In [39]:
```
import pandas as pd
from scipy.stats import zscore,norm
import matplotlib.pyplot as plt
import seaborn as sns
```

Task1: To calculate household probabilities of cellular

In [7]:
```
avg=70
std=11.351
x1=55
x2=80
zscore1=(x1-70)/11.351
zscore2=(x2-70)/11.351
zscore3=(0-70)/11.351
zscore4=(40-70)/11.351
p1= norm.cdf(zscore1)
p2=norm.cdf(zscore2)
p3=norm.cdf(zscore3)
p4=norm.cdf(zscore4)
prob=(1-p1)-(1-p2)
prob2=(1-p3)-(1-p4)
print("Probability of finding between 55USD and 8USD",round(prob,2))
print("Probability of finding less than 40USD",round(prob2,5))
```

```
Probability of finding between 55USD and 8USD 0.72
Probability of finding less than 40USD 0.00411
```

In [8]:
```
smart=pd.read_csv(r"K:\Desktop\NIIT\tables\DS1_C5_S5_SmartCarRiding_Data_Practice.csv")
smart
```

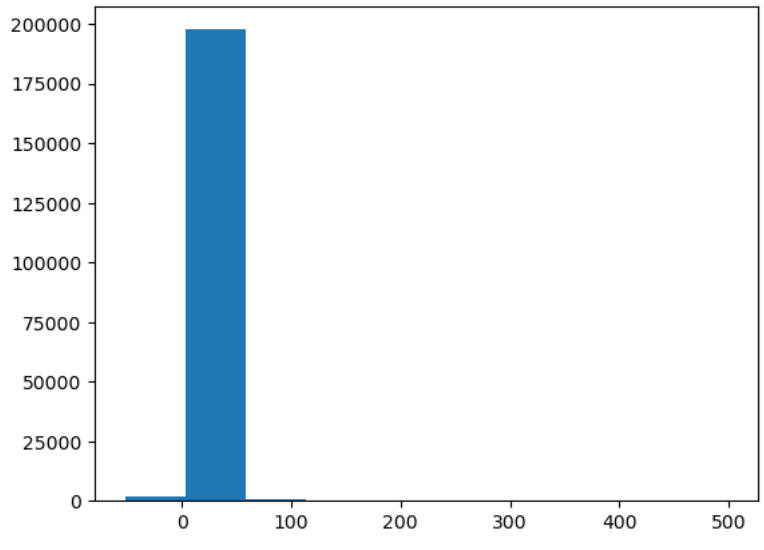
Out[8]:

	Unnamed: 0	key	fare_amount	pickup_datetime	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passenger_count
0	24238194	2015-05-07 19:52:06.0000003	7.5	2015-05-07 19:52:06 UTC	-73.999817	40.738354	-73.999512	40.723217	1
1	27835199	2009-07-17 20:04:56.0000002	7.7	2009-07-17 20:04:56 UTC	-73.994355	40.728225	-73.994710	40.750325	1
2	44984355	2009-08-24 21:45:00.00000061	12.9	2009-08-24 21:45:00 UTC	-74.005043	40.740770	-73.962565	40.772647	1
3	25894730	2009-06-26 08:22:21.0000001	5.3	2009-06-26 08:22:21 UTC	-73.976124	40.790844	-73.965316	40.803349	3
4	17610152	2014-08-28 17:47:00.000000188	16.0	2014-08-28 17:47:00 UTC	-73.925023	40.744085	-73.973082	40.761247	5
...	...	...	...	...	...	...	...	...	...
199995	42598914	2012-10-28 10:49:00.00000053	3.0	2012-10-28 10:49:00 UTC	-73.987042	40.739367	-73.986525	40.740297	1
199996	16382965	2014-03-14 01:09:00.0000008	7.5	2014-03-14 01:09:00 UTC	-73.984722	40.736837	-74.006672	40.739620	1
199997	27804658	2009-06-29 00:42:00.00000078	30.9	2009-06-29 00:42:00 UTC	-73.986017	40.756487	-73.858957	40.692588	2
199998	20259894	2015-05-20 14:56:25.0000004	14.5	2015-05-20 14:56:25 UTC	-73.997124	40.725452	-73.983215	40.695415	1
199999	11951496	2010-05-15 04:08:00.00000076	14.1	2010-05-15 04:08:00 UTC	-73.984395	40.720077	-73.985508	40.768793	1

200000 rows × 9 columns

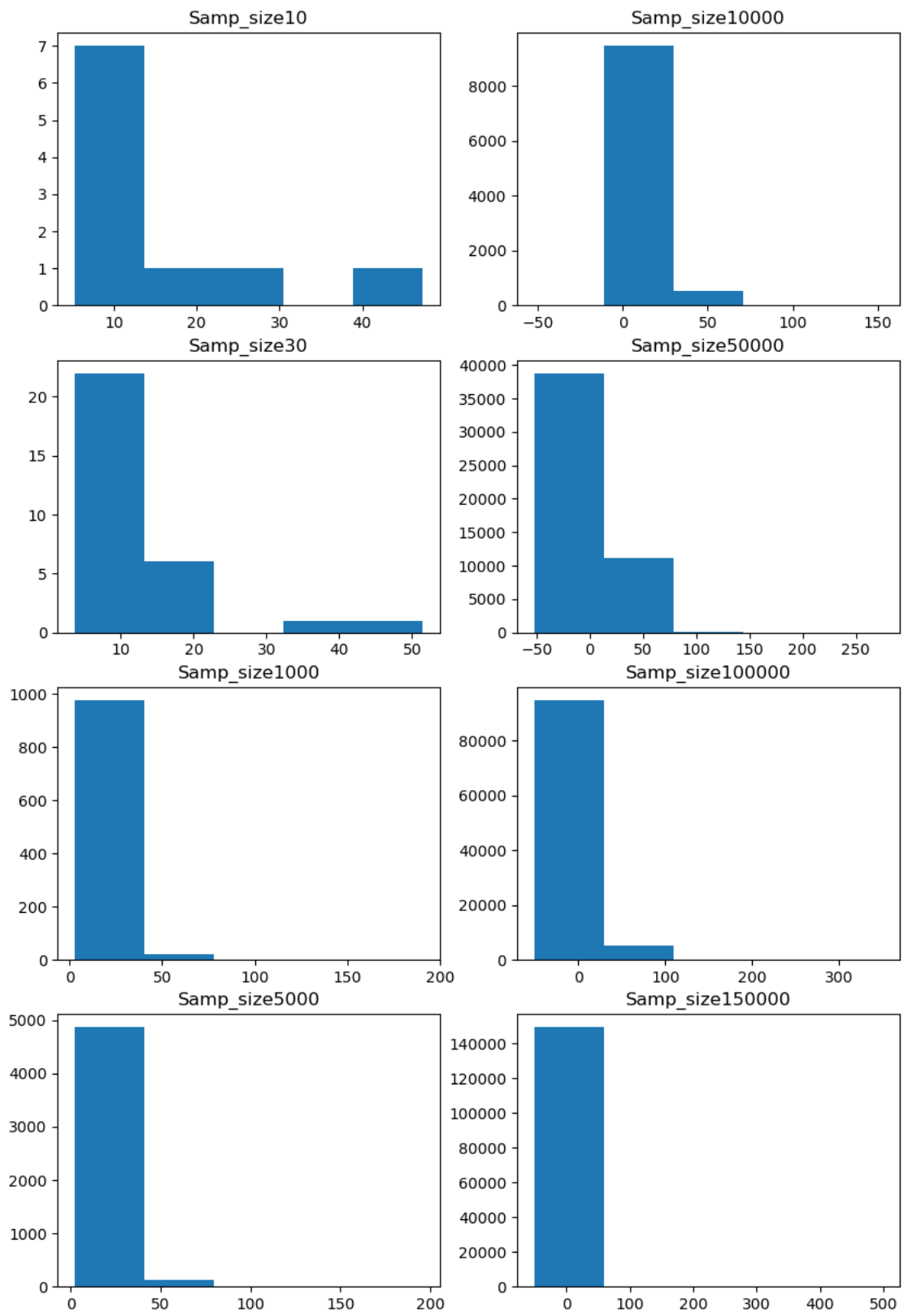
Data preprocessing

In [9]:
```
plt.hist(smart.fare_amount);
```



In [51]:
```
samp=[10,30,1000,5000,10000,50000,100000,150000]
fig,ax=plt.subplots(4,2,figsize=(10,15))
j=0
k=0
for i in samp:
    garb=smart["fare_amount"].sample(i,ignore_index=True)
    ax[j,k].hist(garb,bins=5)
    ax[j,k].set_title("Samp_size"+str(i))
    j+=1
    if(j==4):
        k+=1
        j=0

```



To extract data using random sampling with replace=True

In [11]:
```
from scipy.stats import skew,kurtosis
import statistics as st
```

In [23]:
```
samples=pd.DataFrame()
cols=[("Sample_"+str(i)) for i in range(0,21)]
for i in range(0,21):
    samples[cols[i]]=smart["fare_amount"].sample(200,replace=True,ignore_index=True)
samples


```

Out[23]:

	Sample_0	Sample_1	Sample_2	Sample_3	Sample_4	Sample_5	Sample_6	Sample_7	Sample_8	Sample_9	...	Sample_11	Sample_12	Sample_13	Sample_14	Sample_15	Sample_16	Sample_17	Sample_18	Sample_19	Sa
0	8.50	6.1	26.5	5.00	10.50	8.90	34.83	4.50	7.70	7.7	...	3.70	4.5	8.5	6.5	3.7	12.50	12.1	5.5	13.0	
1	7.50	3.7	14.5	6.90	4.10	6.10	10.50	5.30	11.00	19.0	...	29.85	8.5	11.7	10.5	4.5	7.50	5.8	9.3	9.3	
2	4.90	8.1	4.5	4.50	37.83	4.50	8.00	35.50	57.33	9.3	...	7.30	14.1	9.7	12.0	18.5	28.27	4.1	9.5	7.5	
3	8.50	7.5	9.5	5.30	9.00	38.33	10.50	10.90	11.70	10.5	...	4.50	5.0	8.5	6.0	13.0	4.90	4.5	8.9	8.0	
4	7.50	18.9	6.1	29.07	6.50	17.70	30.00	14.00	14.90	18.5	...	19.50	7.0	6.1	13.0	14.5	4.50	6.5	5.3	13.0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
195	13.50	14.5	47.5	13.50	18.50	8.90	7.70	6.00	3.70	23.0	...	7.50	7.7	10.0	6.9	9.3	9.30	15.3	9.3	5.3	
196	14.00	11.5	8.5	14.00	57.33	6.50	6.10	11.30	5.30	7.3	...	16.50	9.3	11.3	7.5	9.5	6.50	6.5	16.0	24.5	
197	11.00	33.7	9.0	6.90	34.04	45.00	13.30	39.33	8.90	7.0	...	7.70	12.5	4.5	30.5	18.5	7.30	10.1	16.1	6.0	
198	6.50	16.5	4.1	6.00	7.70	7.70	17.30	7.70	26.65	5.5	...	11.30	17.0	8.9	4.5	4.5	6.90	7.5	9.3	8.5	
199	14.25	8.0	13.3	7.70	4.10	4.10	4.00	5.30	4.10	6.1	...	16.00	6.5	6.5	7.3	3.3	5.70	5.3	4.1	8.0	

To find measures of central tendency and d

In [16]:
```
mean=[]
median=[]
mode=[]
kurtosis=[]
skewness=[]
std=[]
for item in samples.columns:
    mean.append(samples[item].mean())
    median.append(samples[item].median())
    mode.append(st.mode(samples[item]))
    std.append(samples[item].std())
    skewness.append(skew(samples[item]))
    kurtosis.append(samples[item].kurtosis())
measures=pd.DataFrame([mean,median,mode,std,skewness,kurtosis],columns=[samples.columns],index=["Mean","Median","Mode","Standard_dev","Skew","Kurtosis"])
```

In [17]:
```
measures
```

Out[17]:

	Sample_0	Sample_1	Sample_2	Sample_3	Sample_4	Sample_5	Sample_6	Sample_7	Sample_8	Sample_9	...	Sample_11	Sample_12	Sample_13	Sample_14	Sample_15	Sample_16	Sample_17	Sample_18	Sample_
Mean	11.517600	11.661000	10.593000	11.830550	12.486750	11.012600	11.841100	10.738850	11.773300	11.976750	...	11.535700	10.525500	11.051150	11.115350	11.035450	11.390400	10.961100	10.386300	11.8217
Median	8.500000	8.500000	8.000000	8.500000	8.700000	8.000000	8.900000	8.500000	8.500000	8.300000	...	8.900000	8.500000	8.000000	8.100000	8.500000	8.500000	8.050000	7.700000	8.5000
Mode	6.500000	6.500000	8.500000	4.900000	8.500000	4.500000	4.500000	4.500000	8.500000	6.500000	...	4.100000	8.500000	4.900000	6.500000	8.500000	6.500000	4.500000	4.500000	6.5000
Standard_dev	10.458328	11.263995	8.468545	10.252627	13.404688	8.969840	11.668283	8.185524	9.366782	10.768075	...	9.229335	8.881917	9.629662	8.886643	8.394579	9.106653	9.710686	8.732315	10.1791
Skew	4.221879	4.304210	3.037639	2.650034	4.975068	2.429899	3.528323	2.848944	2.553835	2.783555	...	2.577480	3.961469	3.109739	2.720313	2.863306	2.433555	2.920328	3.106022	2.9887
Kurtosis	27.433934	29.771915	11.690454	8.005107	36.131199	6.264519	15.440936	10.819176	7.672602	8.576162	...	7.469375	22.274932	12.704213	8.938426	10.761655	6.936306	9.584929	13.295919	11.2780

6 rows × 21 columns

In [53]:
```
print(measures.loc["Mean"].std())
print(smart.fare_amount.std()/(200)**0.5)
print("From the values we can see we have very close values which obeys third rule of CL theorem and all the samples show very similar normal distribution so follows rule1")
```

```
0.5406728817276805
0.7001613114537891
From the values we can see we have very close values which obeys third rule of CL theorem and all the samples show very similar normal distribution so follows rule1
```