

## Phòng thí nghiệm thực hành: Phân tích dữ liệu cổ phiếu/doanh thu lịch sử và xây dựng bảng thông tin

Trích xuất dữ liệu cần thiết từ một tập dữ liệu và hiển thị nó là một phần cần thiết của khoa học dữ liệu; do đó, các cá nhân có thể đưa ra quyết định đúng đắn dựa trên dữ liệu. Trong bài tập này, bạn sẽ trích xuất một số dữ liệu chứng khoán, sau đó bạn sẽ hiển thị dữ liệu này trong biểu đồ.

**Lưu ý:** - Nếu bạn đang làm việc cục bộ bằng anaconda, vui lòng bỏ chú thích đoạn mã sau và thực thi nó. Sử dụng phiên bản theo phiên bản python của bạn.

1	<code>!pip install yfinance</code>
2	<code>!pip install bs4</code>
3	<code>!pip install nbformat</code>
4	<code>!pip install --upgrade plotly</code>

1	<code>import yfinance as yf</code>
2	<code>import pandas as pd</code>
3	<code>import requests</code>
4	<code>from bs4 import BeautifulSoup</code>
5	<code>import plotly.graph_objects as go</code>
6	<code>from plotly.subplots import make_subplots</code>

1	<code>import plotly.io as pio</code>
2	<code>pio.renderers.default = "iframe"</code>

Trong Python, bạn có thể bỏ qua cảnh báo bằng cách sử dụng mô-đun **warnings**. Bạn có thể sử dụng **hàm filterwarnings** để lọc hoặc bỏ qua các thông báo hoặc danh mục cảnh báo cụ thể.

1	<code>import warnings</code>
2	<code># Ignore all warnings</code>
3	<code>warnings.filterwarnings("ignore", category=FutureWarning)</code>

## 1. Định nghĩa hàm đồ thị

Trong phần này, chúng ta định nghĩa hàm `make_graph`. Bạn không cần biết hàm hoạt động như thế nào, bạn chỉ cần quan tâm đến các đầu vào. Hàm này lấy một khung dữ liệu có dữ liệu chứng khoán (khung dữ liệu phải chứa các cột Date và Close), một khung dữ liệu có dữ liệu doanh thu (khung dữ liệu phải chứa các cột Date và Revenue) và tên của chứng khoán.

```
1 def make_graph(stock_data, revenue_data, stock):
2     fig = make_subplots(rows=2, cols=1, shared_xaxes=True,
3       subplot_titles=("Historical Share Price", "Historical Revenue"),
4       vertical_spacing = .3)
5     stock_data_specific = stock_data[stock_data.Date <= '2021-06-14']
6     revenue_data_specific = revenue_data[revenue_data.Date <= '2021-04-30']
7     fig.add_trace(go.Scatter(x=pd.to_datetime(stock_data_specific.Date,
8       infer_datetime_format=True),
9       y=stock_data_specific.Close.astype("float"), name="Share Price"), row=1,
10      col=1)
11     fig.add_trace(go.Scatter(x=pd.to_datetime(revenue_data_specific.Date,
12       infer_datetime_format=True),
13       y=revenue_data_specific.Revenue.astype("float"), name="Revenue"),
14      row=2, col=1)
15     fig.update_xaxes(title_text="Date", row=1, col=1)
16     fig.update_xaxes(title_text="Date", row=2, col=1)
17     fig.update_yaxes(title_text="Price ($US)", row=1, col=1)
18     fig.update_yaxes(title_text="Revenue ($US Millions)", row=2, col=1)
19     fig.update_layout(showlegend=False,
20       height=900,
21       title=stock,
22       xaxis_rangeflider_visible=True)
23     fig.show()
24     from IPython.display import display, HTML
25     fig_html = fig.to_html()
26     display(HTML(fig_html))
```

Sử dụng hàm `make_graph` mà chúng ta đã định nghĩa. Bạn sẽ cần gọi hàm này trong câu hỏi 5 và 6 để hiển thị biểu đồ và tạo bảng điều khiển.

**Lưu ý: Bạn không cần phải định nghĩa lại hàm để vẽ biểu đồ ở bất kỳ nơi nào khác trong sổ tay này; chỉ cần sử dụng hàm hiện có.**

## Bài tập

### Câu hỏi 1: Sử dụng yfinance để trích xuất dữ liệu chứng khoán

Sử dụng hàm `Ticker` nhập ký hiệu ticker của cổ phiếu mà chúng ta muốn trích xuất dữ liệu để tạo đối tượng ticker. Cổ phiếu là Tesla và ký hiệu ticker của nó là `TSLA`.

```
1 tesla_ticker = yf.Ticker('TSLA')
```

Sử dụng đối tượng ticker và hàm `history` để trích xuất thông tin chứng khoán và lưu vào một khung dữ liệu có tên là `tesla_data`. Đặt tham số `period` thành `"max"` để chúng ta có được thông tin trong khoảng thời gian tối đa.

```
1 tesla_data = tesla_ticker.history(period = 'max')
```

**Đặt lại chỉ mục** bằng hàm `reset_index(inplace=True)` trên DataFrame `tesla_data` và hiển thị năm hàng đầu tiên của dataframe `tesla_data` bằng hàm `head` (*phải dùng `reset_index` vì ban đầu cột Date là chỉ mục*). Chụp ảnh màn hình kết quả và mã từ đầu Câu hỏi 1 đến kết quả bên dưới.

```
1 tesla_data.reset_index(inplace = True)
2 tesla_data.head()
```

The screenshot shows a Jupyter Notebook titled 'Final Assignment.ipynb' in a web browser. The notebook is at the 'Code' tab. The first cell contains the following code and output:

```
[23]: tesla_ticker = yf.Ticker("TSLA")
[23]: yfinance.Ticker object <TSLA>
```

The second cell contains the following code and output:

```
[25]: tesla_data = tesla_ticker.history(period='max')
[25]: tesla_data
```

The output is a DataFrame showing stock data for Tesla from 2010 to 2025. The columns are Date, Open, High, Low, Close, Volume, Dividends, and Stock Splits. The data is displayed in a table format.

Date	Open	High	Low	Close	Volume	Dividends	Stock Splits
2010-06-29 00:00:00-04:00	1.266667	1.666667	1.169333	1.592667	281494500	0.0	0.0
2010-06-30 00:00:00-04:00	1.719333	2.028000	1.553333	1.588667	257806500	0.0	0.0
2010-07-01 00:00:00-04:00	1.666667	1.728000	1.351333	1.464000	123282000	0.0	0.0
2010-07-02 00:00:00-04:00	1.533333	1.540000	1.247333	1.280000	77097000	0.0	0.0
2010-07-06 00:00:00-04:00	1.333333	1.333333	1.055333	1.074000	103003500	0.0	0.0
...	...	...	...	...	...	...	...
2025-06-09 00:00:00-04:00	285.959991	309.829987	281.850006	308.579987	140908900	0.0	0.0

The screenshot shows the same Jupyter Notebook as the previous one, but with additional code and output. The third cell contains the following code and output:

```
[27]: tesla_data.reset_index(inplace=True)
[27]: tesla_data.head()
```

The output is a DataFrame showing the first five rows of the stock data. The columns are Date, Open, High, Low, Close, Volume, Dividends, and Stock Splits. The data is displayed in a table format.

	Date	Open	High	Low	Close	Volume	Dividends	Stock Splits
0	2010-06-29 00:00:00-04:00	1.266667	1.666667	1.169333	1.592667	281494500	0.0	0.0
1	2010-06-30 00:00:00-04:00	1.719333	2.028000	1.553333	1.588667	257806500	0.0	0.0
2	2010-07-01 00:00:00-04:00	1.666667	1.728000	1.351333	1.464000	123282000	0.0	0.0
3	2010-07-02 00:00:00-04:00	1.533333	1.540000	1.247333	1.280000	77097000	0.0	0.0
4	2010-07-06 00:00:00-04:00	1.333333	1.333333	1.055333	1.074000	103003500	0.0	0.0

## Câu hỏi 2: Sử dụng Webscraping để trích xuất dữ liệu doanh thu của Tesla

Sử dụng thư viện `requests` để tải xuống trang web <https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNetwork-PY0220EN-SkillsNetwork/labs/project/revenue.htm> Lưu văn bản phản hồi dưới dạng biến có tên là `html_data`.

- |   |                                                                                                                                                                   |
|---|-------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1 | <code>url = 'https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNetwork-PY0220EN-SkillsNetwork/labs/project/revenue.htm'</code> |
| 2 | <code>html_data = requests.get(url).text</code>                                                                                                                   |

Phân tích dữ liệu html bằng `beautiful_soup` bằng trình phân tích cú pháp như `html5lib` hoặc `html.parser`.

```
1 beautiful_soup = BeautifulSoup(html_data, 'html.parser')
```

Sử dụng `BeautifulSoup` hoặc hàm `read_html` để trích xuất bảng có `Tesla Revenue` và lưu trữ vào một khung dữ liệu có tên là `tesla_revenue`. Khung dữ liệu này phải có các cột `Date` và `Revenue`.

### Hướng dẫn từng bước

Sau đây là hướng dẫn từng bước:

1. Tạo một DataFrame trống
2. Tìm bảng có liên quan
3. Kiểm tra Bảng doanh thu quý của Tesla
4. Lặp qua các hàng trong phần thân bảng
5. Trích xuất dữ liệu từ các cột
6. Thêm dữ liệu vào DataFrame

### Nhấp vào đây nếu bạn cần trợ giúp để định vị bảng

Dưới đây là mã để cô lập bảng, bây giờ bạn sẽ cần lặp qua các hàng và cột như trong phòng thí nghiệm trước

```
soup.find_all("tbody")[1]
```

Nếu bạn muốn sử dụng hàm `read_html`, bảng sẽ nằm ở chỉ mục 1

Chúng tôi đang tập trung vào doanh thu theo quý trong phòng thí nghiệm.

```
1 tesla_revenue = pd.DataFrame(columns = ['Date', 'Revenue'])
2 for row in beautiful_soup.find('tbody').find_all('tr'):
3     col = row.find_all('td')
4     date = col[0].text
5     revenue = col[1].text
6
```

```
7 tesla_revenue = pd.concat([tesla_revenue,pd.DataFrame({'Date':[date],
8 'Revenue':[revenue]})], ignore_index=True)
```

Thực hiện dòng sau để xóa dấu phẩy và dấu đô la khỏi cột **Revenue**.

```
1 tesla_revenue['Revenue'] = tesla_revenue['Revenue'].str.replace(",|\$", "",
  regex=True)
```

**Giải thích:**

- **regex=True**: từ **pandas v1.4.0 trở lên**, str.replace() yêu cầu bạn phải ghi rõ regex=True nếu muốn dùng biểu thức chính quy (regular expression).

Thực hiện các dòng sau để xóa chuỗi null hoặc chuỗi rỗng trong cột **Revenue**.

```
1 tesla_revenue.dropna(inplace=True) # Xóa các dòng có giá trị NaN
2 tesla_revenue = tesla_revenue[tesla_revenue['Revenue'] != ""] # Xóa dòng
  có Revenue là chuỗi rỗng
```

Hiển thị 5 hàng cuối cùng của khung dữ liệu tesla\_revenue bằng hàm tail. Chụp ảnh màn hình kết quả.

```
1 tesla_revenue.tail()
```

The screenshot shows a JupyterLab environment with a file explorer on the left and a code editor on the right. The code editor contains the following code blocks:

```
[51]: tesla_revenue = pd.DataFrame(columns = ['Date','Revenue'])
      for row in BeautifulSoup.find('tbody').find_all('tr'):
          col = row.find_all('td')
          date = col[0].text
          revenue = col[1].text
          tesla_revenue = pd.concat([tesla_revenue,pd.DataFrame({'Date':[date], 'Revenue':[revenue]})], ignore_index=True)
```

Execute the following line to remove the comma and dollar sign from the Revenue column.

```
[76]: tesla_revenue["Revenue"] = tesla_revenue["Revenue"].str.replace(",|\$", "", regex=True)
```

Execute the following lines to remove an null or empty strings in the Revenue column.

```
[77]: tesla_revenue.dropna(inplace=True)
      tesla_revenue = tesla_revenue[tesla_revenue['Revenue'] != ""]
```

Display the last 5 row of the tesla\_revenue dataframe using the tail function. Take a screenshot of the results.

```
[78]: tesla_revenue.tail()
```

The output of the tail function is displayed as a table:

	Date	Revenue
8	2013	2013
9	2012	413
10	2011	204
11	2010	117
12	2009	112

### Câu hỏi 3: Sử dụng yfinance để trích xuất dữ liệu chứng khoán

Sử dụng hàm `Ticker` nhập ký hiệu ticker của cổ phiếu mà chúng ta muốn trích xuất dữ liệu để tạo đối tượng ticker. Cổ phiếu là GameStop và ký hiệu ticker của nó là `GME`.

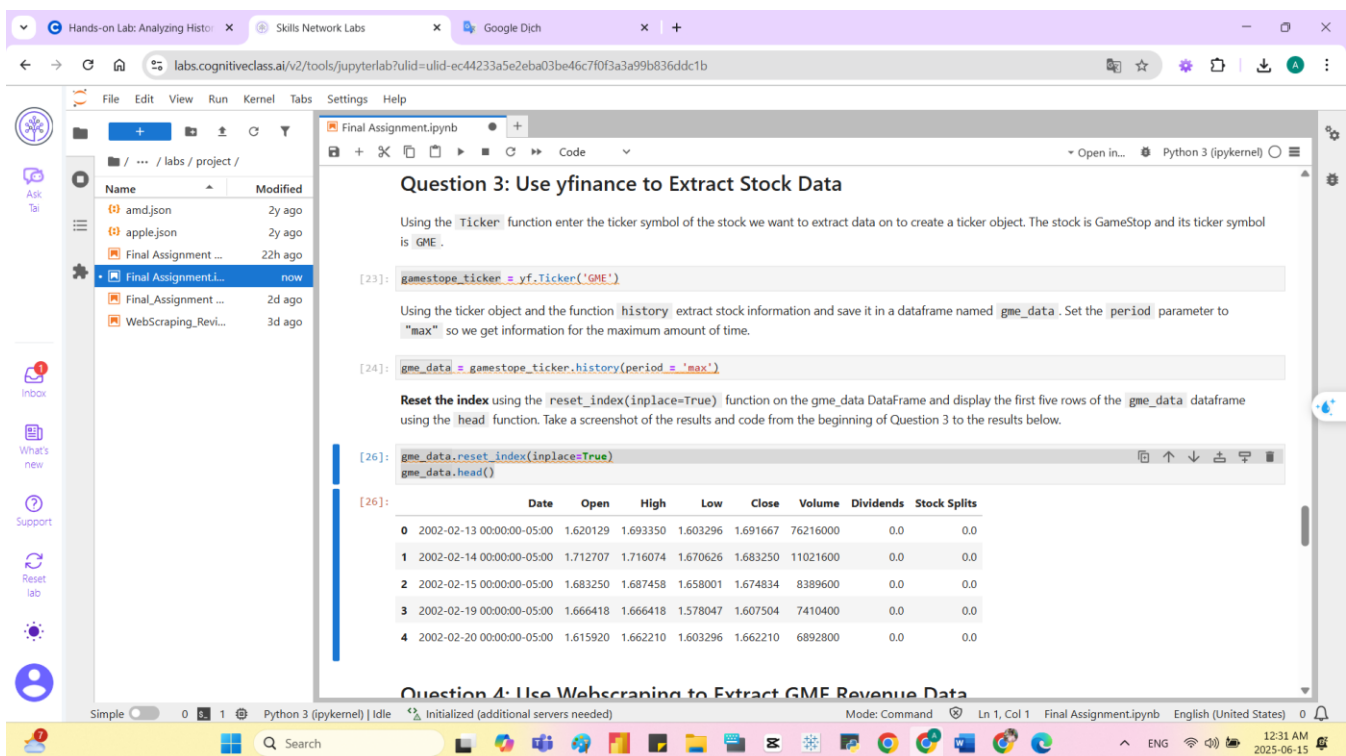
```
1 gamestope_ticker = yf.Ticker('GME')
```

Sử dụng đối tượng ticker và hàm `history` để trích xuất thông tin chứng khoán và lưu vào một khung dữ liệu có tên `gme_data`. Đặt tham số `period` thành `"max"` để chúng ta có được thông tin trong khoảng thời gian tối đa.

```
1 gme_data = gamestope_ticker.history(period = 'max')
```

Đặt lại chỉ mục bằng hàm `reset_index(inplace=True)` trên DataFrame `gme_data` và hiển thị năm hàng đầu tiên của dataframe `gme_data` bằng hàm `head`. Chụp ảnh màn hình kết quả và mã từ đầu Câu hỏi 3 đến kết quả bên dưới.

```
1 gme_data.reset_index(inplace=True)
2 gme_data.head()
```



The screenshot shows a Jupyter Notebook interface with the following content:

**Question 3: Use yfinance to Extract Stock Data**

Using the `Ticker` function enter the ticker symbol of the stock we want to extract data on to create a ticker object. The stock is GameStop and its ticker symbol is `GME`.

```
[23]: gamestope_ticker = yf.Ticker('GME')
```

Using the ticker object and the function `history` extract stock information and save it in a dataframe named `gme_data`. Set the `period` parameter to `"max"` so we get information for the maximum amount of time.

```
[24]: gme_data = gamestope_ticker.history(period = 'max')
```

Reset the index using the `reset_index(inplace=True)` function on the `gme_data` DataFrame and display the first five rows of the `gme_data` dataframe using the `head` function. Take a screenshot of the results and code from the beginning of Question 3 to the results below.

```
[26]: gme_data.reset_index(inplace=True)
gme_data.head()
```

```
[26]:
```

	Date	Open	High	Low	Close	Volume	Dividends	Stock Splits
0	2002-02-13 00:00:00-05:00	1.620129	1.693350	1.603296	1.691667	76216000	0.0	0.0
1	2002-02-14 00:00:00-05:00	1.712707	1.716074	1.670626	1.683250	11021600	0.0	0.0
2	2002-02-15 00:00:00-05:00	1.683250	1.687458	1.658001	1.674834	8389600	0.0	0.0
3	2002-02-19 00:00:00-05:00	1.666418	1.666418	1.578047	1.607504	7410400	0.0	0.0
4	2002-02-20 00:00:00-05:00	1.615920	1.662210	1.603296	1.662210	6892800	0.0	0.0

**Question 4: Use Web scraping to Extract GME Revenue Data**

#### Câu hỏi 4: Sử dụng Webscraping để trích xuất dữ liệu doanh thu GME

Sử dụng thư viện `requests` để tải xuống trang web <https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNetwork-PY0220EN-SkillsNetwork/labs/project/stock.html>. Lưu văn bản phản hồi dưới dạng biến có tên `html_data_2`.

```
1 url1 = 'https://cf-courses-data.s3.us.cloud-object-  
storage.appdomain.cloud/IBMDeveloperSkillsNetwork-PY0220EN-  
SkillsNetwork/labs/project/stock.html'  
2 html_data_2 = requests.get(url1).text
```

Phân tích dữ liệu html bằng `beautiful_soup` bằng trình phân tích cú pháp như `html5lib` hoặc `html.parser`.

```
1 beautiful_soup2 = BeautifulSoup(html_data_2, 'html.parser')
```

Sử dụng `BeautifulSoup` hoặc hàm `read_html` để trích xuất bảng có `GameStop Revenue` và lưu trữ vào một khung dữ liệu có tên `gme_revenue`. Khung dữ liệu phải có các cột `Date` và `Revenue`. Đảm bảo dấu phẩy và dấu đô la đã được xóa khỏi cột `Revenue`.

**Lưu ý:** Sử dụng phương pháp tương tự như những gì bạn đã làm ở câu hỏi 2.

```
1 gme_revenue = pd.DataFrame(columns = ['Date','Revenue'])  
2 for row in beautiful_soup2.find('tbody').find_all('tr'):  
3     col = row.find_all('td')  
4     date = col[0].text  
5     revenue = col[1].text  
6     gme_revenue = pd.concat([gme_revenue, pd.DataFrame({'Date':[date],  
'Revenue':[revenue]}), ignore_index=True)  
7  
8  
9     gme_revenue['Revenue'] = gme_revenue['Revenue'].str.replace(",|\$", "",  
10    regex=True)  
10 gme_revenue.dropna(inplace=True)
```



```
11 gme_revenue = gme_revenue[gme_revenue['Revenue'] != ""]
```

Hiện thị năm hàng cuối cùng của khung dữ liệu `gme_revenue` bằng hàm `tail`.  
Chụp ảnh màn hình kết quả.

```
1 gme_revenue.tail()
```

The screenshot shows a JupyterLab environment with a file explorer on the left, a code editor in the center, and a console/output area at the bottom. The code editor contains the following Python code:

```
[79]: gme_revenue = pd.DataFrame(columns = ['Date', 'Revenue'])
for row in BeautifulSoup2.find('tbody').find_all('tr'):
    col = row.find_all('td')
    date = col[0].text
    revenue = col[1].text
    gme_revenue = pd.concat([gme_revenue, pd.DataFrame({'Date': [date], 'Revenue': [revenue]}), ignore_index=True])

gme_revenue['Revenue'] = gme_revenue['Revenue'].str.replace(",", "\\", regex=True)
gme_revenue.dropna(inplace=True)
gme_revenue = gme_revenue[gme_revenue['Revenue'] != ""]
```

Below the code, there is a text instruction: "Display the last five rows of the `gme_revenue` dataframe using the `tail` function. Take a screenshot of the results."

The output of the `tail()` function is displayed in the console:

```
[80]: gme_revenue.tail()
[80]:   Date  Revenue
11  2009     8806
12  2008     7094
13  2007     5319
14  2006     3092
15  2005     1843
```

The bottom of the screenshot shows the Windows taskbar with the time 12:27 AM on 2025-06-16.

## Câu hỏi 5: Vẽ đồ thị cổ phiếu Tesla

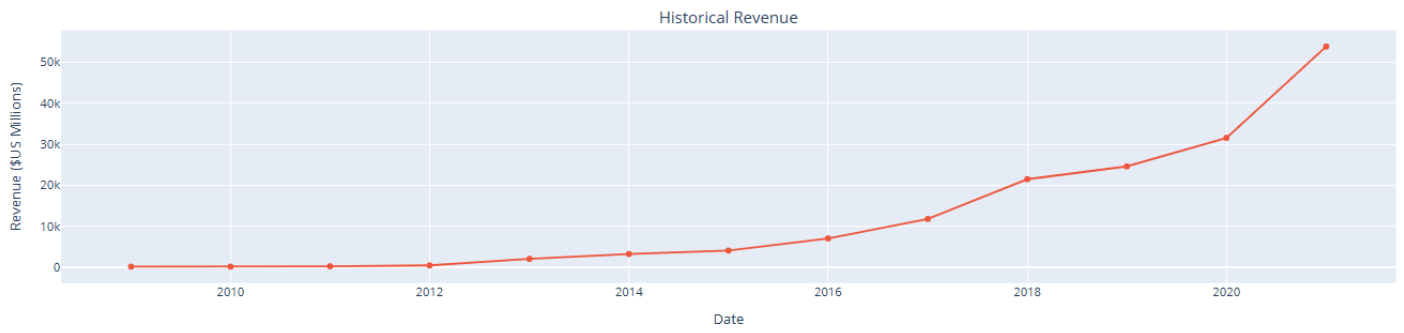
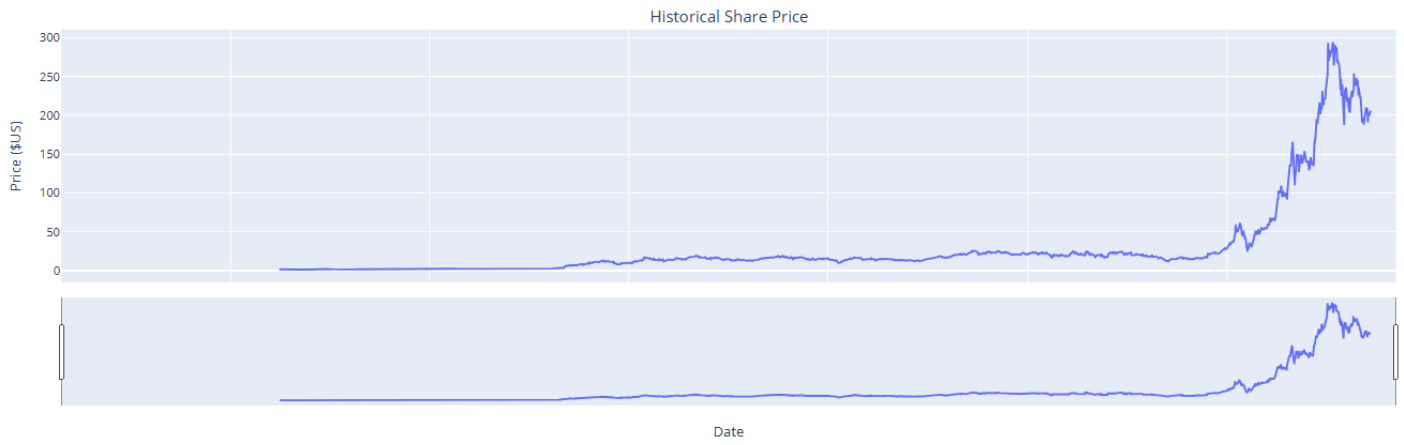
Sử dụng hàm `make_graph` để vẽ đồ thị Dữ liệu cổ phiếu Tesla, đồng thời cung cấp tiêu đề cho đồ thị. Lưu ý đồ thị sẽ chỉ hiển thị dữ liệu đến tháng 6 năm 2021.

### Gợi ý

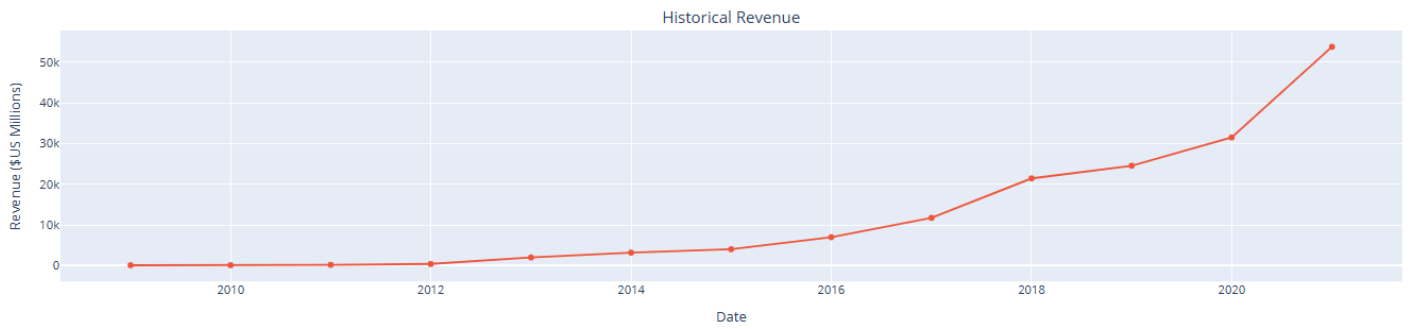
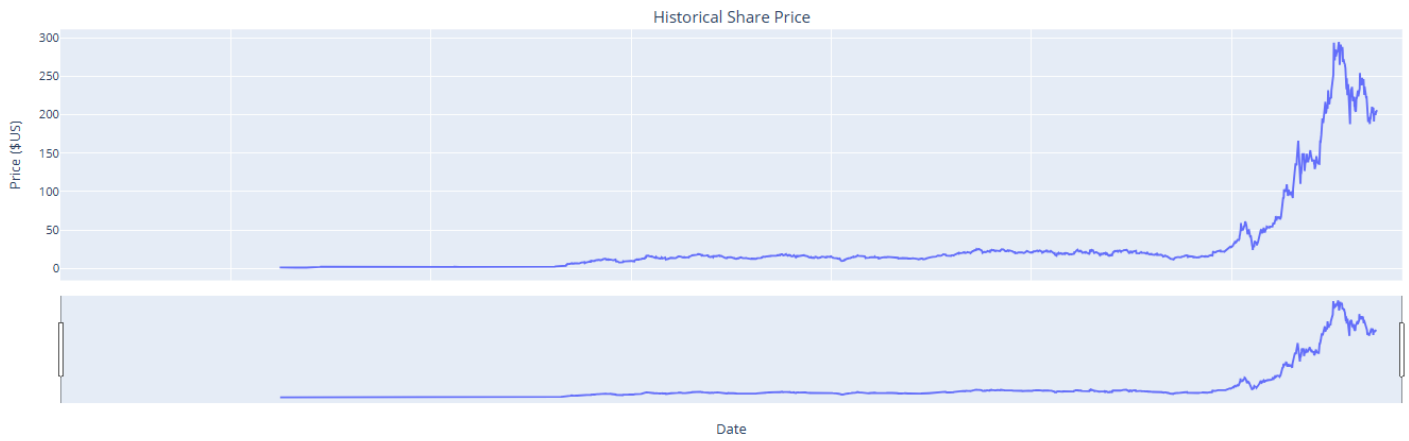
Bạn chỉ cần gọi hàm `make_graph` với tham số cần thiết để in biểu đồ. Cấu trúc để gọi hàm `'make_graph'` là `'make_graph(tesla_data, tesla_revenue, 'Tesla')`.

```
1 make_graph(tesla_data, tesla_revenue, 'Tesla')
```

## Tesla



## Tesla



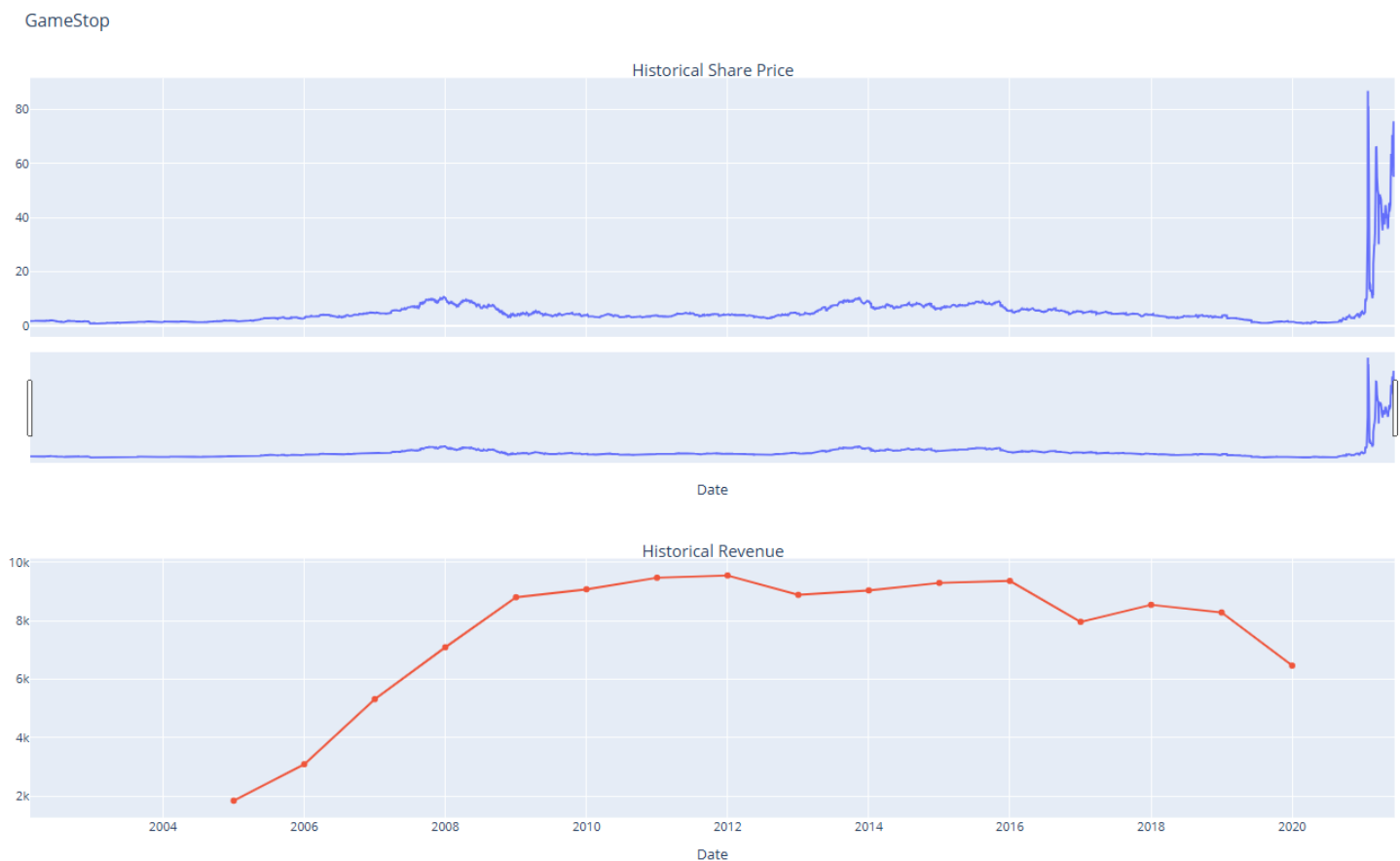
## Câu hỏi 6: Vẽ đồ thị cổ phiếu GameStop

Sử dụng hàm `make_graph` để vẽ đồ thị Dữ liệu cổ phiếu GameStop, đồng thời cung cấp tiêu đề cho đồ thị. Cấu trúc để gọi hàm `make_graph` là `make_graph(gme_data, gme_revenue, 'GameStop')`. Lưu ý rằng đồ thị sẽ chỉ hiển thị dữ liệu đến tháng 6 năm 2021.

### Gợi ý

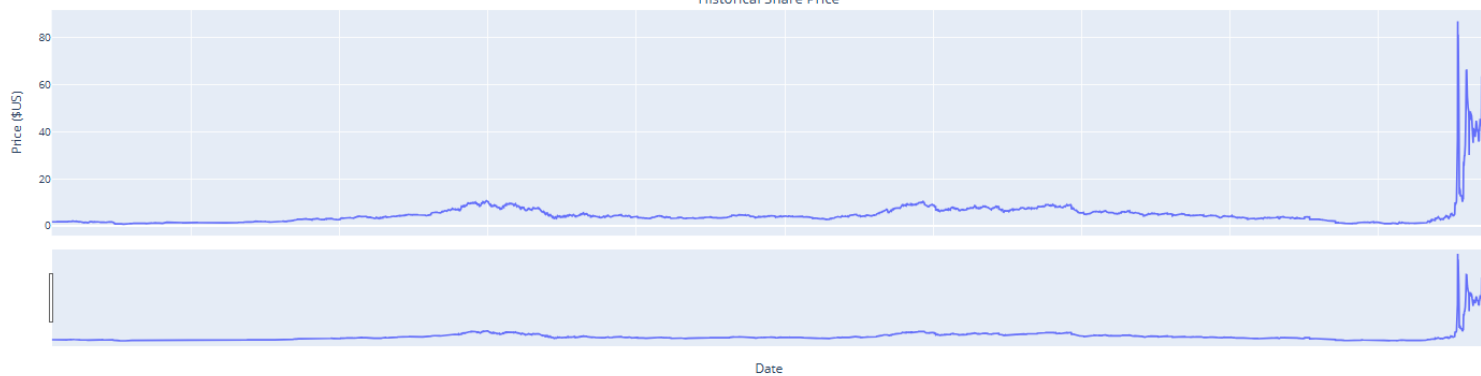
Bạn chỉ cần gọi hàm `make_graph` với tham số bắt buộc để in biểu đồ. Cấu trúc để gọi hàm `'make_graph'` là `'make_graph(gme_data, gme_revenue, 'GameStop')`

1	<code>make_graph(gme_data, gme_revenue, 'GameStop')</code>
---	------------------------------------------------------------



# GameStop

## Historical Share Price



## Historical Revenue

