```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np

sales_data
=pd.read_csv("https://raw.githubusercontent.com/KeithGalli/Pandas-
Data-Science-Tasks/master/SalesAnalysis/Output/all_data.csv")

sales_data.head
```

```
<bound method NDFrame.head of        Order ID
Product Quantity Ordered Price Each  \
0        176558        USB-C Charging Cable                2
11.95
1           NaN                         NaN              NaN
NaN
2        176559  Bose SoundSport Headphones                1
99.99
3        176560                Google Phone                1
600
4        176560            Wired Headphones                1
11.99
...         ...                         ...              ...         ..
.
186845   259353      AAA Batteries (4-pack)                3
2.99
186846   259354                      iPhone                1
700
186847   259355                      iPhone                1
700
186848   259356      34in Ultrawide Monitor                1
379.99
186849   259357        USB-C Charging Cable                1
11.95

            Order Date                         Purchase Address
0       04/19/19 08:46             917 1st St, Dallas, TX 75001
1                  NaN                                      NaN
2       04/07/19 22:30        682 Chestnut St, Boston, MA 02215
3       04/12/19 14:38     669 Spruce St, Los Angeles, CA 90001
4       04/12/19 14:38     669 Spruce St, Los Angeles, CA 90001
...                ...                                      ...
186845  09/17/19 20:56   840 Highland St, Los Angeles, CA 90001
186846  09/01/19 16:00  216 Dogwood St, San Francisco, CA 94016
186847  09/23/19 07:39     220 12th St, San Francisco, CA 94016
186848  09/19/19 17:30   511 Forest St, San Francisco, CA 94016
186849  09/30/19 00:18   250 Meadow St, San Francisco, CA 94016

[186850 rows x 6 columns]>
```

Let's clean the data

remove rows of nan

```
nan_sales_data = sales_data[sales_data.isna().all(axis=1)]
nan_sales_data.head

sales_data = sales_data.dropna(how="all")
sales_data.head
```

```
<bound method NDFrame.head of        Order ID
Product Quantity Ordered Price Each  \
0        176558        USB-C Charging Cable                2
11.95
2        176559  Bose SoundSport Headphones                1
99.99
3        176560                Google Phone                1
600
4        176560            Wired Headphones                1
11.99
5        176561            Wired Headphones                1
11.99
...         ...                         ...              ...          ..
.
186845   259353      AAA Batteries (4-pack)                3
2.99
186846   259354                      iPhone                1
700
186847   259355                      iPhone                1
700
186848   259356      34in Ultrawide Monitor                1
379.99
186849   259357        USB-C Charging Cable                1
11.95

            Order Date                         Purchase Address
0       04/19/19 08:46             917 1st St, Dallas, TX 75001
2       04/07/19 22:30        682 Chestnut St, Boston, MA 02215
3       04/12/19 14:38     669 Spruce St, Los Angeles, CA 90001
4       04/12/19 14:38     669 Spruce St, Los Angeles, CA 90001
5       04/30/19 09:27        333 8th St, Los Angeles, CA 90001
...                ...                                      ...
186845  09/17/19 20:56   840 Highland St, Los Angeles, CA 90001
186846  09/01/19 16:00  216 Dogwood St, San Francisco, CA 94016
186847  09/23/19 07:39     220 12th St, San Francisco, CA 94016
186848  09/19/19 17:30   511 Forest St, San Francisco, CA 94016
186849  09/30/19 00:18   250 Meadow St, San Francisco, CA 94016

[186305 rows x 6 columns]>
```

finding 'Or" and deleting it

```
sales_data = sales_data[sales_data['Order Date'].str[0:2] != "Or"]
sales_data.head()
```

```
  Order ID                     Product Quantity Ordered Price Each  \
0   176558        USB-C Charging Cable                2      11.95
2   176559  Bose SoundSport Headphones                1      99.99
3   176560                Google Phone                1        600
4   176560            Wired Headphones                1      11.99
5   176561            Wired Headphones                1      11.99

       Order Date                      Purchase Address
0  04/19/19 08:46          917 1st St, Dallas, TX 75001
2  04/07/19 22:30     682 Chestnut St, Boston, MA 02215
3  04/12/19 14:38  669 Spruce St, Los Angeles, CA 90001
4  04/12/19 14:38  669 Spruce St, Los Angeles, CA 90001
5  04/30/19 09:27     333 8th St, Los Angeles, CA 90001
```

ADDING MONTH COLUMN

```
sales_data["Month"] = sales_data["Order Date"].str[0:2]
sales_data["Month"] = sales_data["Month"].astype('int32')
sales_data.head(20)
```

```
C:\Users\HP\AppData\Local\Temp\ipykernel_15852\550544151.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
  sales_data["Month"] = sales_data["Order Date"].str[0:2]
C:\Users\HP\AppData\Local\Temp\ipykernel_15852\550544151.py:2:
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
  sales_data["Month"] = sales_data["Month"].astype('int32')

   Order ID                     Product Quantity Ordered Price Each  \
0    176558        USB-C Charging Cable                2      11.95
2    176559  Bose SoundSport Headphones                1      99.99
3    176560                Google Phone                1        600
4    176560            Wired Headphones                1      11.99
5    176561            Wired Headphones                1      11.99
```

```
6    176562        USB-C Charging Cable                1      11.95
7    176563  Bose SoundSport Headphones                1      99.99
8    176564        USB-C Charging Cable                1      11.95
9    176565          Macbook Pro Laptop                1       1700
10   176566           Wired Headphones                 1      11.99
11   176567               Google Phone                 1        600
12   176568    Lightning Charging Cable                1      14.95
13   176569       27in 4K Gaming Monitor               1     389.99
14   176570        AA Batteries (4-pack)               1       3.84
15   176571    Lightning Charging Cable                1      14.95
16   176572    Apple Airpods Headphones                1        150
17   176573        USB-C Charging Cable                1      11.95
18   176574               Google Phone                 1        600
19   176574        USB-C Charging Cable                1      11.95
20   176575       AAA Batteries (4-pack)               1       2.99

        Order Date                          Purchase Address  Month
0   04/19/19 08:46              917 1st St, Dallas, TX 75001      4
2   04/07/19 22:30         682 Chestnut St, Boston, MA 02215      4
3   04/12/19 14:38      669 Spruce St, Los Angeles, CA 90001      4
4   04/12/19 14:38      669 Spruce St, Los Angeles, CA 90001      4
5   04/30/19 09:27         333 8th St, Los Angeles, CA 90001      4
6   04/29/19 13:03    381 Wilson St, San Francisco, CA 94016      4
7   04/02/19 07:46          668 Center St, Seattle, WA 98101      4
8   04/12/19 10:58           790 Ridge St, Atlanta, GA 30301      4
9   04/24/19 10:38    915 Willow St, San Francisco, CA 94016      4
10  04/08/19 14:05              83 7th St, Boston, MA 02215      4
11  04/18/19 17:18         444 7th St, Los Angeles, CA 90001      4
12  04/15/19 12:18             438 Elm St, Seattle, WA 98101      4
13  04/16/19 19:23             657 Hill St, Dallas, TX 75001      4
14  04/22/19 15:09             186 12th St, Dallas, TX 75001      4
15  04/19/19 14:29         253 Johnson St, Atlanta, GA 30301      4
16  04/04/19 20:30   149 Dogwood St, New York City, NY 10001      4
17  04/27/19 18:41  214 Chestnut St, San Francisco, CA 94016      4
18  04/03/19 19:42         20 Hill St, Los Angeles, CA 90001      4
19  04/03/19 19:42         20 Hill St, Los Angeles, CA 90001      4
20  04/27/19 00:30      433 Hill St, New York City, NY 10001      4
```

convert columns tto the correcct type

```
sales_data["Quantity Ordered"] = pd.to_numeric (sales_data["Quantity Ordered"])# make int
sales_data["Price Each"]= pd.to_numeric(sales_data["Price Each"]) # make float
sales_data.head()
```

```
C:\Users\HP\AppData\Local\Temp\ipykernel_15852\4161471391.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
  sales_data["Quantity Ordered"] = pd.to_numeric (sales_data["Quantity
Ordered"])# make int
C:\Users\HP\AppData\Local\Temp\ipykernel_15852\4161471391.py:2:
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
  sales_data["Price Each"]= pd.to_numeric(sales_data["Price Each"]) #
make float

  Order ID                     Product  Quantity Ordered  Price
Each  \
0   176558        USB-C Charging Cable                 2       11.95

2   176559  Bose SoundSport Headphones                 1       99.99

3   176560                Google Phone                 1      600.00

4   176560            Wired Headphones                 1       11.99

5   176561            Wired Headphones                 1       11.99


       Order Date                      Purchase Address  Month
0  04/19/19 08:46          917 1st St, Dallas, TX 75001      4
2  04/07/19 22:30     682 Chestnut St, Boston, MA 02215      4
3  04/12/19 14:38  669 Spruce St, Los Angeles, CA 90001      4
4  04/12/19 14:38  669 Spruce St, Los Angeles, CA 90001      4
5  04/30/19 09:27     333 8th St, Los Angeles, CA 90001      4
```

What was the best month for sales? how much was earned that month?

add a sales column

```
sales_data['Sales'] = sales_data['Quantity Ordered'] *
sales_data['Price Each']
sales_data.head

C:\Users\HP\AppData\Local\Temp\ipykernel_15852\449679353.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
  sales_data['Sales'] = sales_data['Quantity Ordered'] *
sales_data['Price Each']

<bound method NDFrame.head of        Order ID
Product  Quantity Ordered  Price Each  \
0        176558        USB-C Charging Cable                2
11.95
2        176559  Bose SoundSport Headphones                1
99.99
3        176560                Google Phone                1
600.00
4        176560            Wired Headphones                1
11.99
5        176561            Wired Headphones                1
11.99
...         ...                         ...              ...
...
186845   259353      AAA Batteries (4-pack)                3
2.99
186846   259354                      iPhone                1
700.00
186847   259355                      iPhone                1
700.00
186848   259356      34in Ultrawide Monitor                1
379.99
186849   259357        USB-C Charging Cable                1
11.95

            Order Date                         Purchase Address  Month
Sales
0       04/19/19 08:46             917 1st St, Dallas, TX 75001      4
23.90
2       04/07/19 22:30        682 Chestnut St, Boston, MA 02215      4
99.99
3       04/12/19 14:38     669 Spruce St, Los Angeles, CA 90001      4
600.00
4       04/12/19 14:38     669 Spruce St, Los Angeles, CA 90001      4
11.99
5       04/30/19 09:27        333 8th St, Los Angeles, CA 90001      4
11.99
...                ...                                      ...    ...
...
186845  09/17/19 20:56   840 Highland St, Los Angeles, CA 90001      9
8.97
186846  09/01/19 16:00  216 Dogwood St, San Francisco, CA 94016      9
700.00
```

```
186847  09/23/19 07:39     220 12th St, San Francisco, CA 94016      9
700.00
186848  09/19/19 17:30   511 Forest St, San Francisco, CA 94016      9
379.99
186849  09/30/19 00:18   250 Meadow St, San Francisco, CA 94016      9
11.95

[185950 rows x 8 columns]>
```

Add a city Column we will do this with the .apply function

```
sales_data["city"]= sales_data["Purchase Address"].apply (lambda
x:x.split (",")[1])
sales_data.head()

#this can also be done
# def get_city(address):
#        return address.split(",")[1]
#
#sales_data ["city"] = sales_data ["purchase address"].apply(lambda x:
get_city(x))
#lambda allows us to grab cells contents

C:\Users\HP\AppData\Local\Temp\ipykernel_15852\126474782.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
  sales_data["city"]= sales_data["Purchase Address"].apply (lambda
x:x.split (",")[1])

  Order ID                    Product  Quantity Ordered  Price
Each  \
0   176558        USB-C Charging Cable                 2       11.95

2   176559  Bose SoundSport Headphones                 1       99.99

3   176560                Google Phone                 1      600.00

4   176560            Wired Headphones                 1       11.99

5   176561            Wired Headphones                 1       11.99


       Order Date                      Purchase Address  Month   Sales
\
0  04/19/19 08:46          917 1st St, Dallas, TX 75001      4   23.90
```

```
2  04/07/19 22:30     682 Chestnut St, Boston, MA 02215      4   99.99

3  04/12/19 14:38  669 Spruce St, Los Angeles, CA 90001      4  600.00

4  04/12/19 14:38  669 Spruce St, Los Angeles, CA 90001      4   11.99

5  04/30/19 09:27     333 8th St, Los Angeles, CA 90001      4   11.99


           city
0        Dallas
2        Boston
3   Los Angeles
4   Los Angeles
5   Los Angeles
```

```
#we need to grab the state alongside the state code becasue some 
cities might have the same name across the world
def get_city(address):
       return address.split(",")[1]
def get_state(address):
       return address.split(",")[2].split(" ")[1]

sales_data ["City"] = sales_data ["Purchase Address"].apply(lambda x: 
get_city(x)+ ' ' + get_state(x))
sales_data
#lambda allows us to grab cells contents
```

```
C:\Users\HP\AppData\Local\Temp\ipykernel_15852\2956218034.py:7: 
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
  sales_data ["City"] = sales_data ["Purchase Address"].apply(lambda 
x: get_city(x)+ ' ' + get_state(x))

       Order ID                     Product  Quantity Ordered  Price 
Each  \
0        176558        USB-C Charging Cable                 2 
11.95   
2        176559  Bose SoundSport Headphones                 1 
99.99   
3        176560                Google Phone                 1 
600.00   
4        176560            Wired Headphones                 1 
11.99   
5        176561            Wired Headphones                 1 
```

```
11.99
...          ...                          ...                ...
...
186845   259353      AAA Batteries (4-pack)                  3
2.99
186846   259354                       iPhone                 1
700.00
186847   259355                       iPhone                 1
700.00
186848   259356      34in Ultrawide Monitor                  1
379.99
186849   259357        USB-C Charging Cable                  1
11.95

            Order Date                         Purchase Address  Month
\
0       04/19/19 08:46             917 1st St, Dallas, TX 75001      4

2       04/07/19 22:30        682 Chestnut St, Boston, MA 02215      4

3       04/12/19 14:38     669 Spruce St, Los Angeles, CA 90001      4

4       04/12/19 14:38     669 Spruce St, Los Angeles, CA 90001      4

5       04/30/19 09:27        333 8th St, Los Angeles, CA 90001      4

...                ...                                      ...    ...

186845  09/17/19 20:56   840 Highland St, Los Angeles, CA 90001      9

186846  09/01/19 16:00  216 Dogwood St, San Francisco, CA 94016      9

186847  09/23/19 07:39     220 12th St, San Francisco, CA 94016      9

186848  09/19/19 17:30   511 Forest St, San Francisco, CA 94016      9

186849  09/30/19 00:18   250 Meadow St, San Francisco, CA 94016      9


         Sales            city              City
0        23.90          Dallas         Dallas TX
2        99.99          Boston         Boston MA
3       600.00     Los Angeles    Los Angeles CA
4        11.99     Los Angeles    Los Angeles CA
5        11.99     Los Angeles    Los Angeles CA
...        ...             ...               ...
186845    8.97     Los Angeles    Los Angeles CA
186846  700.00   San Francisco  San Francisco CA
186847  700.00   San Francisco  San Francisco CA
186848  379.99   San Francisco  San Francisco CA
```

```
186849   11.95   San Francisco   San Francisco CA

[185950 rows x 10 columns]
```

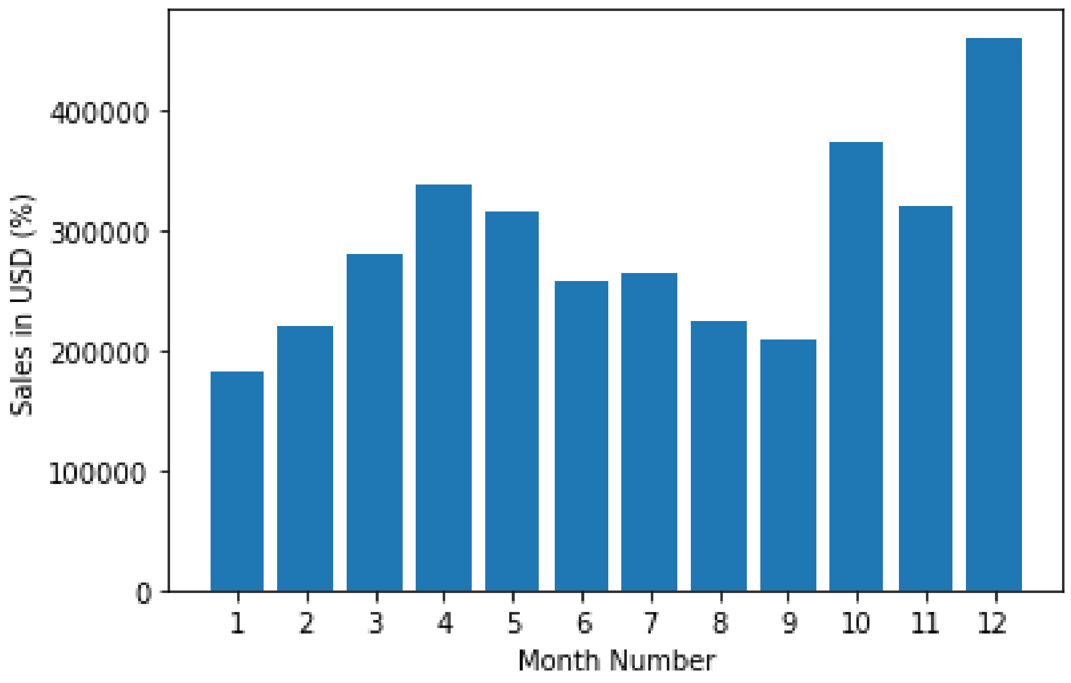
Best month for sales and how much was made in that month?

```
Total_sales = sales_data.groupby ("Month").sum()
print(Total_sales)
```

```
       Quantity Ordered  Price Each       Sales
Month
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
```

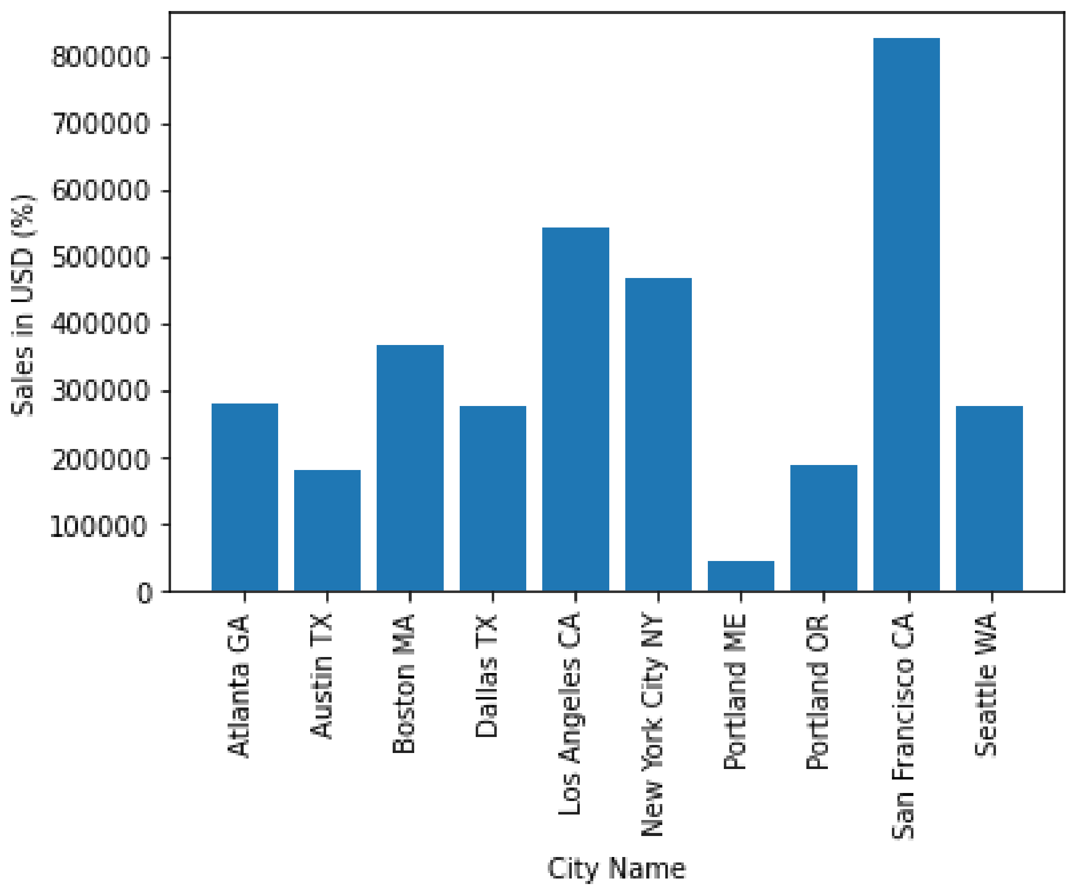
```
months = range(1,13) # Define the range of months to be plotted (1 to 
12)
plt.bar(months, Total_sales["Sales"]/10) # Plot a bar chart for the 
Sales column of Total_sales dataframe with x-axis as months
plt.xticks(months) # Set the ticks on x-axis to be the range of months 
(1 to 12)
plt.ylabel('Sales in USD (%)') # Define the label for y-axis
plt.xlabel('Month Number') # Define the label for x-axis
```

```
Text(0.5, 0, 'Month Number')
```



What US city had the highest number of sales

```
Total_sales = sales_data.groupby ("City").sum()
Total_sales

                   Quantity Ordered  Price Each   Month       Sales
City
 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

cities = [city for city, df in sales_data.groupby("City")]
plt.bar(cities, Total_sales["Sales"]/10) # Plot a bar chart for the
Sales column of Total_sales dataframe with x-axis as months
plt.xticks(cities, rotation = "vertical", size = 10)
plt.ylabel('Sales in USD (%)') # Define the label for y-axis
plt.xlabel('City Name')

Text(0.5, 0, 'City Name')
```



```
sales_data["Order Date"] = pd.to_datetime(sales_data["Order Date"])
sales_data.head()
```

```
C:\Users\HP\AppData\Local\Temp\ipykernel_15852\1463456202.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
  sales_data["Order Date"] = pd.to_datetime(sales_data["Order Date"])

  Order ID                     Product  Quantity Ordered  Price
Each  \
0   176558        USB-C Charging Cable                 2       11.95

2   176559  Bose SoundSport Headphones                 1       99.99

3   176560                Google Phone                 1      600.00

4   176560            Wired Headphones                 1       11.99
```

```
5   176561           Wired Headphones                 1       11.99


           Order Date                      Purchase Address  Month
Sales  \
0 2019-04-19 08:46:00         917 1st St, Dallas, TX 75001      4
23.90
2 2019-04-07 22:30:00     682 Chestnut St, Boston, MA 02215      4
99.99
3 2019-04-12 14:38:00  669 Spruce St, Los Angeles, CA 90001      4
600.00
4 2019-04-12 14:38:00  669 Spruce St, Los Angeles, CA 90001      4
11.99
5 2019-04-30 09:27:00     333 8th St, Los Angeles, CA 90001      4
11.99

           city             City
0        Dallas        Dallas TX
2        Boston        Boston MA
3   Los Angeles   Los Angeles CA
4   Los Angeles   Los Angeles CA
5   Los Angeles   Los Angeles CA
```

```
sales_data ["Hour"] = sales_data ["Order Date"].dt.hour
sales_data["Minute"] = sales_data["Order Date"].dt.minute
sales_data.head()
```

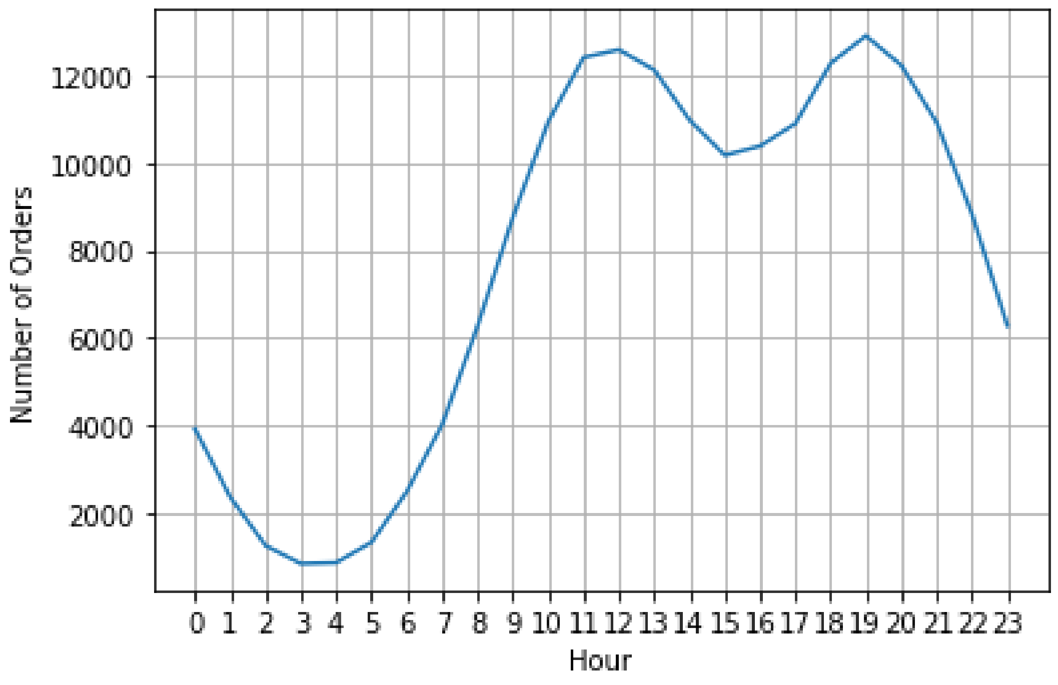
```
C:\Users\HP\AppData\Local\Temp\ipykernel_15852\2232285383.py:1:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy
  sales_data ["Hour"] = sales_data ["Order Date"].dt.hour
C:\Users\HP\AppData\Local\Temp\ipykernel_15852\2232285383.py:2:
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
  sales_data["Minute"] = sales_data["Order Date"].dt.minute

  Order ID                     Product  Quantity Ordered  Price
Each  \
0   176558        USB-C Charging Cable                 2       11.95

2   176559  Bose SoundSport Headphones                 1       99.99
```

```
3   176560                Google Phone                 1      600.00

4   176560            Wired Headphones                 1       11.99

5   176561            Wired Headphones                 1       11.99


          Order Date                      Purchase Address  Month
Sales  \
0 2019-04-19 08:46:00          917 1st St, Dallas, TX 75001      4
23.90
2 2019-04-07 22:30:00     682 Chestnut St, Boston, MA 02215      4
99.99
3 2019-04-12 14:38:00  669 Spruce St, Los Angeles, CA 90001      4
600.00
4 2019-04-12 14:38:00  669 Spruce St, Los Angeles, CA 90001      4
11.99
5 2019-04-30 09:27:00     333 8th St, Los Angeles, CA 90001      4
11.99

           city              City  Hour  Minute
0        Dallas         Dallas TX     8      46
2        Boston         Boston MA    22      30
3   Los Angeles    Los Angeles CA    14      38
4   Los Angeles    Los Angeles CA    14      38
5   Los Angeles    Los Angeles CA     9      27
```

what time should we display adverts so as to maximise likelihood of customers buying products?

```
sales_count_by_hour =
sales_data.groupby(["Hour"]).count().reset_index()
hours = [hour for hour, df in sales_data.groupby("Hour")]
plt.plot(sales_count_by_hour["Hour"], sales_count_by_hour["Sales"])
plt.xticks(hours)
plt.xlabel("Hour")
plt.ylabel("Number of Orders")
plt.grid()
print(sales_data.groupby("Hour").count())

      Order ID  Product  Quantity Ordered  Price Each  Order Date  \
Hour
0         3910     3910              3910        3910        3910
1         2350     2350              2350        2350        2350
2         1243     1243              1243        1243        1243
3          831      831               831         831         831
4          854      854               854         854         854
5         1321     1321              1321        1321        1321
6         2482     2482              2482        2482        2482
7         4011     4011              4011        4011        4011
```

```
8         6256     6256              6256        6256        6256
9         8748     8748              8748        8748        8748
10       10944    10944             10944       10944       10944
11       12411    12411             12411       12411       12411
12       12587    12587             12587       12587       12587
13       12129    12129             12129       12129       12129
14       10984    10984             10984       10984       10984
15       10175    10175             10175       10175       10175
16       10384    10384             10384       10384       10384
17       10899    10899             10899       10899       10899
18       12280    12280             12280       12280       12280
19       12905    12905             12905       12905       12905
20       12228    12228             12228       12228       12228
21       10921    10921             10921       10921       10921
22        8822     8822              8822        8822        8822
23        6275     6275              6275        6275        6275

      Purchase Address  Month  Sales   city   City  Minute
Hour
0                 3910   3910   3910   3910   3910    3910
1                 2350   2350   2350   2350   2350    2350
2                 1243   1243   1243   1243   1243    1243
3                  831    831    831    831    831     831
4                  854    854    854    854    854     854
5                 1321   1321   1321   1321   1321    1321
6                 2482   2482   2482   2482   2482    2482
7                 4011   4011   4011   4011   4011    4011
8                 6256   6256   6256   6256   6256    6256
9                 8748   8748   8748   8748   8748    8748
10               10944  10944  10944  10944  10944   10944
11               12411  12411  12411  12411  12411   12411
12               12587  12587  12587  12587  12587   12587
13               12129  12129  12129  12129  12129   12129
14               10984  10984  10984  10984  10984   10984
15               10175  10175  10175  10175  10175   10175
16               10384  10384  10384  10384  10384   10384
17               10899  10899  10899  10899  10899   10899
18               12280  12280  12280  12280  12280   12280
19               12905  12905  12905  12905  12905   12905
20               12228  12228  12228  12228  12228   12228
21               10921  10921  10921  10921  10921   10921
22                8822   8822   8822   8822   8822    8822
23                6275   6275   6275   6275   6275    6275
```



```
#This will create a new DataFrame with a numeric index that can be
used with the plt.plot() function.
```

what products were often sold together?

```
df= sales_data[sales_data["Order ID"].duplicated(keep=False)]
df["Grouped"] = df.groupby("Order ID")["Product"].transform(lambda x:
",".join(x))
df = df[["Order ID", 'Grouped']].drop_duplicates()
print(df)

       Order ID                                           Grouped
3        176560                      Google Phone,Wired Headphones
18       176574                  Google Phone,USB-C Charging Cable
30       176585  Bose SoundSport Headphones,Bose SoundSport Hea...
32       176586                AAA Batteries (4-pack),Google Phone
119      176672      Lightning Charging Cable,USB-C Charging Cable
...         ...                                               ...
186781   259296  Apple Airpods Headphones,Apple Airpods Headphones
186783   259297  iPhone,Lightning Charging Cable,Lightning Char...
186791   259303       34in Ultrawide Monitor,AA Batteries (4-pack)
186803   259314           Wired Headphones,AAA Batteries (4-pack)
186841   259350                  Google Phone,USB-C Charging Cable

[7136 rows x 2 columns]
```

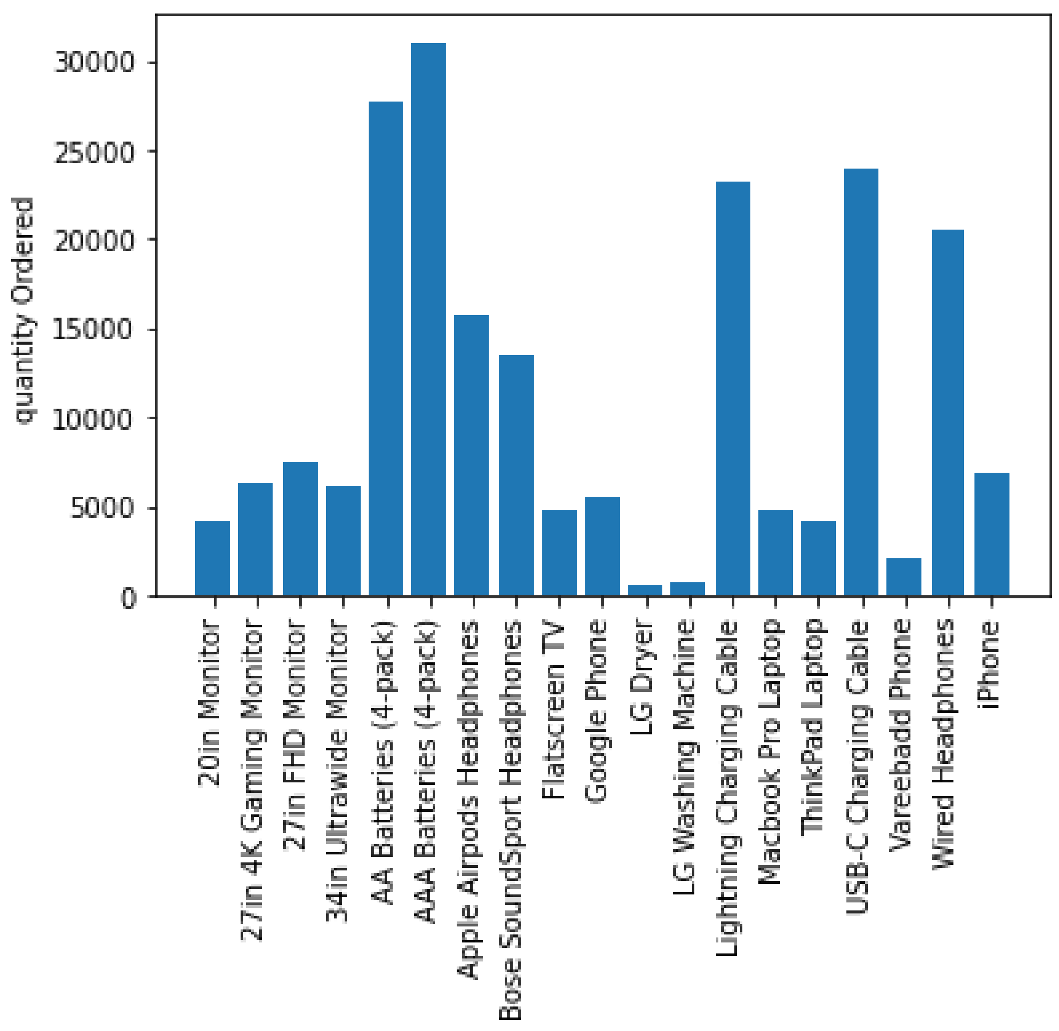
```
C:\Users\HP\AppData\Local\Temp\ipykernel_15852\3692291283.py:2:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

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))
```

```
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, 1)))

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

```
('iPhone,Lightning Charging Cable',) 882
('Google Phone,USB-C Charging Cable',) 856
('iPhone,Wired Headphones',) 361
('Vareebadd Phone,USB-C Charging Cable',) 312
('Google Phone,Wired Headphones',) 303
('iPhone,Apple Airpods Headphones',) 286
('Google Phone,Bose SoundSport Headphones',) 161
('Vareebadd Phone,Wired Headphones',) 104
('Google Phone,USB-C Charging Cable,Wired Headphones',) 77
('Vareebadd Phone,Bose SoundSport Headphones',) 60
```

what products sold the most and why?

```
grouped_products = sales_data.groupby("Product")
quantity_ordered = grouped_products.sum()["Quantity Ordered"]

product = [products for products, df in grouped_products]
plt.bar(product, quantity_ordered)
plt.ylabel ('quantity Ordered')fi
plt.xticks (product, rotation = "vertical", size = 10)
plt.show()
```



TO know the prices of each of this product so as to determine why some are sold more than the rest, we first find the average pprice of each product.

```
prices = sales_data.groupby("Product").mean()["Price Each"]
prices
```

```
Product
20in Monitor                  109.99
27in 4K Gaming Monitor        389.99
27in FHD Monitor              149.99
34in Ultrawide Monitor        379.99
AA Batteries (4-pack)           3.84
AAA Batteries (4-pack)          2.99
Apple Airpods Headphones      150.00
Bose SoundSport Headphones     99.99
Flatscreen TV                 300.00
Google Phone                  600.00
LG Dryer                      600.00
LG Washing Machine            600.00
```

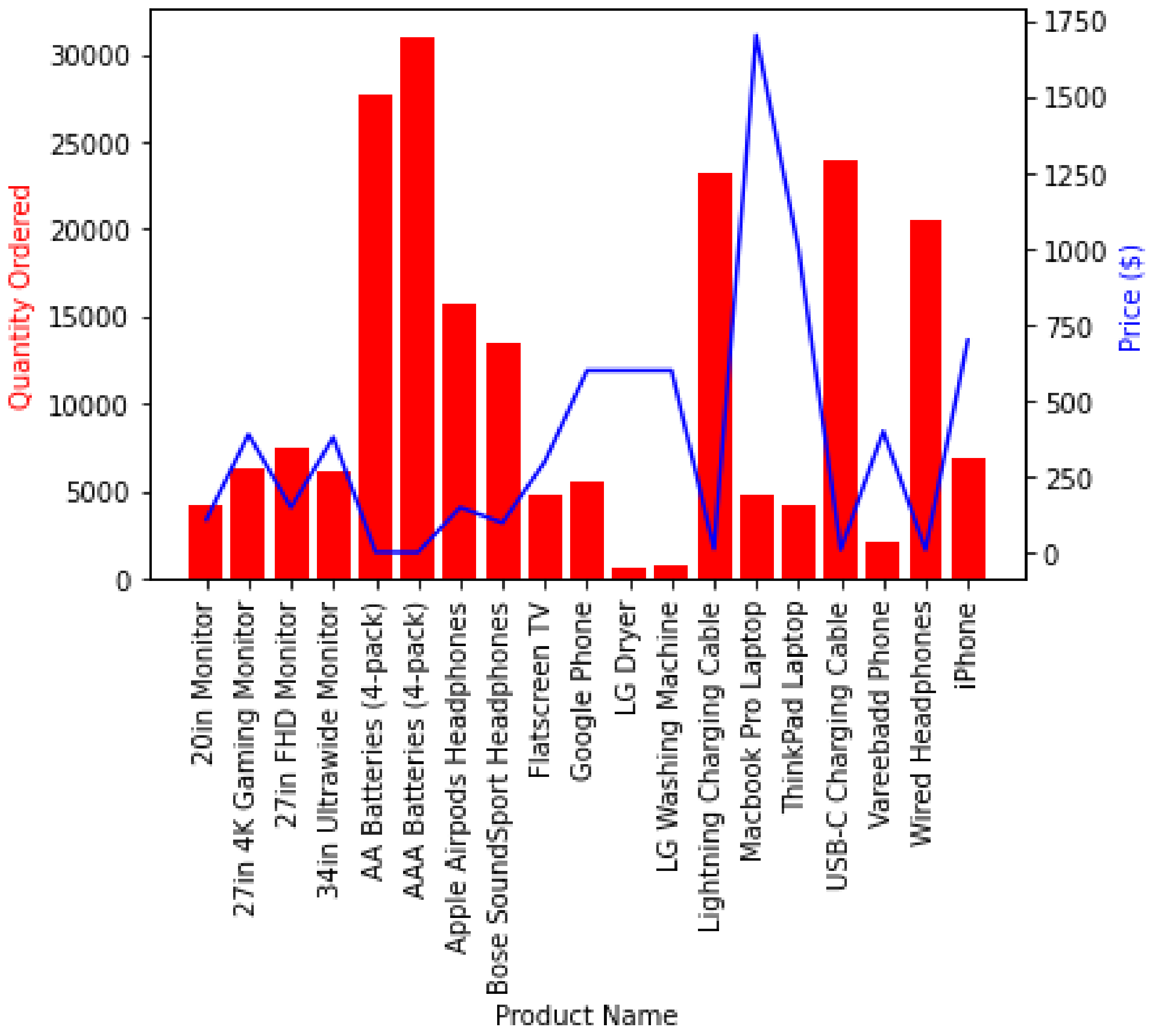
```
Lightning Charging Cable        14.95
Macbook Pro Laptop            1700.00
ThinkPad Laptop                999.99
USB-C Charging Cable            11.95
Vareebadd Phone                400.00
Wired Headphones                11.99
iPhone                         700.00
Name: Price Each, dtype: float64
```

we then add this average prices as a subplot to the above plot

```
fig, ax1 = plt.subplots()

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

ax1.set_xlabel ("Product Name")
ax1.set_ylabel ("Quantity Ordered", color = "r")
ax2.set_ylabel("Price ($)", color = "b")
ax1.set_xticklabels(product, rotation = "vertical", size = 10)

plt.show
```

```
C:\Users\HP\AppData\Local\Temp\ipykernel_15852\1496643603.py:10:
UserWarning: FixedFormatter should only be used together with
FixedLocator
  ax1.set_xticklabels(product, rotation = "vertical", size = 10)
```

```
<function matplotlib.pyplot.show(close=None, block=None)>
```


Quantity Ordered
Price ($)
Product Name
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