Question 1: Given some sample data, write a program to answer the following: click here to access the required data set

On Shopify, we have exactly 100 sneaker shops, and each of these shops sells only one model of shoe. We want to do some analysis of the average order value (AOV). When we look at orders data over a 30 day window, we naively calculate an AOV of $3145.13. Given that we know these shops are selling sneakers, a relatively affordable item, something seems wrong with our analysis.

Think about what could be going wrong with our calculation. Think about a better way to evaluate this data. What metric would you report for this dataset? What is its value?

1-Importing Data and Create a Data Frame

In [1]:
```
import numpy as np
import pandas as pd
```

In [2]:
```
df=pd.read_csv('data3.csv')
```

In [3]:
```
df.head()
```

Out[3]:

	order_id	shop_id	user_id	order_amount	total_items	payment_method	created_at
0	1	53	746	224	2	cash	3/13/2017 12:36
1	2	92	925	90	1	cash	3/3/2017 17:38
2	3	44	861	144	1	cash	3/14/2017 4:23
3	4	18	935	156	1	credit_card	3/26/2017 12:43
4	5	18	883	156	1	credit_card	3/1/2017 4:35

2- Analysis of Data

In [4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5000 entries, 0 to 4999
Data columns (total 7 columns):
 #   Column          Non-Null Count  Dtype 
---  ------          --------------  ----- 
 0   order_id        5000 non-null   int64 
 1   shop_id         5000 non-null   int64 
 2   user_id         5000 non-null   int64 
 3   order_amount    5000 non-null   int64 
 4   total_items     5000 non-null   int64 
 5   payment_method  5000 non-null   object
 6   created_at      5000 non-null   object
dtypes: int64(5), object(2)
memory usage: 273.6+ KB
```

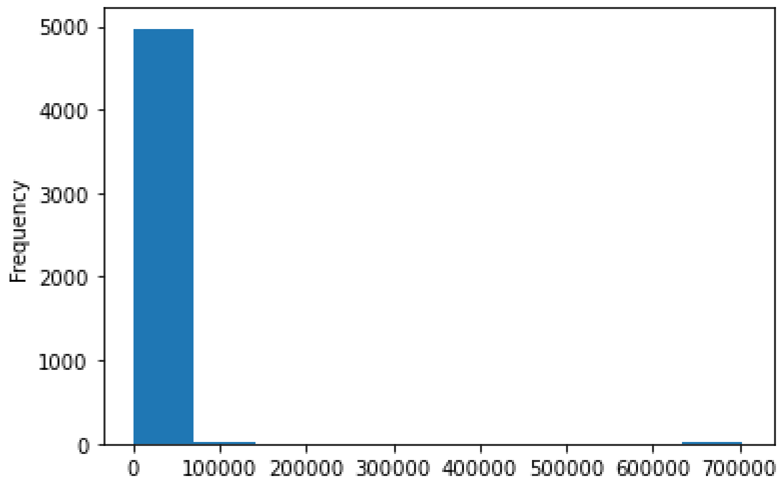
In [5]:
```
df.describe()
```

Out[5]:

	order_id	shop_id	user_id	order_amount	total_items

	order_id	shop_id	user_id	order_amount	total_items
count	5000.000000	5000.000000	5000.000000	5000.000000	5000.00000
mean	2500.500000	50.078800	849.092400	3145.128000	8.78720
std	1443.520003	29.006118	87.798982	41282.539349	116.32032
min	1.000000	1.000000	607.000000	90.000000	1.00000
25%	1250.750000	24.000000	775.000000	163.000000	1.00000
50%	2500.500000	50.000000	849.000000	284.000000	2.00000
75%	3750.250000	75.000000	925.000000	390.000000	3.00000
max	5000.000000	100.000000	999.000000	704000.000000	2000.00000

```
In [6]: ax = df['order_amount'].plot.hist()
```

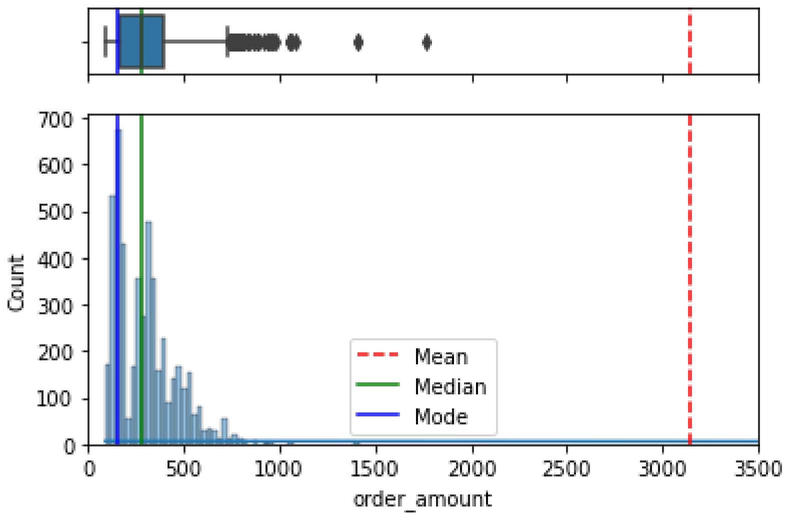


```
In [32]: #Percentage of orders more than 1000
         np.sum(df['order_amount'] > 3500)/5000*100

Out[32]: 1.26
```

Regards to the hist plot and the statistical table,the order amount data is skewed(has outliers).

1. " Order amount" standard deviation is almost 13 times of the mean value.
2. 75% of the orders have values smaller than 390.
3. less than 2% of orders have a value more than 3500. So, the mean (3145.128) can be misleading because the most common values in the distribution are not be near the mean.

```
In [29]: from matplotlib import pyplot as plt
         import seaborn as sns

         f, (ax_box, ax_hist) = plt.subplots(2, sharex=True, gridspec_kw= {"height_ratios": (0.2
         ax_hist.set_xlim([0, 3500])
         mean=df['order_amount'].mean()
         median=df['order_amount'].median()
         mode=df['order_amount'].mode().values[0]

         sns.boxplot(data=df, x=df['order_amount'], ax=ax_box)
         ax_box.axvline(mean, color='r', linestyle='--')
         ax_box.axvline(median, color='g', linestyle='-')
         ax_box.axvline(mode, color='b', linestyle='-')

         sns.histplot(data=df, x=df['order_amount'], ax=ax_hist, kde=True)
         ax_hist.axvline(mean, color='r', linestyle='--', label="Mean")
         ax_hist.axvline(median, color='g', linestyle='-', label="Median")
         ax_hist.axvline(mode, color='b', linestyle='-', label="Mode")

         ax_hist.legend()

         ax_box.set(xlabel='')
         plt.show()
```



The above picture only shows data in the range of 0 to 3500(around 98% of the data).The mean is realy far from the majority of data. The Median is a better metric to describe the order amount status.

3- Better Metric

In [8]:
```
#Calculate Median
df['order_amount'].median()
```

Out[8]:
```
284.0
```

Mode is also in a range of around 200.

In [31]:
```
#Calculate Mode
df['order_amount'].mode()
```

Out[31]:
```
0    153
dtype: int64
```

Summary: Only one statistical metric like “ MEAN” is not always enough to give a great overall view of data .For data with outliers "MEDIAN” is more reasonable metric .

In [ ]: