In [1]:

```
import pandas as pd
import numpy as np
import seaborn as sns
```

```
C:\Users\rushi\Anaconda3\lib\site-packages\statsmodels\tools\_testing.py:19: Fu
tureWarning: pandas.util.testing is deprecated. Use the functions in the public
API at pandas.testing instead.
  import pandas.util.testing as tm
```

Question 1

In [2]:

```
df = pd.read_excel('2019 Winter Data Science Intern Challenge Data Set.xlsx'
```

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	2017-03-13 12:36:56.190
1	2	92	925	90	1	cash	2017-03-03 17:38:51.999
2	3	44	861	144	1	cash	2017-03-14 04:23:55.595
3	4	18	935	156	1	credit_card	2017-03-26 12:43:36.649
4	5	18	883	156	1	credit_card	2017-03-01 04:35:10.773

In [4]:

```
df.describe()
```

Out[4]:

	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 [5]:

```
df['order_amount'].describe()
```

Out[5]:

```
count      5000.000000
mean       3145.128000
std       41282.539349
min          90.000000
25%         163.000000
50%         284.000000
75%         390.000000
max      704000.000000
Name: order_amount, dtype: float64
```

Question 1(a): Think about what could be going wrong with our calculation. Think about a better way to evaluate this data.

1.As we can see above that mean is 3145.12 and the standard deviation is 41282.53, which means that value vary 41282.53 from the mean which is very high. High Standard deviation value means that mean is not right representation of AOV.

2.we can also see that minumum value, first quartile value, third quartile value and median is very small compared to maximum value which means that there are outliers in our order_amount column which are draging mean value up.

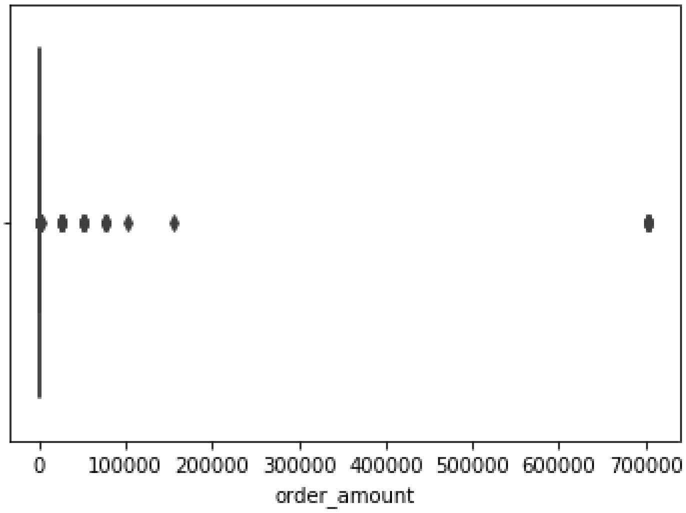
3.Much better way to evaluate this dataset is going by the median of the the column.

In [6]:

```
# Let us check the boxplot for better representation of distribution in orde
sns.boxplot('order_amount', data = df)
```

Out[6]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x23bebdf7780>
```



In [7]:

```python
# Lets try to group the order_amount So that we can know that how many time
amount_group = df.groupby('order_amount').size().reset_index(name = 'count')
amount_group.head(10)
```

Out[7]:

	order_amount	count
257	704000	17
256	154350	1
255	102900	1
254	77175	9
253	51450	16
252	25725	19
251	1760	1
250	1408	2
249	1086	1
248	1064	1

From Above we can clearly See that order_amount 70400 has 17 counts in the similar way 77175, 51450 and 25725 has 9,16 and 19 counts. Let us try to look for this values in the dataset.

In [8]:

```
df.loc[df['order_amount'].isin([704000,77175,51450,25725])].sort_values(by =
```

Out[8]:

	order_id	shop_id	user_id	order_amount	total_items	payment_method	created_at
15	16	42	607	704000	2000	credit_card	2017-03-07 04:00:00.000
1362	1363	42	607	704000	2000	credit_card	2017-03-15 04:00:00.000
3332	3333	42	607	704000	2000	credit_card	2017-03-24 04:00:00.000
4056	4057	42	607	704000	2000	credit_card	2017-03-28 04:00:00.000
2969	2970	42	607	704000	2000	credit_card	2017-03-28 04:00:00.000
...	...	...	...	...	...	...	...
1204	1205	78	970	25725	1	credit_card	2017-03-17 22:32:21.438
1193	1194	78	944	25725	1	debit	2017-03-16 16:38:25.551
1056	1057	78	800	25725	1	debit	2017-03-15 10:16:44.830
160	161	78	990	25725	1	credit_card	2017-03-12 05:56:56.834
4918	4919	78	823	25725	1	cash	2017-03-15 13:26:46.262

61 rows × 7 columns

From above we can see that all the transcation of order_amount 704000 was at the same time on different dates as shownn in created_at column and also the total_items column shows that 2000 units were purchases and also we can see that this transactions were done by the same user_id on same shop_id, so it means that this transcation might be done by supplier or by wholesaler.

Question 1(b) What metric would you report for this dataset?

As I have already mention above that due to outlier mean is not the right metric for representation , So I would report median as a right metric for this dataset.

Question 1(c) What is its value?

In [11]:

```
df['order_amount'].median()
```

Out[11]:

```
284.0
```

The better way to represent AOV is by median. The value of median is 284 as mention above.

Question 2

Question 2(a) How many orders were shipped by Speedy Express in total?

```sql
SELECT COUNT(*) as TotalOrders
FROM Orders ord
JOIN Shippers sp ON ord.ShipperID = sp.ShipperID
Where sp.ShipperName = "Speedy Express"
```

Total Number of orders shipped by Speedy Express is 54.

Question 2(b) What is the last name of the employee with the most orders?

```sql
SELECT emp.LastName, COUNT(*) AS TotalOrders
FROM Orders ord
JOIN Employees emp ON emp.EmployeeID = ord.EmployeeID
GROUP BY emp.LastName
ORDER BY TotalOrders DESC
LIMIT 1
```

Peacock has the most orders that is 40

Question 2(c) What product was ordered the most by customers in Germany?

```sql
SELECT pd.ProductName, SUM(od.Quantity) AS Amount_of_order
FROM Orders ord JOIN Customers cus ON Cus.CustomerID = ord.CustomerID
JOIN OrderDetails od ON od.OrderID = Ord.OrderID
JOIN Products  pd ON pd.ProductID = od.ProductID WHERE cus.Country = 'Germany'
GROUP BY od.ProductID
ORDER BY Amount_of_order DESC
```

Boston Crab Meat was ordered the most by customers in germany with 160 orders.

In [ ]:

```

```