Name: Muskan Dosi

Div : F4

Roll no. : 676

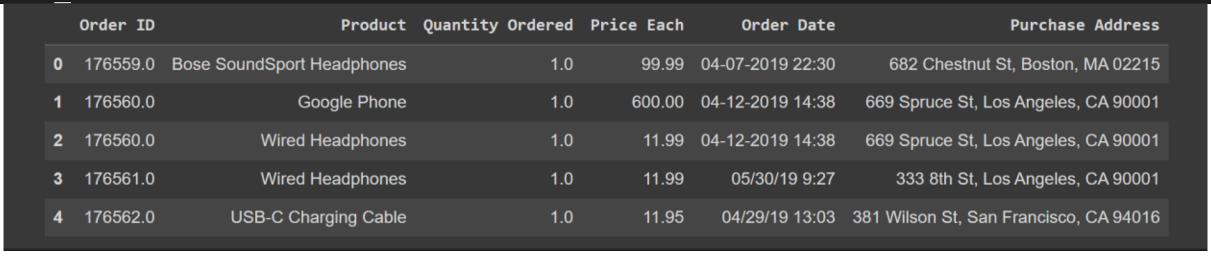
PRN : 202201040128

Code:-

```
import numpy as np
import pandas as pd

all_data=pd.read_csv("/content/drive/MyDrive/Colab
Notebooks/1686715083343_all_data.csv")

all_data.head()
```



	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address
0	176559.0	Bose SoundSport Headphones	1.0	99.99	04-07-2019 22:30	682 Chestnut St, Boston, MA 02215
1	176560.0	Google Phone	1.0	600.00	04-12-2019 14:38	669 Spruce St, Los Angeles, CA 90001
2	176560.0	Wired Headphones	1.0	11.99	04-12-2019 14:38	669 Spruce St, Los Angeles, CA 90001
3	176561.0	Wired Headphones	1.0	11.99	05/30/19 9:27	333 8th St, Los Angeles, CA 90001
4	176562.0	USB-C Charging Cable	1.0	11.95	04/29/19 13:03	381 Wilson St, San Francisco, CA 94016

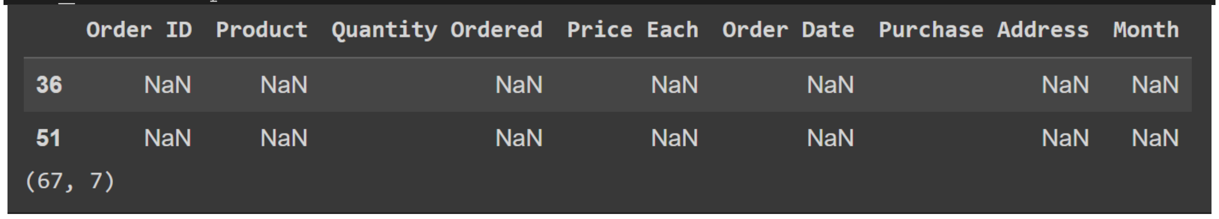
Drop rows of NAN

```
#Find NAN
nan_df = all_data[all_data.isna().any(axis=1)]
display(nan_df.head())

all_data.shape

all_data = all_data.dropna(how='all')
all_data.head()

all_data.shape
```

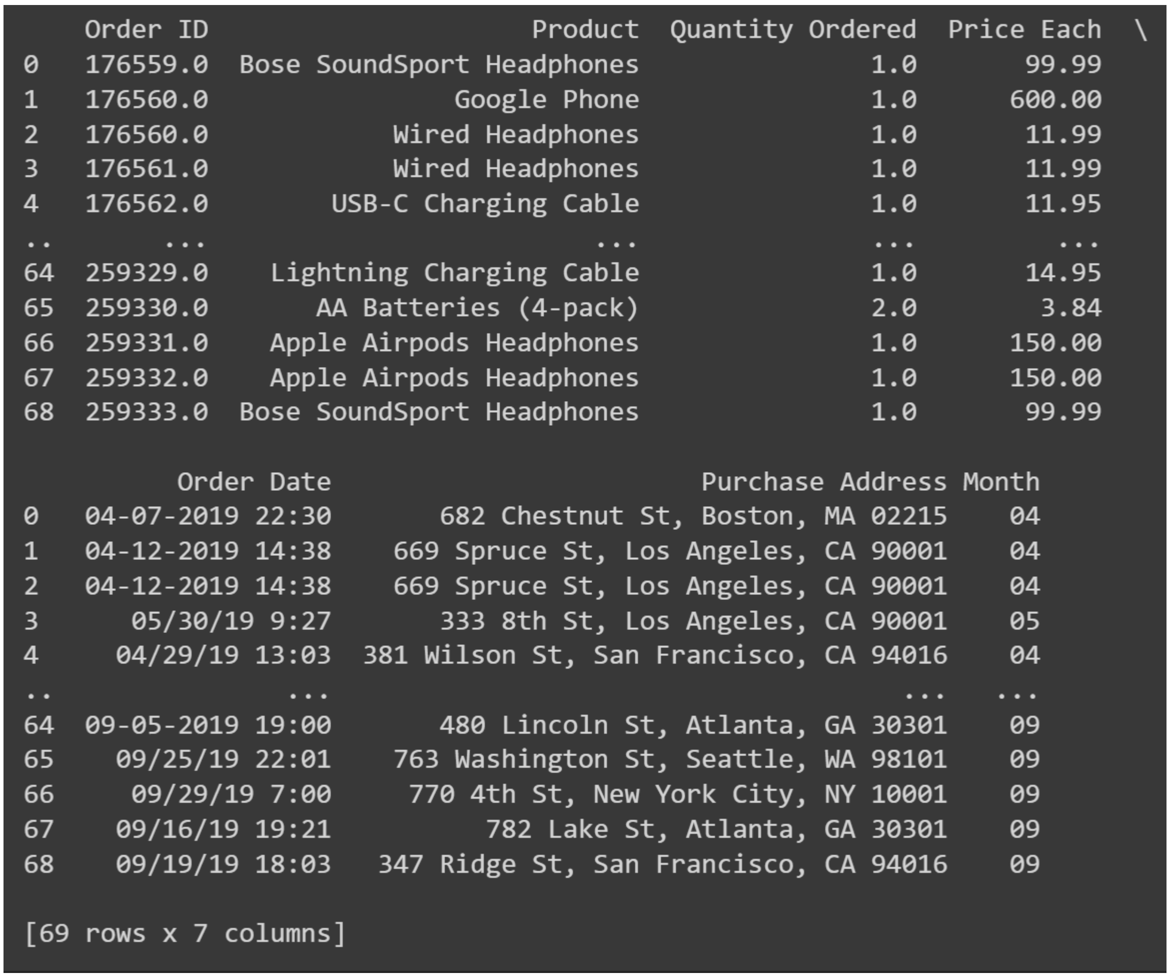


	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month
36	NaN	NaN	NaN	NaN	NaN	NaN	NaN
51	NaN	NaN	NaN	NaN	NaN	NaN	NaN

```
(67, 7)
```

Get rid of text in order date column

```
all_data = all_data[all_data['Order Date'].str[0:2]!='Or']
print(all_data)
```



```
    Order ID                     Product  Quantity Ordered  Price Each  \
0   176559.0  Bose SoundSport Headphones               1.0       99.99
1   176560.0                Google Phone               1.0      600.00
2   176560.0            Wired Headphones               1.0       11.99
3   176561.0            Wired Headphones               1.0       11.99
4   176562.0        USB-C Charging Cable               1.0       11.95
..       ...                         ...               ...         ...
64  259329.0      Lightning Charging Cable             1.0       14.95
65  259330.0       AA Batteries (4-pack)               2.0        3.84
66  259331.0      Apple Airpods Headphones             1.0      150.00
67  259332.0      Apple Airpods Headphones             1.0      150.00
68  259333.0  Bose SoundSport Headphones               1.0       99.99

          Order Date                        Purchase Address Month
0   04-07-2019 22:30       682 Chestnut St, Boston, MA 02215    04
1   04-12-2019 14:38    669 Spruce St, Los Angeles, CA 90001    04
2   04-12-2019 14:38    669 Spruce St, Los Angeles, CA 90001    04
3      05/30/19 9:27       333 8th St, Los Angeles, CA 90001    05
4     04/29/19 13:03  381 Wilson St, San Francisco, CA 94016    04
..               ...                                     ...   ...
64  09-05-2019 19:00       480 Lincoln St, Atlanta, GA 30301    09
65     09/25/19 22:01    763 Washington St, Seattle, WA 98101    09
66      09/29/19 7:00     770 4th St, New York City, NY 10001    09
67     09/16/19 19:21         782 Lake St, Atlanta, GA 30301    09
68     09/19/19 18:03   347 Ridge St, San Francisco, CA 94016    09

[69 rows x 7 columns]
```

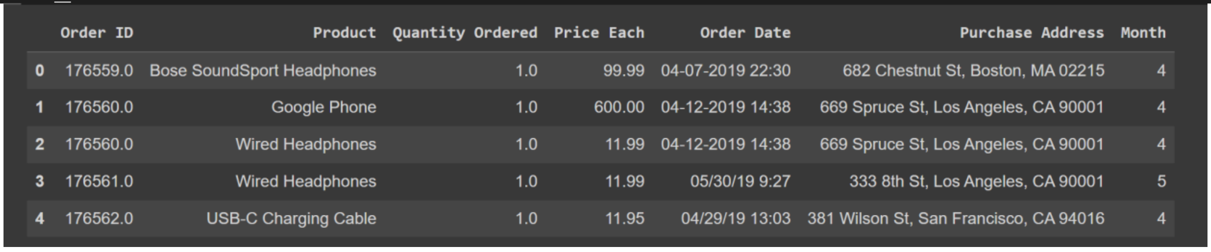
Make columns correct type

```
all_data['Quantity Ordered'] = pd.to_numeric(all_data['Quantity
Ordered'])
all_data['Price Each'] = pd.to_numeric(all_data['Price Each'])
```

Augment data with additional columns

Add month column

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



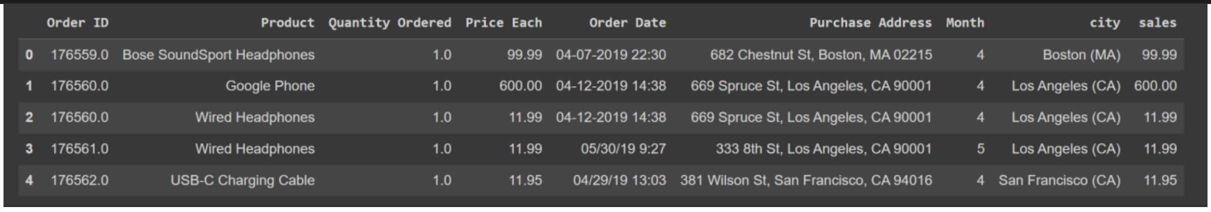
	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month
0	176559.0	Bose SoundSport Headphones	1.0	99.99	04-07-2019 22:30	682 Chestnut St, Boston, MA 02215	4
1	176560.0	Google Phone	1.0	600.00	04-12-2019 14:38	669 Spruce St, Los Angeles, CA 90001	4
2	176560.0	Wired Headphones	1.0	11.99	04-12-2019 14:38	669 Spruce St, Los Angeles, CA 90001	4
3	176561.0	Wired Headphones	1.0	11.99	05/30/19 9:27	333 8th St, Los Angeles, CA 90001	5
4	176562.0	USB-C Charging Cable	1.0	11.95	04/29/19 13:03	381 Wilson St, San Francisco, CA 94016	4

Add city column

```
from pandas.core.ops.methods import add_flex_arithmetic_methods
def get_city(address):
  return address.split(",")[1].strip(" ")
```

```python
def get_state(address):
  return address.split(",")[2].split(" ")[1]

all_data['city'] = all_data["Purchase Address"].apply(lambda
x:f"{get_city(x)}  ({get_state(x)})")
all_data.head()
```

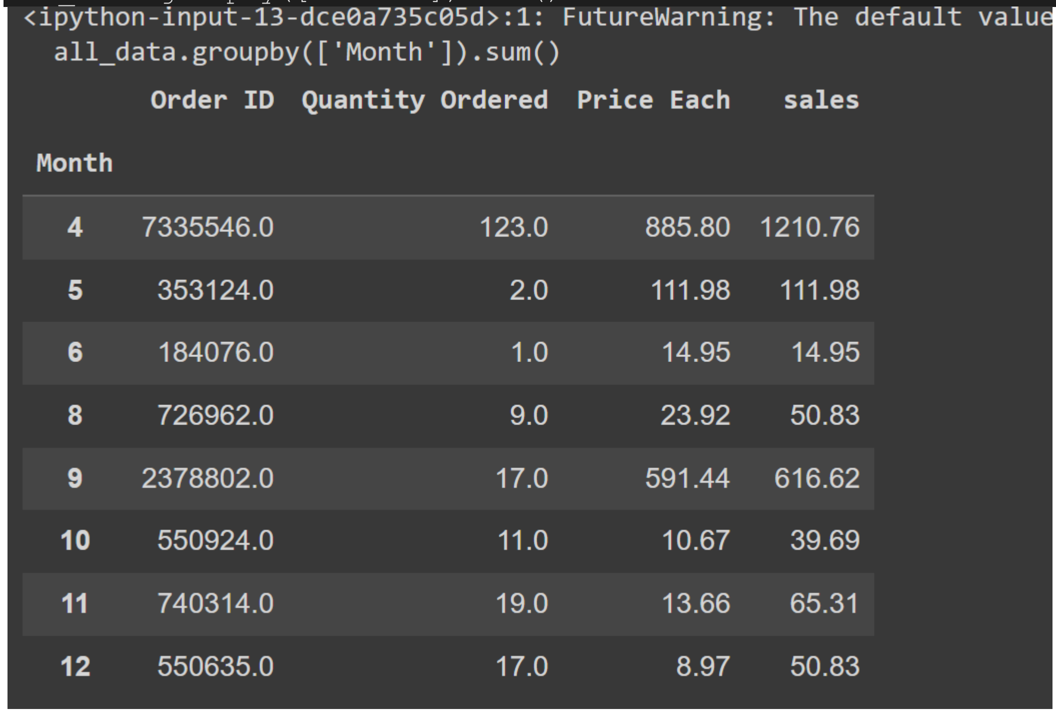


	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	city	sales
0	176559.0	Bose SoundSport Headphones	1.0	99.99	04-07-2019 22:30	682 Chestnut St, Boston, MA 02215	4	Boston (MA)	99.99
1	176560.0	Google Phone	1.0	600.00	04-12-2019 14:38	669 Spruce St, Los Angeles, CA 90001	4	Los Angeles (CA)	600.00
2	176560.0	Wired Headphones	1.0	11.99	04-12-2019 14:38	669 Spruce St, Los Angeles, CA 90001	4	Los Angeles (CA)	11.99
3	176561.0	Wired Headphones	1.0	11.99	05/30/19 9:27	333 8th St, Los Angeles, CA 90001	5	Los Angeles (CA)	11.99
4	176562.0	USB-C Charging Cable	1.0	11.95	04/29/19 13:03	381 Wilson St, San Francisco, CA 94016	4	San Francisco (CA)	11.95

Data Exploration!

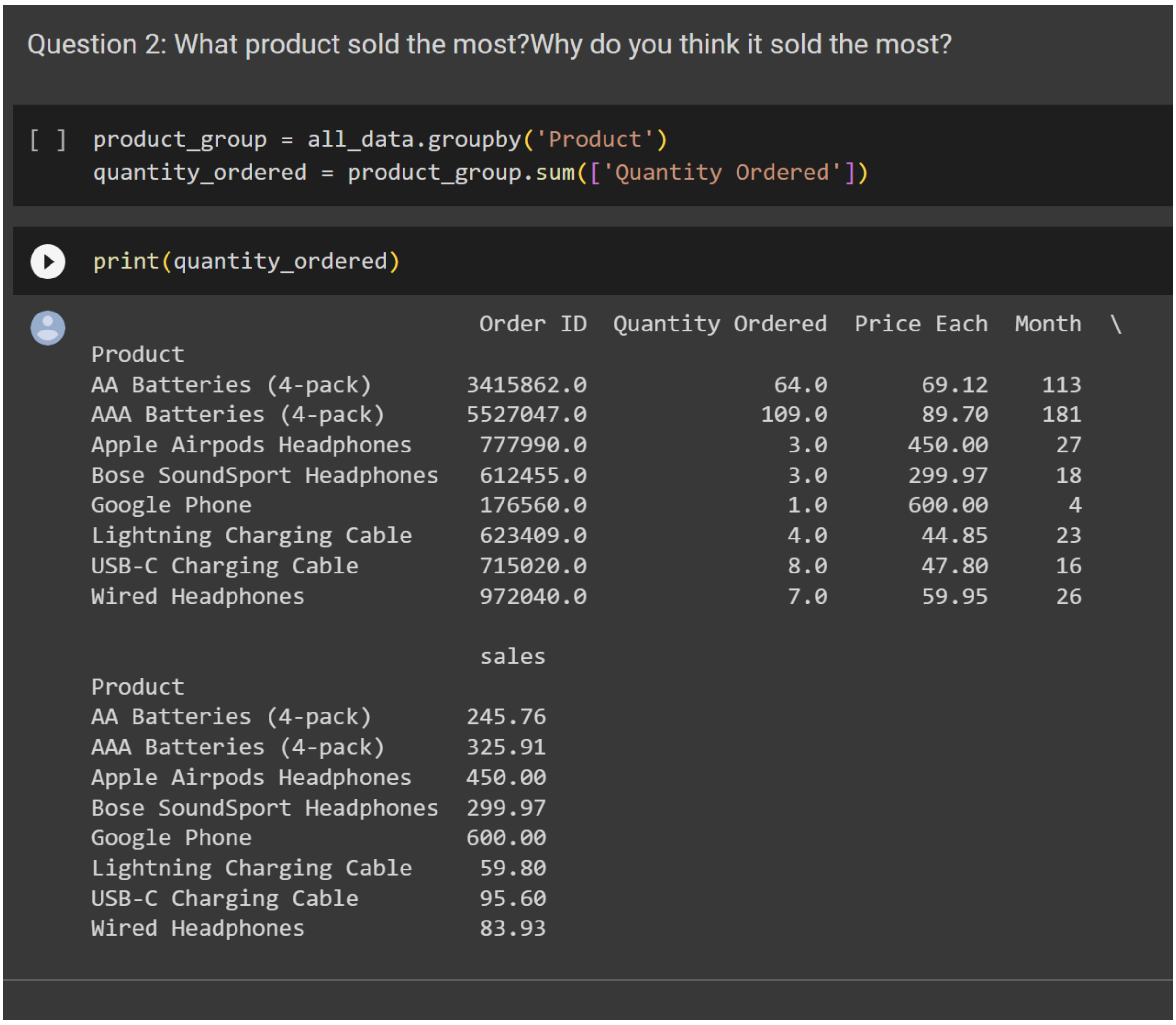
Question1: What was the best month for sales? How much was earned that month?

```python
all_data['sales'] = all_data['Quantity
Ordered'].astype('int')*all_data['Price Each'].astype('float')
all_data.groupby(['Month']).sum()
```



```
<ipython-input-13-dce0a735c05d>:1: FutureWarning: The default value
  all_data.groupby(['Month']).sum()
```

Month	Order ID	Quantity Ordered	Price Each	sales
4	7335546.0	123.0	885.80	1210.76
5	353124.0	2.0	111.98	111.98
6	184076.0	1.0	14.95	14.95
8	726962.0	9.0	23.92	50.83
9	2378802.0	17.0	591.44	616.62
10	550924.0	11.0	10.67	39.69
11	740314.0	19.0	13.66	65.31
12	550635.0	17.0	8.97	50.83

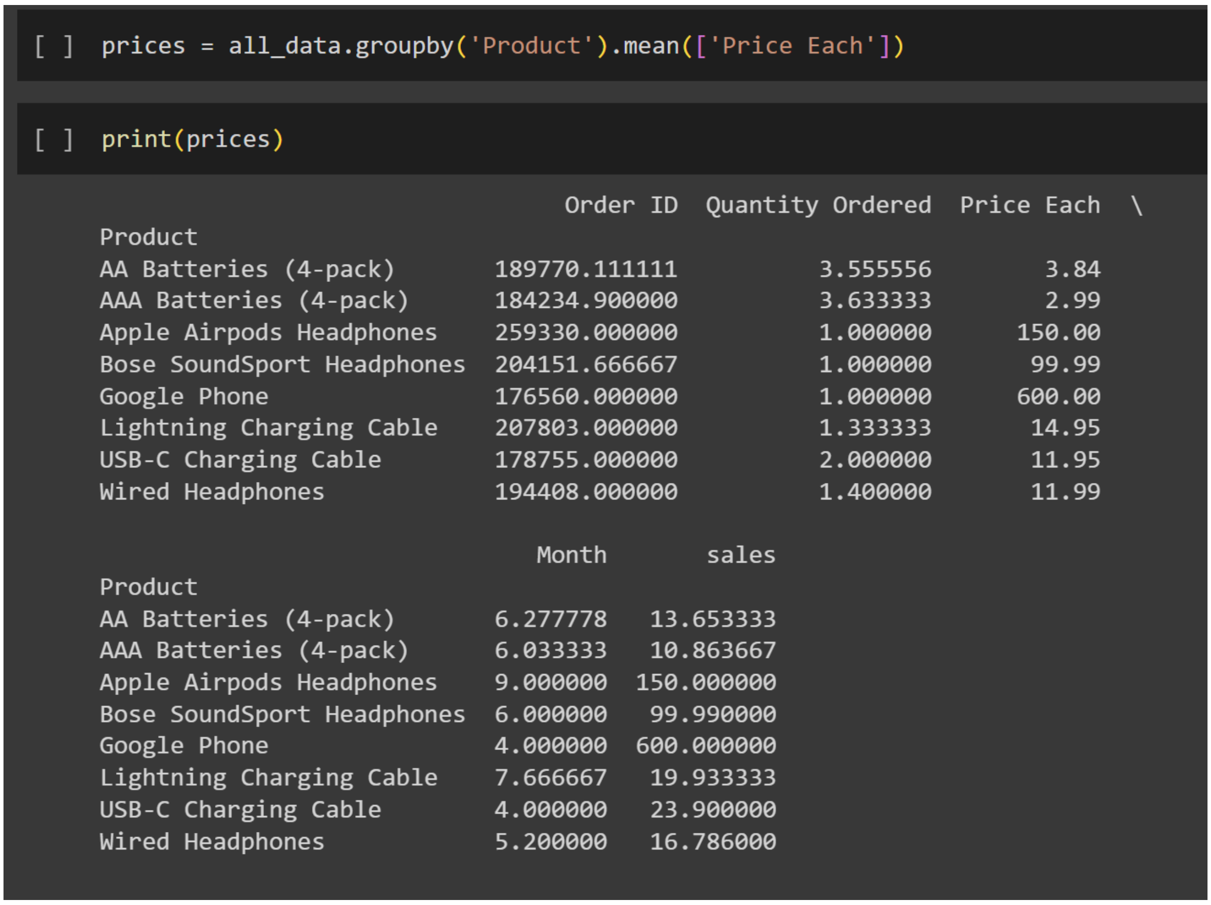

Question 2: What product sold the most?Why do you think it sold the most?

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

```
print(quantity_ordered)
```

```
                            Order ID  Quantity Ordered  Price Each  Month  \
Product
AA Batteries (4-pack)      3415862.0              64.0       69.12    113
AAA Batteries (4-pack)     5527047.0             109.0       89.70    181
Apple Airpods Headphones    777990.0               3.0      450.00     27
Bose SoundSport Headphones  612455.0               3.0      299.97     18
Google Phone                176560.0               1.0      600.00      4
Lightning Charging Cable    623409.0               4.0       44.85     23
USB-C Charging Cable        715020.0               8.0       47.80     16
Wired Headphones            972040.0               7.0       59.95     26

                             sales
Product
AA Batteries (4-pack)       245.76
AAA Batteries (4-pack)      325.91
Apple Airpods Headphones    450.00
Bose SoundSport Headphones  299.97
Google Phone                600.00
Lightning Charging Cable     59.80
USB-C Charging Cable         95.60
Wired Headphones             83.93
```

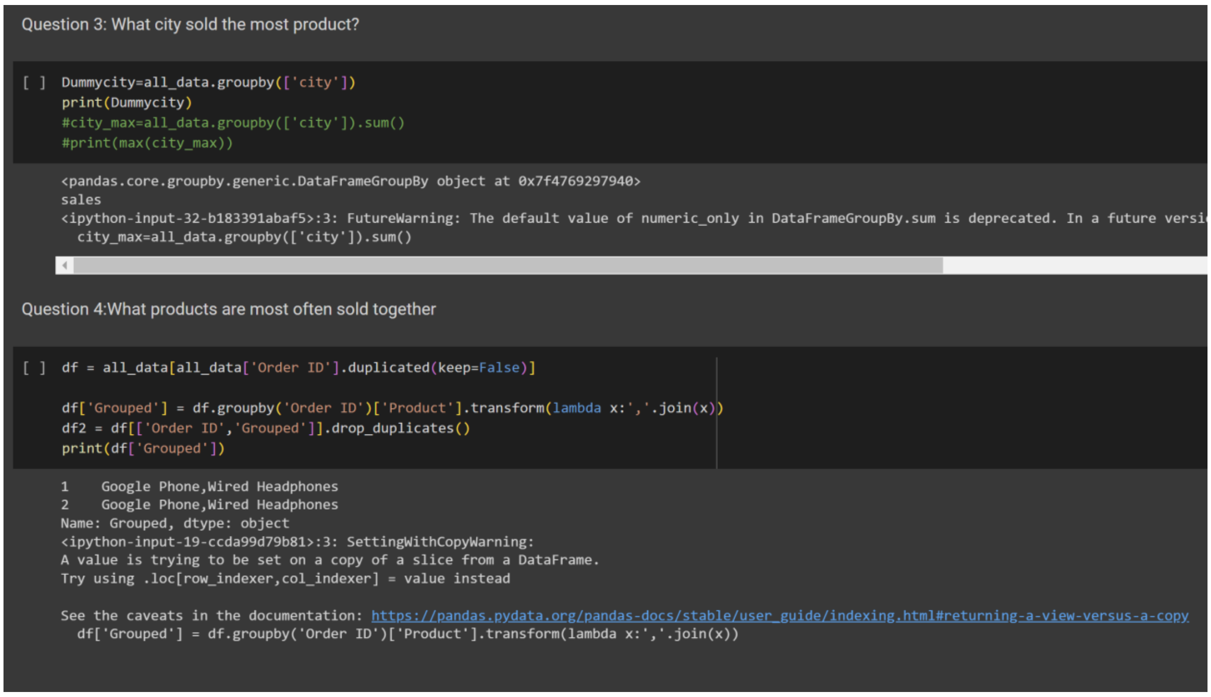


```
[ ]  prices = all_data.groupby('Product').mean(['Price Each'])
```

```
[ ]  print(prices)
```

```
                                Order ID  Quantity Ordered  Price Each  \
Product
AA Batteries (4-pack)      189770.111111          3.555556        3.84
AAA Batteries (4-pack)     184234.900000          3.633333        2.99
Apple Airpods Headphones   259330.000000          1.000000      150.00
Bose SoundSport Headphones 204151.666667          1.000000       99.99
Google Phone               176560.000000          1.000000      600.00
Lightning Charging Cable   207803.000000          1.333333       14.95
USB-C Charging Cable       178755.000000          2.000000       11.95
Wired Headphones           194408.000000          1.400000       11.99

                              Month       sales
Product
AA Batteries (4-pack)      6.277778   13.653333
AAA Batteries (4-pack)     6.033333   10.863667
Apple Airpods Headphones   9.000000  150.000000
Bose SoundSport Headphones 6.000000   99.990000
Google Phone               4.000000  600.000000
Lightning Charging Cable   7.666667   19.933333
USB-C Charging Cable       4.000000   23.900000
Wired Headphones           5.200000   16.786000
```



Question 3: What city sold the most product?

```
[ ]  Dummycity=all_data.groupby(['city'])
     print(Dummycity)
     #city_max=all_data.groupby(['city']).sum()
     #print(max(city_max))
```

```
<pandas.core.groupby.generic.DataFrameGroupBy object at 0x7f4769297940>
sales
<ipython-input-32-b183391abaf5>:3: FutureWarning: The default value of numeric_only in DataFrameGroupBy.sum is deprecated. In a future versi
  city_max=all_data.groupby(['city']).sum()
```

Question 4:What products are most often sold together

```
[ ]  df = all_data[all_data['Order ID'].duplicated(keep=False)]

     df['Grouped'] = df.groupby('Order ID')['Product'].transform(lambda x:','.join(x))
     df2 = df[['Order ID','Grouped']].drop_duplicates()
     print(df['Grouped'])
```

```
1    Google Phone,Wired Headphones
2    Google Phone,Wired Headphones
Name: Grouped, dtype: object
<ipython-input-19-ccda99d79b81>:3: 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))
```