SALES ANALYSIS

In [1]:
```
import pandas as pd
import os
```

Task1: Merging 12 months of sales data into single file

In [2]:
```
path = "./sales_data"
files = [file for file in os.listdir(path) if not file.startswith('.')] # Ig

all_months_data = pd.DataFrame()

for file in files:
    current_data = pd.read_csv(path+"/"+file)
    all_months_data = pd.concat([all_months_data, current_data])

all_months_data.to_csv("all_data_copy.csv", index=False)
```

Read in Updated DataFrame

In [3]:
```
all_data = pd.read_csv("all_data_copy.csv")
all_data.head()
```

Out[3]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address
0	295665	Macbook Pro Laptop	1	1700	12/30/19 00:01	136 Church St, New York City, NY 10001
1	295666	LG Washing Machine	1	600.0	12/29/19 07:03	562 2nd St, New York City, NY 10001
2	295667	USB-C Charging Cable	1	11.95	12/12/19 18:21	277 Main St, New York City, NY 10001
3	295668	27in FHD Monitor	1	149.99	12/22/19 15:13	410 6th St, San Francisco, CA 94016
4	295669	USB-C Charging Cable	1	11.95	12/18/19 12:38	43 Hill St, Atlanta, GA 30301

Clean up the data

Drop the NAN!

In [4]:
```
nan_df = all_data[all_data.isna().any(axis=1)]
nan_df.head()

all_data = all_data.dropna(how ="all")
```

In [5]:
```
all_data.head()
```

Out[5]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address
0	295665	Macbook Pro Laptop	1	1700	12/30/19 00:01	136 Church St, New York City, NY 10001
1	295666	LG Washing Machine	1	600.0	12/29/19 07:03	562 2nd St, New York City, NY 10001
2	295667	USB-C Charging Cable	1	11.95	12/12/19 18:21	277 Main St, New York City, NY 10001
3	295668	27in FHD Monitor	1	149.99	12/22/19 15:13	410 6th St, San Francisco, CA 94016
4	295669	USB-C Charging Cable	1	11.95	12/18/19 12:38	43 Hill St, Atlanta, GA 30301

Checking for "Or" as it might the duplicate the headings for every single .csv file

In [6]:
```
all_data = all_data[all_data["Order Date"].str[0:2] != "Or"]
```

In [7]:
```
all_data.head()
```

Out[7]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address
0	295665	Macbook Pro Laptop	1	1700	12/30/19 00:01	136 Church St, New York City, NY 10001
1	295666	LG Washing Machine	1	600.0	12/29/19 07:03	562 2nd St, New York City, NY 10001
2	295667	USB-C Charging Cable	1	11.95	12/12/19 18:21	277 Main St, New York City, NY 10001
3	295668	27in FHD Monitor	1	149.99	12/22/19 15:13	410 6th St, San Francisco, CA 94016
4	295669	USB-C Charging Cable	1	11.95	12/18/19 12:38	43 Hill St, Atlanta, GA 30301

Augment Data with additional columns

Task2: Adding the month column

In [8]:
```
all_data["Month"] = all_data["Order Date"].str[0:2]
all_data["Month"] = all_data["Month"].astype("int32")
all_data.head()
```

Out[8]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month
0	295665	Macbook Pro Laptop	1	1700	12/30/19 00:01	136 Church St, New York City, NY 10001	12
1	295666	LG Washing Machine	1	600.0	12/29/19 07:03	562 2nd St, New York City, NY 10001	12
2	295667	USB-C Charging Cable	1	11.95	12/12/19 18:21	277 Main St, New York City, NY 10001	12
3	295668	27in FHD Monitor	1	149.99	12/22/19 15:13	410 6th St, San Francisco, CA 94016	12
4	295669	USB-C Charging Cable	1	11.95	12/18/19 12:38	43 Hill St, Atlanta, GA 30301	12

Make the Quantity Ordered and Price Each column numeric

In [9]:
```
all_data["Quantity Ordered"] = pd.to_numeric(all_data["Quantity Ordered"]) #
all_data["Price Each"] = pd.to_numeric(all_data["Price Each"]) #Make float
```

Task3: Add a sales column--> Quantity * Price

In [10]:
```
all_data["Sales"] = all_data["Quantity Ordered"] * all_data["Price Each"]
all_data.head()
```

Out[10]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sales
0	295665	Macbook Pro Laptop	1	1700.00	12/30/19 00:01	136 Church St, New York City, NY 10001	12	1700.00
1	295666	LG Washing Machine	1	600.00	12/29/19 07:03	562 2nd St, New York City, NY 10001	12	600.00
2	295667	USB-C Charging Cable	1	11.95	12/12/19 18:21	277 Main St, New York City, NY 10001	12	11.95
3	295668	27in FHD Monitor	1	149.99	12/22/19 15:13	410 6th St, San Francisco, CA 94016	12	149.99
4	295669	USB-C Charging Cable	1	11.95	12/18/19 12:38	43 Hill St, Atlanta, GA 30301	12	11.95

Task4: Add a City column

In [12]:
```
# Let's use .apply() method

all_data["City"] = all_data["Purchase Address"].apply(lambda x: x.split(",")

all_data.head()
```

Out[12]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sales	City
0	295665	Macbook Pro Laptop	1	1700.00	12/30/19 00:01	136 Church St, New York City, NY 10001	12	1700.00	New York City (NY)
1	295666	LG Washing Machine	1	600.00	12/29/19 07:03	562 2nd St, New York City, NY 10001	12	600.00	New York City (NY)
2	295667	USB-C Charging Cable	1	11.95	12/12/19 18:21	277 Main St, New York City, NY 10001	12	11.95	New York City (NY)
3	295668	27in FHD Monitor	1	149.99	12/22/19 15:13	410 6th St, San Francisco, CA 94016	12	149.99	San Francisco (CA)
4	295669	USB-C Charging Cable	1	11.95	12/18/19 12:38	43 Hill St, Atlanta, GA 30301	12	11.95	Atlanta (GA)

In [ ]:

```
```

Q1: What was the best month for sale? How much was earned in that month?

In [13]:

```
# For this we gonna need price column, so we made one extra column for sales
result = all_data.groupby("Month").sum()
result
```

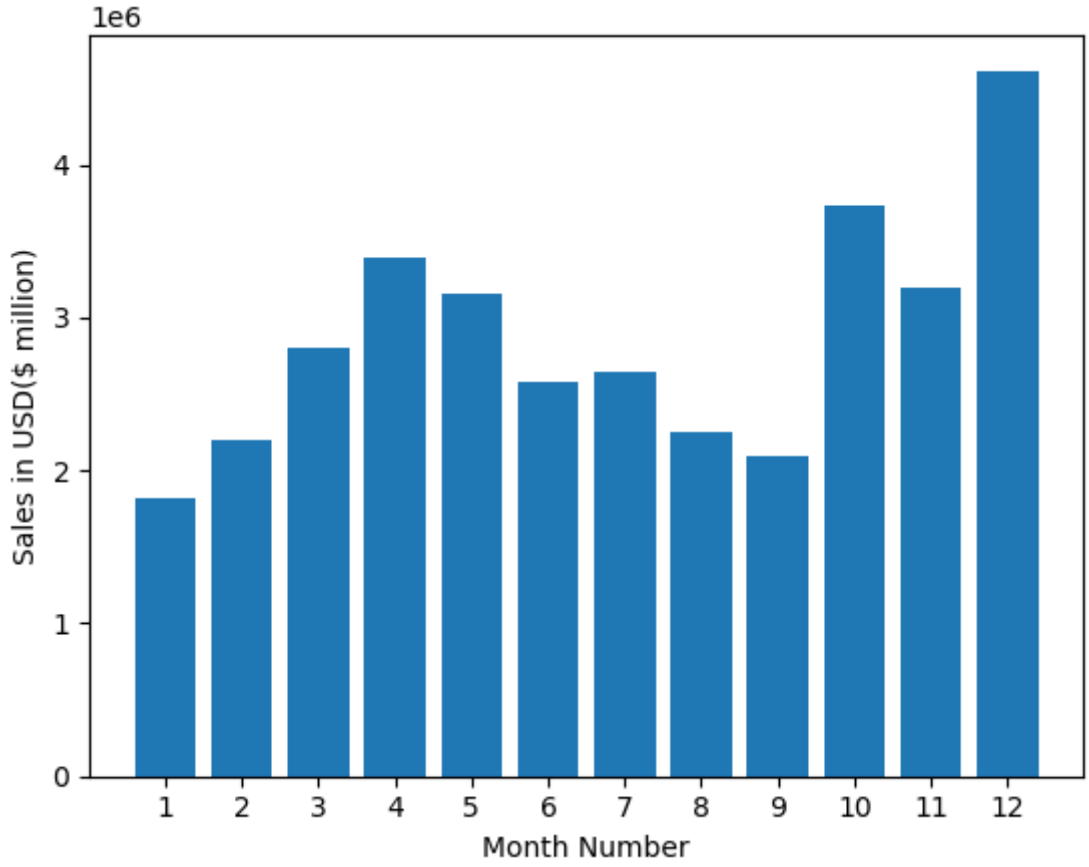
Out[13]:

Month	Quantity Ordered	Price Each	Sales
1	10903	1811768.38	1822256.73
2	13449	2188884.72	2202022.42
3	17005	2791207.83	2807100.38
4	20558	3367671.02	3390670.24
5	18667	3135125.13	3152606.75
6	15253	2562025.61	2577802.26
7	16072	2632539.56	2647775.76
8	13448	2230345.42	2244467.88
9	13109	2084992.09	2097560.13
10	22703	3715554.83	3736726.88
11	19798	3180600.68	3199603.20
12	28114	4588415.41	4613443.34

In [14]:

```
import matplotlib.pyplot as plt
```

```
months = range(1,13)

plt.bar(months, result["Sales"])
plt.xticks(months)
plt.ylabel("Sales in USD($ million)")
plt.xlabel("Month Number")
plt.show()
```



ANS1: December has the best sales and the sale is $4613443.34

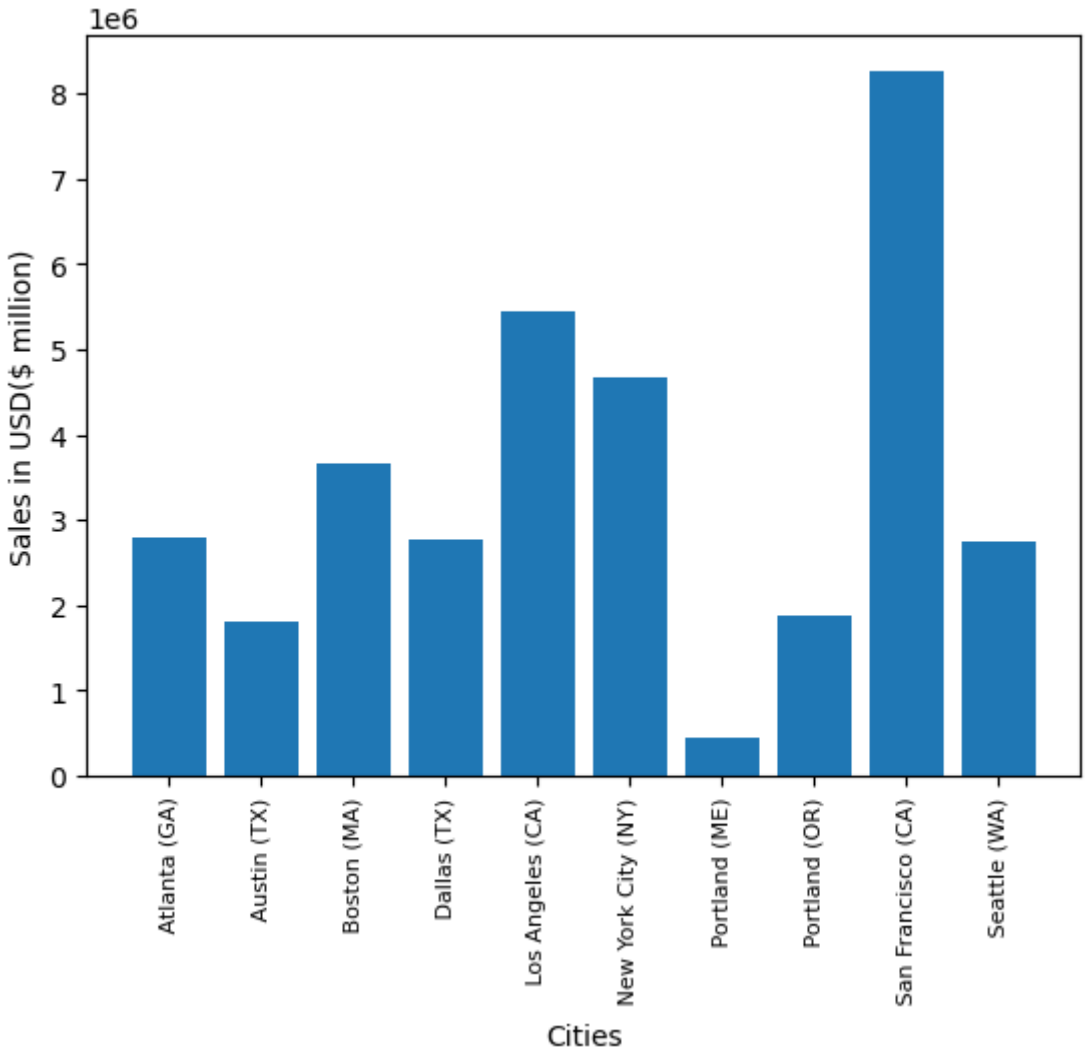
Q2: What city has the highest number of sales?

In [15]:
```
# we added a City column

results = all_data.groupby("City").sum()
results
```

Out[15]:

City	Quantity Ordered	Price Each	Month	Sales
Atlanta (GA)	16602	2779908.20	104794	2795498.58
Austin (TX)	11153	1809873.61	69829	1819581.75
Boston (MA)	22528	3637409.77	141112	3661642.01
Dallas (TX)	16730	2752627.82	104620	2767975.40
Los Angeles (CA)	33289	5421435.23	208325	5452570.80
New York City (NY)	27932	4635370.83	175741	4664317.43
Portland (ME)	2750	447189.25	17144	449758.27
Portland (OR)	11303	1860558.22	70621	1870732.34
San Francisco (CA)	50239	8211461.74	315520	8262203.91
Seattle (WA)	16553	2733296.01	104941	2747755.48

In [16]:
```
city = [city for city, df in all_data.groupby("City")]

plt.bar(city, results["Sales"])
plt.xticks(city, rotation = "vertical", size = 8)
plt.ylabel("Sales in USD($ million)")
plt.xlabel("Cities")
plt.show()
```



ANS2: San Fransisco (CA) has the highest number of sales

Q3: What time should we display advertisement to maximize likelihood of customers buying product?

In [17]:
```
all_data.head()
```

Out[17]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sales	City
0	295665	Macbook Pro Laptop	1	1700.00	12/30/19 00:01	136 Church St, New York City, NY 10001	12	1700.00	New York City (NY)
1	295666	LG Washing Machine	1	600.00	12/29/19 07:03	562 2nd St, New York City, NY 10001	12	600.00	New York City (NY)
2	295667	USB-C Charging Cable	1	11.95	12/12/19 18:21	277 Main St, New York City, NY 10001	12	11.95	New York City (NY)
3	295668	27in FHD Monitor	1	149.99	12/22/19 15:13	410 6th St, San Francisco, CA 94016	12	149.99	San Francisco (CA)
4	295669	USB-C Charging Cable	1	11.95	12/18/19 12:38	43 Hill St, Atlanta, GA 30301	12	11.95	Atlanta (GA)

In [19]:
```
all_data["Order Date"] = pd.to_datetime(all_data["Order Date"])
```

In [20]:
```
all_data.head()
```

Out[20]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sales	City
0	295665	Macbook Pro Laptop	1	1700.00	2019-12-30 00:01:00	136 Church St, New York City, NY 10001	12	1700.00	New York City (NY)
1	295666	LG Washing Machine	1	600.00	2019-12-29 07:03:00	562 2nd St, New York City, NY 10001	12	600.00	New York City (NY)
2	295667	USB-C Charging Cable	1	11.95	2019-12-12 18:21:00	277 Main St, New York City, NY 10001	12	11.95	New York City (NY)
3	295668	27in FHD Monitor	1	149.99	2019-12-22 15:13:00	410 6th St, San Francisco, CA 94016	12	149.99	San Francisco (CA)
4	295669	USB-C Charging Cable	1	11.95	2019-12-18 12:38:00	43 Hill St, Atlanta, GA 30301	12	11.95	Atlanta (GA)

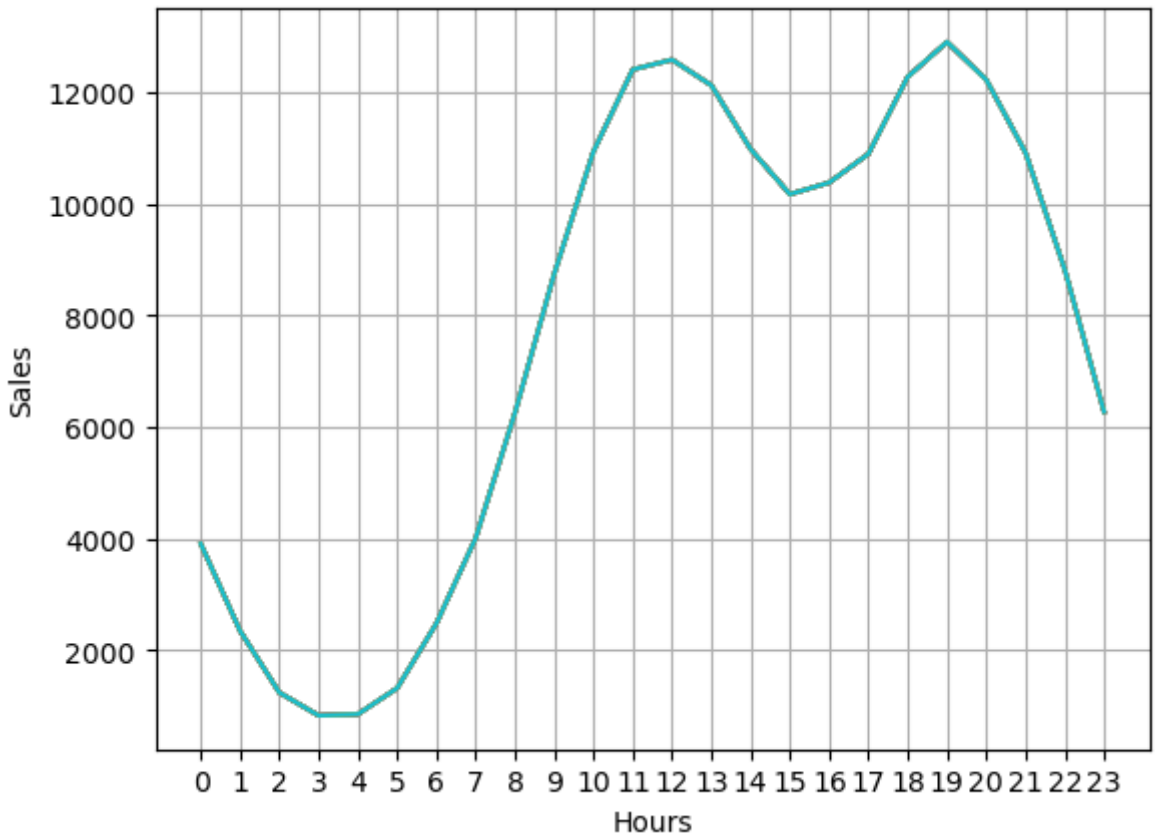
In [22]:
```
all_data["Hour"] = all_data["Order Date"].dt.hour
all_data["Minute"] = all_data["Order Date"].dt.minute
all_data.head()
```

Out[22]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sales	City	Hou
0	295665	Macbook Pro Laptop	1	1700.00	2019-12-30 00:01:00	136 Church St, New York City, NY 10001	12	1700.00	New York City (NY)	
1	295666	LG Washing Machine	1	600.00	2019-12-29 07:03:00	562 2nd St, New York City, NY 10001	12	600.00	New York City (NY)	
2	295667	USB-C Charging Cable	1	11.95	2019-12-12 18:21:00	277 Main St, New York City, NY 10001	12	11.95	New York City (NY)	1
3	295668	27in FHD Monitor	1	149.99	2019-12-22 15:13:00	410 6th St, San Francisco, CA 94016	12	149.99	San Francisco (CA)	1
4	295669	USB-C Charging Cable	1	11.95	2019-12-18 12:38:00	43 Hill St, Atlanta, GA 30301	12	11.95	Atlanta (GA)	1

In [27]:
```
hours = [hour for hour, df in all_data.groupby("Hour")]

plt.plot(hours, all_data.groupby(["Hour"]).count())
plt.xticks(hours)
plt.grid()
plt.xlabel("Hours")
plt.ylabel("Sales")
```

```
plt.show()

#My recomendation is around 11am to 8pm to advertise to maximize the likelih
```



Q3: What products are most often sold together?

In [29]:
```
all_data.head()
#if the order_id is the same that means products were ordered together
```

Out[29]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sales	City	Hou
0	295665	Macbook Pro Laptop	1	1700.00	2019-12-30 00:01:00	136 Church St, New York City, NY 10001	12	1700.00	New York City (NY)	
1	295666	LG Washing Machine	1	600.00	2019-12-29 07:03:00	562 2nd St, New York City, NY 10001	12	600.00	New York City (NY)	
2	295667	USB-C Charging Cable	1	11.95	2019-12-12 18:21:00	277 Main St, New York City, NY 10001	12	11.95	New York City (NY)	1
3	295668	27in FHD Monitor	1	149.99	2019-12-22 15:13:00	410 6th St, San Francisco, CA 94016	12	149.99	San Francisco (CA)	1
4	295669	USB-C Charging Cable	1	11.95	2019-12-18 12:38:00	43 Hill St, Atlanta, GA 30301	12	11.95	Atlanta (GA)	1

In [31]:
```
df = all_data[all_data["Order ID"].duplicated(keep=False)]
df.head(6)
```

Out[31]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sales	City	Hou
16	295681	Google Phone	1	600.00	2019-12-25 12:37:00	79 Elm St, Boston, MA 02215	12	600.00	Boston (MA)	1
17	295681	USB-C Charging Cable	1	11.95	2019-12-25 12:37:00	79 Elm St, Boston, MA 02215	12	11.95	Boston (MA)	1
18	295681	Bose SoundSport Headphones	1	99.99	2019-12-25 12:37:00	79 Elm St, Boston, MA 02215	12	99.99	Boston (MA)	1
19	295681	Wired Headphones	1	11.99	2019-12-25 12:37:00	79 Elm St, Boston, MA 02215	12	11.99	Boston (MA)	1
36	295698	Vareebadd Phone	1	400.00	2019-12-13 14:32:00	175 1st St, New York City, NY 10001	12	400.00	New York City (NY)	1
37	295698	USB-C Charging Cable	2	11.95	2019-12-13 14:32:00	175 1st St, New York City, NY 10001	12	23.90	New York City (NY)	1

In [33]:
```
df["Grouped"] = df.groupby("Order ID")["Product"].transform(lambda x: ",".jo

df = df[["Order ID", "Grouped"]].drop_duplicates()

df.head()
```

```
/var/folders/d1/qkfg3wgx4b1f8kmsbs6l6p6m0000gn/T/ipykernel_56388/3206138208.
py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/
stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df["Grouped"] = df.groupby("Order ID")["Product"].transform(lambda x: ",".
join(x))
```

Out[33]:

	Order ID	Grouped
16	295681	Google Phone,USB-C Charging Cable,Bose SoundSp...
36	295698	Vareebadd Phone,USB-C Charging Cable
42	295703	AA Batteries (4-pack),Bose SoundSport Headphones
66	295726	iPhone,Lightning Charging Cable
76	295735	iPhone,Apple Airpods Headphones,Wired Headphones

In [36]:
```
# Counting pairs of products for products which are order in two
from itertools import combinations
from collections import Counter

count = Counter()

for row in df["Grouped"]:
    row_list = row.split(",")
    count.update(Counter(combinations(row_list, 2)))

for key, value in count.most_common(10):
    print(key, value)
```

```
('iPhone', 'Lightning Charging Cable') 1005
('Google Phone', 'USB-C Charging Cable') 987
('iPhone', 'Wired Headphones') 447
('Google Phone', 'Wired Headphones') 414
('Vareebadd Phone', 'USB-C Charging Cable') 361
('iPhone', 'Apple Airpods Headphones') 360
('Google Phone', 'Bose SoundSport Headphones') 220
('USB-C Charging Cable', 'Wired Headphones') 160
('Vareebadd Phone', 'Wired Headphones') 143
('Lightning Charging Cable', 'Wired Headphones') 92
```

In [40]:
```
# Counting pairs of products for products which are order in three
count = Counter()

for row in df["Grouped"]:
    row_list = row.split(",")
    count.update(Counter(combinations(row_list, 3)))

for key, value in count.most_common(10):
    print(key, value)
```

```
('Google Phone', 'USB-C Charging Cable', 'Wired Headphones') 87
('iPhone', 'Lightning Charging Cable', 'Wired Headphones') 62
('iPhone', 'Lightning Charging Cable', 'Apple Airpods Headphones') 47
('Google Phone', 'USB-C Charging Cable', 'Bose SoundSport Headphones') 35
('Vareebadd Phone', 'USB-C Charging Cable', 'Wired Headphones') 33
('iPhone', 'Apple Airpods Headphones', 'Wired Headphones') 27
('Google Phone', 'Bose SoundSport Headphones', 'Wired Headphones') 24
('Vareebadd Phone', 'USB-C Charging Cable', 'Bose SoundSport Headphones') 16
('USB-C Charging Cable', 'Bose SoundSport Headphones', 'Wired Headphones') 5
('Vareebadd Phone', 'Bose SoundSport Headphones', 'Wired Headphones') 5
```

Q3: What product sold the most? Why do you think it sold the most?

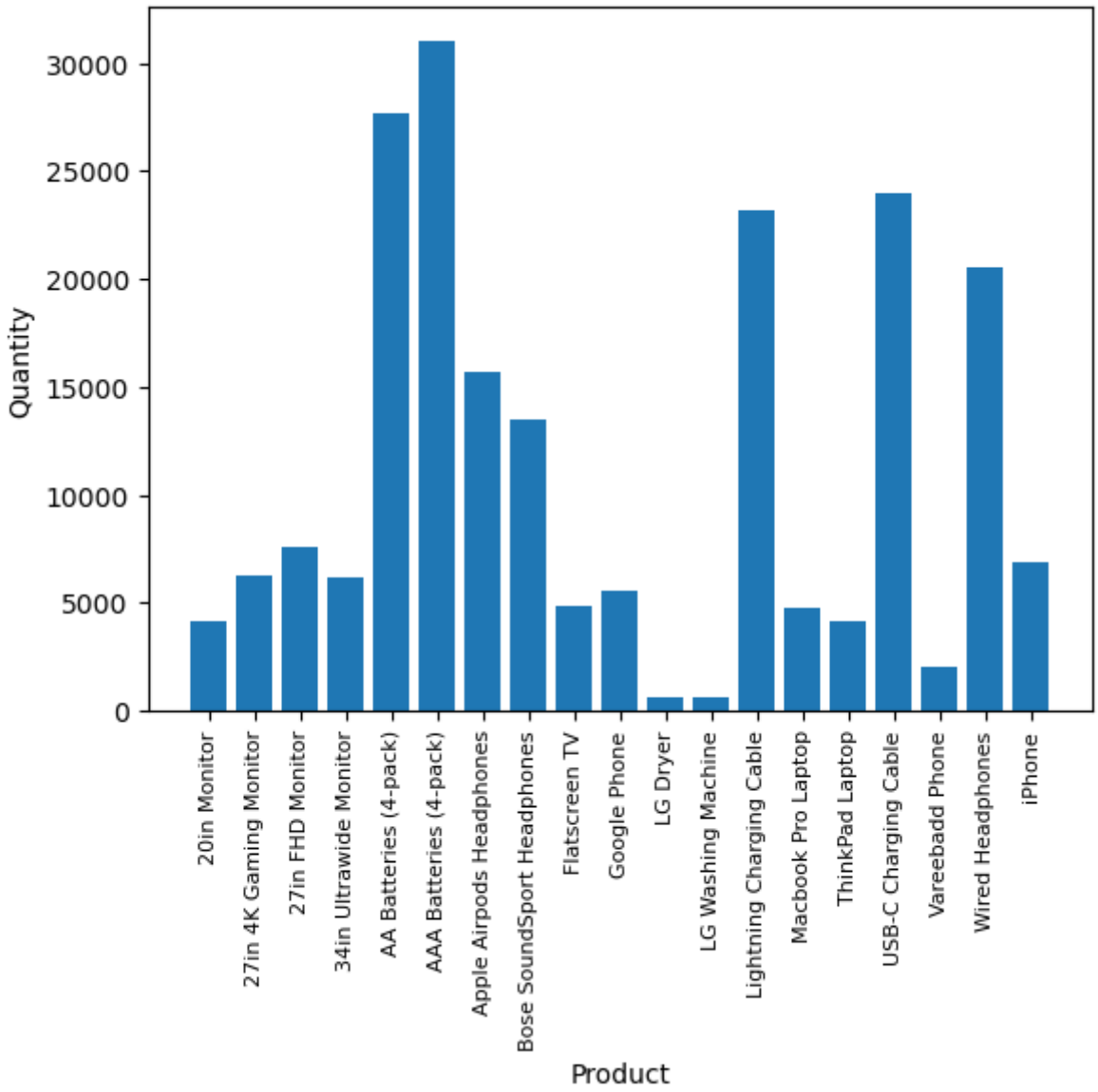
In [41]:
```
all_data.head()
```

Out[41]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sales	City	Hou
0	295665	Macbook Pro Laptop	1	1700.00	2019-12-30 00:01:00	136 Church St, New York City, NY 10001	12	1700.00	New York City (NY)	
1	295666	LG Washing Machine	1	600.00	2019-12-29 07:03:00	562 2nd St, New York City, NY 10001	12	600.00	New York City (NY)	
2	295667	USB-C Charging Cable	1	11.95	2019-12-12 18:21:00	277 Main St, New York City, NY 10001	12	11.95	New York City (NY)	1
3	295668	27in FHD Monitor	1	149.99	2019-12-22 15:13:00	410 6th St, San Francisco, CA 94016	12	149.99	San Francisco (CA)	1
4	295669	USB-C Charging Cable	1	11.95	2019-12-18 12:38:00	43 Hill St, Atlanta, GA 30301	12	11.95	Atlanta (GA)	1

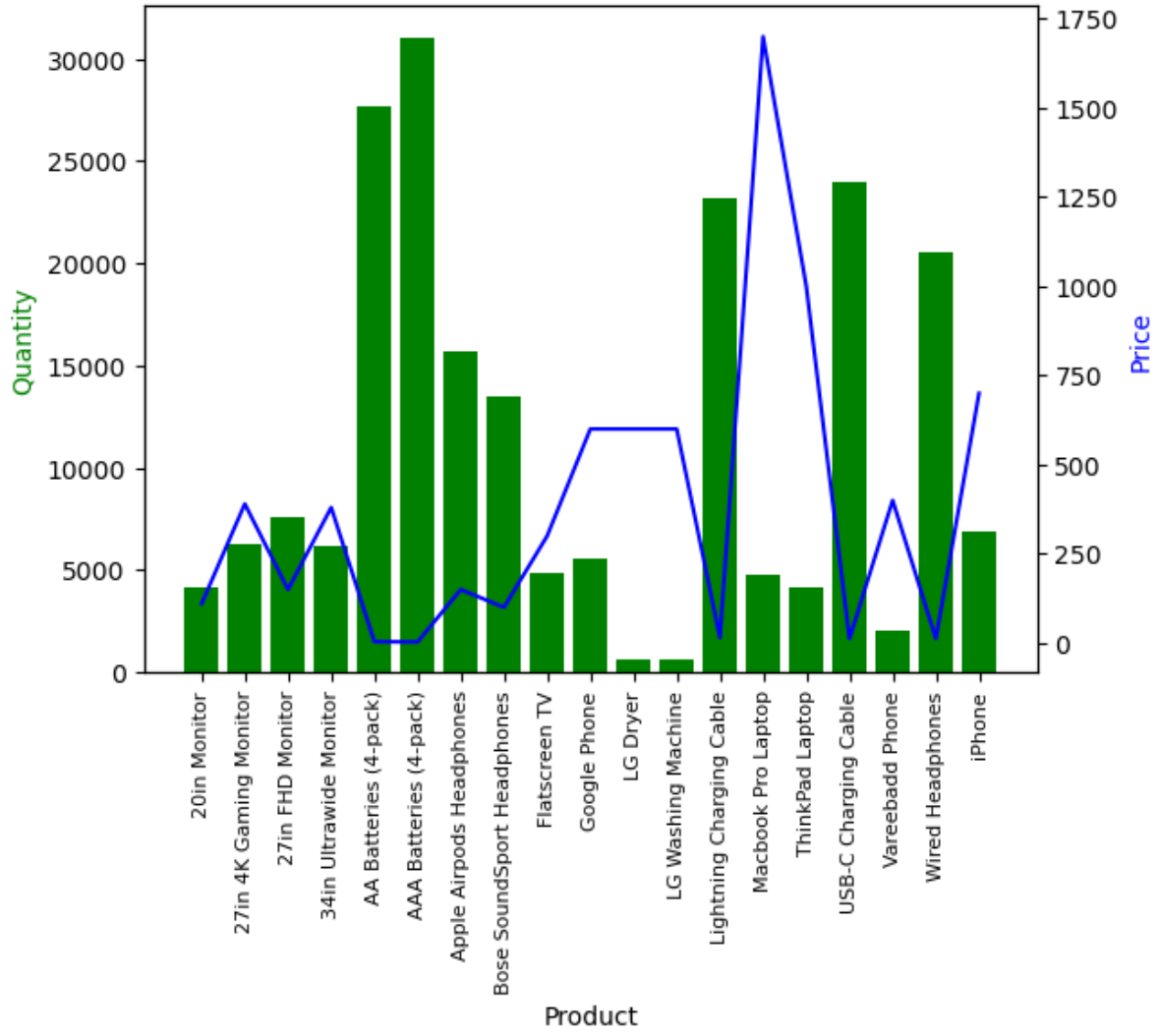
In [44]:

```
product_group = all_data.groupby("Product")
quantity_ordered = product_group.sum()["Quantity Ordered"]

products = [product for product, df in product_group]

plt.bar(products, quantity_ordered)
plt.xticks(products, rotation="vertical", size = 8)
plt.xlabel("Product")
plt.ylabel("Quantity")
plt.show()
```



In [52]:
```
prices = all_data.groupby("Product").mean()["Price Each"]

fig, ax1 = plt.subplots()

ax2 = ax1.twinx()
ax1.bar(products, quantity_ordered, color = "g")
ax2.plot(products, prices, 'b-')

ax1.set_xlabel('Product')
ax1.set_ylabel('Quantity', color='g')
ax2.set_ylabel('Price', color='b')
ax1.set_xticklabels(products, rotation = "vertical", size = 8)

plt.show()
```

```
/var/folders/d1/qkfg3wgx4b1f8kmsbs6l6p6m0000gn/T/ipykernel_56388/1644720022.
py:12: UserWarning: FixedFormatter should only be used together with FixedLo
cator
  ax1.set_xticklabels(products, rotation = "vertical", size = 8)
```


Quantity
Price
Product
0
5000
10000
15000
20000
25000
30000
0
250
500
750
1000
1250
1500
1750
20in Monitor
27in 4K Gaming Monitor
27in FHD Monitor
34in Ultrawide Monitor
AA Batteries (4-pack)
AAA Batteries (4-pack)
Apple Airpods Headphones
Bose SoundSport Headphones
Flatscreen TV
Google Phone
LG Dryer
LG Washing Machine
Lightning Charging Cable
Macbook Pro Laptop
ThinkPad Laptop
USB-C Charging Cable
Vareebadd Phone
Wired Headphones
iPhone