appdata\local\programs\python\python310\lib\site-packages (from pandas) (2023.3)
Requirement already satisfied: six>=1.5 in c:\users\jyoti\appdata\local\programs\python\python310\lib\site-packages (from python-dateutil>=2.8.2->pandas) (1.1
6.0)
```

```
[notice] A new release of pip available: 22.2.1 -> 23.1.2
[notice] To update, run: python.exe -m pip install --upgrade pip
```

```
Requirement already satisfied: pyalgotrading in c:\users\jyoti\appdata\local\programs\python\python310\lib\site-packages (2023.6.1)
Requirement already satisfied: pandas>=0.25.3 in c:\users\jyoti\appdata\local\programs\python\python310\lib\site-packages (from pyalgotrading) (2.0.2)
Requirement already satisfied: requests>=2.24.0 in c:\users\jyoti\appdata\local\programs\python\python310\lib\site-packages (from pyalgotrading) (2.31.0)
Requirement already satisfied: pytz>=2020.1 in c:\users\jyoti\appdata\local\programs\python\python310\lib\site-packages (from pandas>=0.25.3->pyalgotrading)
(2023.3)
Requirement already satisfied: tzdata>=2022.1 in c:\users\jyoti\appdata\local\programs\python\python310\lib\site-packages (from pandas>=0.25.3->pyalgotrading)
(2023.3)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\jyoti\appdata\local\programs\python\python310\lib\site-packages (from pandas>=0.25.3->pyalgo
trading) (2.8.2)
Requirement already satisfied: numpy>=1.21.0 in c:\users\jyoti\appdata\local\programs\python\python310\lib\site-packages (from pandas>=0.25.3->pyalgotrading)
(1.23.5)
Requirement already satisfied: idna<4,>=2.5 in c:\users\jyoti\appdata\local\programs\python\python310\lib\site-packages (from requests>=2.24.0->pyalgotrading)
(3.4)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\jyoti\appdata\local\programs\python\python310\lib\site-packages (from requests>=2.24.0->py
algotrading) (3.1.0)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\jyoti\appdata\local\programs\python\python310\lib\site-packages (from requests>=2.24.0->pyalgotr
ading) (2.0.3)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\jyoti\appdata\local\programs\python\python310\lib\site-packages (from requests>=2.24.0->pyalgotr
ading) (2023.5.7)
Requirement already satisfied: six>=1.5 in c:\users\jyoti\appdata\local\programs\python\python310\lib\site-packages (from python-dateutil>=2.8.2->pandas>=0.2
5.3->pyalgotrading) (1.16.0)
```

```
[notice] A new release of pip available: 22.2.1 -> 23.1.2
[notice] To update, run: python.exe -m pip install --upgrade pip
```

In [39]:

```python
#install matplotlib
pip install matplotlib
```

```
Collecting matplotlib
  Downloading matplotlib-3.7.1-cp310-cp310-win_amd64.whl (7.6 MB)
     ---------------------------------------- 7.6/7.6 MB 4.4 MB/s eta 0:00:00
Collecting cycler>=0.10
  Downloading cycler-0.11.0-py3-none-any.whl (6.4 kB)
Collecting pillow>=6.2.0
  Downloading Pillow-9.5.0-cp310-cp310-win_amd64.whl (2.5 MB)
     ---------------------------------------- 2.5/2.5 MB 1.4 MB/s eta 0:00:00
Collecting kiwisolver>=1.0.1
  Downloading kiwisolver-1.4.4-cp310-cp310-win_amd64.whl (55 kB)
     -------------------------------------- 55.3/55.3 kB 221.8 kB/s eta 0:00:00
Collecting contourpy>=1.0.1
  Downloading contourpy-1.1.0-cp310-cp310-win_amd64.whl (470 kB)
     -------------------------------------- 470.4/470.4 kB 2.5 MB/s eta 0:00:00
Requirement already satisfied: packaging>=20.0 in c:\users\jyoti\appdata\local\programs\python\python310\lib\site-packages (from matplotlib) (21.3)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\jyoti\appdata\local\programs\python\python310\lib\site-packages (from matplotlib) (3.0.9)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\jyoti\appdata\local\programs\python\python310\lib\site-packages (from matplotlib) (2.8.2)
Collecting fonttools>=4.22.0
  Downloading fonttools-4.40.0-cp310-cp310-win_amd64.whl (1.9 MB)
     ---------------------------------------- 1.9/1.9 MB 3.3 MB/s eta 0:00:00
Requirement already satisfied: numpy>=1.20 in c:\users\jyoti\appdata\local\programs\python\python310\lib\site-packages (from matplotlib) (1.23.5)
Requirement already satisfied: six>=1.5 in c:\users\jyoti\appdata\local\programs\python\python310\lib\site-packages (from python-dateutil>=2.7->matplotlib)
(1.16.0)
Installing collected packages: pillow, kiwisolver, fonttools, cycler, contourpy, matplotlib
Successfully installed contourpy-1.1.0 cycler-0.11.0 fonttools-4.40.0 kiwisolver-1.4.4 matplotlib-3.7.1 pillow-9.5.0
```

```
[notice] A new release of pip available: 22.2.1 -> 23.1.2
[notice] To update, run: python.exe -m pip install --upgrade pip
```

In [71]:

```python
#import important libraries
import os
import pandas as pd
import requests
import numpy as np
import matplotlib.pyplot as plt
```

In [56]:

```python
# Assign API key
API_KEY="84XJB3CYD1I8056S"
```

In [57]:

```python
# create class and functions
class ScriptData:
    def __init__(self):
        self.raw_scripts = {}
        self.scripts = {}

    def fetch_intraday_data(self, script):
        url = 'https://www.alphavantage.co/query'
        params = {
            "function": "TIME_SERIES_INTRADAY",
            "symbol": script,
            "apikey": API_KEY,
            "interval": "60min"
        }

        try:
            response = requests.get(url, params=params)
            response.raise_for_status()
        except requests.exceptions.HTTPError as e:
            print("Cannot fetch data from Alpha Vantage")
            return

        data = response.json()
        data = data["Time Series (60min)"]
        self.raw_scripts[script] = {"timestamp": [], "open": [], "high": [], "low": [], "close": [], "volume": []}
        for timestamp, values in data.items():
            self.raw_scripts[script]["timestamp"].append(pd.Timestamp(timestamp))
            self.raw_scripts[script]["open"].append(float(values["1. open"]))
            self.raw_scripts[script]["high"].append(float(values["2. high"]))
            self.raw_scripts[script]["low"].append(float(values["3. low"]))
            self.raw_scripts[script]["close"].append(float(values["4. close"]))
            self.raw_scripts[script]["volume"].append(int(values["5. volume"]))

        for key, value in self.raw_scripts[script].items():
            self.raw_scripts[script][key] = reversed(value)

    def convert_intraday_data(self, script):
        if script not in self.raw_scripts.keys():
            print("Please fetch the data for this script first")
            return

        self.scripts[script] = pd.DataFrame.from_dict(self.raw_scripts[script]).sort_values(by=["timestamp"])

    def __getitem__(self, item):
        return self.scripts[item]

    def __setitem__(self, key, value):
        self.scripts[key] = value

    def __contains__(self, item):
        return item in self.scripts.keys()
```

In [58]:

```python
script_data = ScriptData()
script_data.fetch_intraday_data("GOOGL")
script_data.convert_intraday_data("GOOGL")
script_data["GOOGL"]
```

Out[58]:

	timestamp	open	high	low	close	volume
0	2023-06-06 16:00:00	127.3100	127.6900	127.08	127.3900	1111985
1	2023-06-06 17:00:00	127.4500	127.9900	127.31	127.3600	6505
2	2023-06-06 18:00:00	127.3100	127.3100	127.10	127.2000	9130
3	2023-06-06 19:00:00	127.0400	127.2500	127.00	127.2500	15358
4	2023-06-07 04:00:00	127.1800	127.1800	126.82	126.8800	3495
...	...	...	...	...	...	...
95	2023-06-14 15:00:00	123.2400	124.0200	122.88	123.7000	4835358
96	2023-06-14 16:00:00	123.6700	123.8942	123.57	123.8600	1701141
97	2023-06-14 17:00:00	123.6700	123.8876	123.67	123.8876	204798
98	2023-06-14 18:00:00	123.8882	123.8882	123.71	123.7100	8857
99	2023-06-14 19:00:00	123.6800	123.7600	123.55	123.5500	7106

100 rows × 6 columns

In [59]:

```python
script_data.fetch_intraday_data("AAPL")
script_data.convert_intraday_data("AAPL")
script_data["AAPL"]
```

Out[59]:

	timestamp	open	high	low	close	volume
0	2023-06-06 16:00:00	179.16	179.3000	178.93	179.04	1921239
1	2023-06-06 17:00:00	179.04	179.2100	179.00	179.10	40897
2	2023-06-06 18:00:00	179.10	179.1000	178.89	178.90	37185
3	2023-06-06 19:00:00	178.92	179.0500	178.90	179.00	30843
4	2023-06-07 04:00:00	179.07	179.0700	178.69	178.75	14389
...	...	...	...	...	...	...
95	2023-06-14 15:00:00	183.59	184.2050	182.95	183.96	10054258
96	2023-06-14 16:00:00	183.94	184.2500	183.86	184.08	3327983
97	2023-06-14 17:00:00	184.15	184.1500	183.95	184.05	781306
98	2023-06-14 18:00:00	184.06	184.1100	184.00	184.03	23246
99	2023-06-14 19:00:00	184.00	184.0001	183.82	183.88	19704

100 rows × 6 columns

In [60]:

```python
"GOOGL" in script_data
```

Out[60]:

```
True
```

In [61]:

```python
"AAPL" in script_data
```

Out[61]:

```
True
```

In [62]:

```python
"NVDA" in script_data
```

Out[62]:

```
False
```

In [63]:

```python
def indicator1(df: pd.DataFrame, timeperiod: int):
    data = {"timestamp": [], "indicator": []}
    moving_sum = 0

    # Create a moving sum that always has the sum of the timeperiod days ending at the current day

    for index in df.index:
        data["timestamp"].append(df["timestamp"][index])
        moving_sum += df["close"][index]
        if index+1 < timeperiod:
            data["indicator"].append(None)
        else:
            if index+1 > timeperiod:
                moving_sum -= df["close"][index-timeperiod]
            data["indicator"].append(moving_sum/timeperiod)

    return pd.DataFrame.from_dict(data)
```

In [64]:

```python
indicator1(script_data["GOOGL"], 5)
```

Out[64]:

	timestamp	indicator
0	2023-06-06 16:00:00	NaN
1	2023-06-06 17:00:00	NaN
2	2023-06-06 18:00:00	NaN
3	2023-06-06 19:00:00	NaN
4	2023-06-07 04:00:00	127.21600
...	...	...
95	2023-06-14 15:00:00	123.40802
96	2023-06-14 16:00:00	123.46802
97	2023-06-14 17:00:00	123.53352
98	2023-06-14 18:00:00	123.67552
99	2023-06-14 19:00:00	123.74152

100 rows × 2 columns

In [65]:

```python
indicator1(script_data["AAPL"], 5)
```

Out[65]:

	timestamp	indicator
0	2023-06-06 16:00:00	NaN
1	2023-06-06 17:00:00	NaN
2	2023-06-06 18:00:00	NaN
3	2023-06-06 19:00:00	NaN
4	2023-06-07 04:00:00	178.958
...	...	...
95	2023-06-14 15:00:00	183.882
96	2023-06-14 16:00:00	183.880
97	2023-06-14 17:00:00	183.872
98	2023-06-14 18:00:00	183.936
99	2023-06-14 19:00:00	184.000

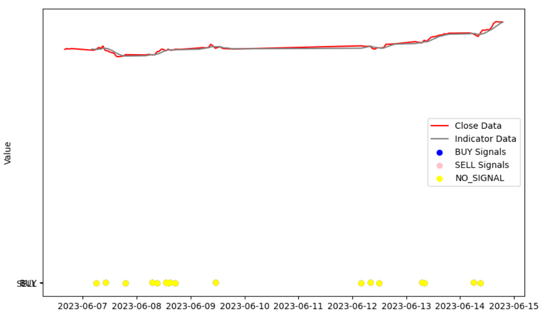
100 rows × 2 columns

In [68]:

```python
class Strategy:
    def __init__(self, script):
        self.script = script
        self.script_data = ScriptData()

    def get_script_data(self):
        self.script_data.fetch_intraday_data(self.script)
        self.script_data.convert_intraday_data(self.script)

    def get_signals(self):
        indicator_data = indicator1(self.script_data[self.script], 5)
        df = self.script_data[self.script]
        signals = pd.DataFrame(columns=["timestamp", "signal"])

        for index in indicator_data.index[1:]:
            if indicator_data["indicator"][index] == "nan":
                continue

            result = {
                "timestamp": [indicator_data["timestamp"][index]],
                "signal": ["NO_SIGNAL"]
            }

            # For BUY signal, previous day's close must be higher than indicator
            # and current day's close must be lower than indicator

            # For SELL signal, previous day's close must be lower than indicator
            # and current day's close must be greater than indicator
            if indicator_data["indicator"][index - 1] < df["close"][index-1] \
                    and df["close"][index] < indicator_data["indicator"][index]:
                result["signal"][0] = "BUY"
            elif indicator_data["indicator"][index - 1] > df["close"][index-1] \
                    and df["close"][index] > indicator_data["indicator"][index]:
                result["signal"][0] = "SELL"

            if result["signal"][0] != "NO_SIGNAL":
                signals = pd.concat([signals, pd.DataFrame.from_dict(result)], ignore_index=True)

        # Plotting the graph
        plt.figure(figsize=(10, 6))
        plt.plot(df['timestamp'], df['close'], color='red', label='Close Data')
        plt.plot(indicator_data['timestamp'], indicator_data['indicator'], color='grey', label='Indicator Data')
        plt.scatter(signals['timestamp'], signals['signal'], color='blue', marker='o', label='BUY Signals')
        plt.scatter(signals['timestamp'], signals['signal'], color='pink', marker='o', label='SELL Signals')
        plt.scatter(signals['timestamp'], signals['signal'], color='yellow', marker='o', label='NO_SIGNAL')

        plt.xlabel('Timestamp')
        plt.ylabel('Value')
        plt.legend()
        plt.show()

        return signals
```

In [69]:

```python
strategy = Strategy("NVDA")
strategy.get_script_data()
strategy.get_signals()
```



Out[69]:

	timestamp	signal
0	2023-06-07 06:00:00	SELL
1	2023-06-07 10:00:00	BUY
2	2023-06-07 19:00:00	SELL
3	2023-06-08 07:00:00	BUY
4	2023-06-08 09:00:00	SELL
5	2023-06-08 13:00:00	BUY
6	2023-06-08 14:00:00	SELL
7	2023-06-08 15:00:00	BUY
8	2023-06-08 17:00:00	SELL
9	2023-06-09 11:00:00	BUY
10	2023-06-12 04:00:00	SELL
11	2023-06-12 08:00:00	BUY
12	2023-06-12 12:00:00	SELL
13	2023-06-13 07:00:00	BUY
14	2023-06-13 08:00:00	SELL
15	2023-06-14 06:00:00	BUY
16	2023-06-14 09:00:00	SELL

In [ ]: