In [271]:
```
#Name - Paul Galvez
#Final Project DSC530
#Date - 3/4/23
```

In [272]:
```
import pandas as pd
```

In [273]:
```
import thinkstats2
```

In [274]:
```
import numpy as np
```

In [275]:
```
import thinkplot
```

In [276]:
```
import nsfg
```

In [277]:
```
import matplotlib
```

In [278]:
```
import first
```

In [279]:
```
#Importing the airlines delay data CSV file into Python/Juypter

pd.read_csv('AirlinesEdit.csv')
```

0	CO	3	15	205	1
1	US	3	15	222	1
2	AA	3	20	165	1
3	AA	3	20	195	1
4	AS	3	30	202	0
...	...	...	...	...	...
539378	CO	5	1439	326	0
539379	FL	5	1439	305	0
539380	FL	5	1439	255	0
539381	UA	5	1439	313	1
539382	US	5	1439	301	1

539383 rows × 5 columns

In [280]:
```
Airline_delay = pd.read_csv('AirlinesEdit.csv')
```

In [281]:

```
Airline_delay.head(10)
```

Out[281]:

	Airline	DayOfWeek	Time	Length	Delay
0	CO	3	15	205	1
1	US	3	15	222	1
2	AA	3	20	165	1
3	AA	3	20	195	1
4	AS	3	30	202	0
5	CO	3	30	181	1
6	DL	3	30	220	0
7	DL	3	30	228	0
8	DL	3	35	216	1
9	AA	3	40	200	1

In [282]:

```
df =Airline_delay
print(df)
```

```
       Airline  DayOfWeek  Time  Length  Delay
0           CO          3    15     205      1
1           US          3    15     222      1
2           AA          3    20     165      1
3           AA          3    20     195      1
4           AS          3    30     202      0
...        ...        ...   ...     ...    ...
539378      CO          5  1439     326      0
539379      FL          5  1439     305      0
539380      FL          5  1439     255      0
539381      UA          5  1439     313      1
539382      US          5  1439     301      1

[539383 rows x 5 columns]
```

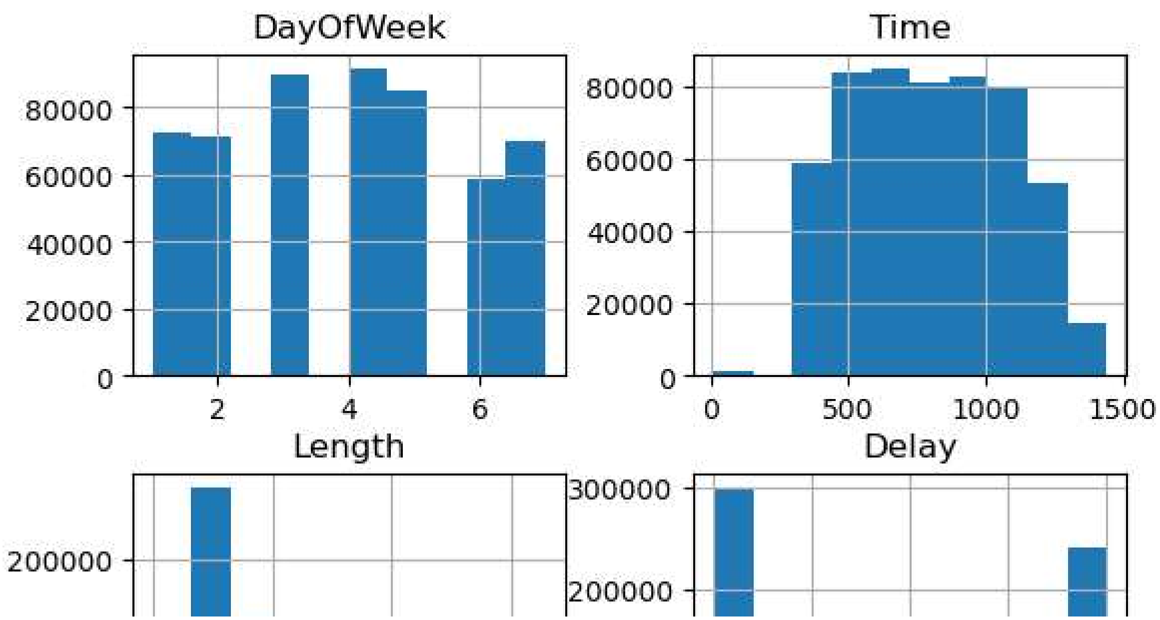
In [283]:

```
#Describe what the 5 variables mean in the dataset:

#The project will be centered on flight delays and exploring the data that centers around
#predicting, analyzing, and exploring those delays across multiple airlines.

#The variables we will see are:

#Airline - the specific airlines associted with the delays. The airline codes are as follows:
#UA United Air Lines Inc.
#AA American Airlines Inc.
#US US Airways Inc.
#F9 Frontier Airlines Inc.
#B6 JetBlue Airways
#OO Skywest Airlines Inc.
#AS Alaska Airlines Inc.
#NK Spirit Air Lines
#WN Southwest Airlines Co.
#DL Delta Air Lines Inc.
#EV Atlantic Southeast Airlines
#HA Hawaiian Airlines Inc.
#MQ American Eagle Airlines Inc.
#VX Virgin America

#DayOfWeek - The day of the week where the flight delay was recorded. The days are associted with
#a number for ID purposes. The number 1 is Monday and 7 is Sunday. Each day will fall in order of the
#calendar week between 1 and 7.

#Time -  the time of the delay is recorded in 24hr formatting. For example, if a flight delay was recorde
#at 1425, the time is 2:45pm on the 12hr clock.

#Length - The length of the delay represented in minutes.

#Delay - The record of weither the flight experienced a delay as a result of the incident. The data
#is shown as 0 or 1 where 0 is false or the flight was not delayed and 1 meaning true where the flight
#was recorded as a delay
```

In [284]:

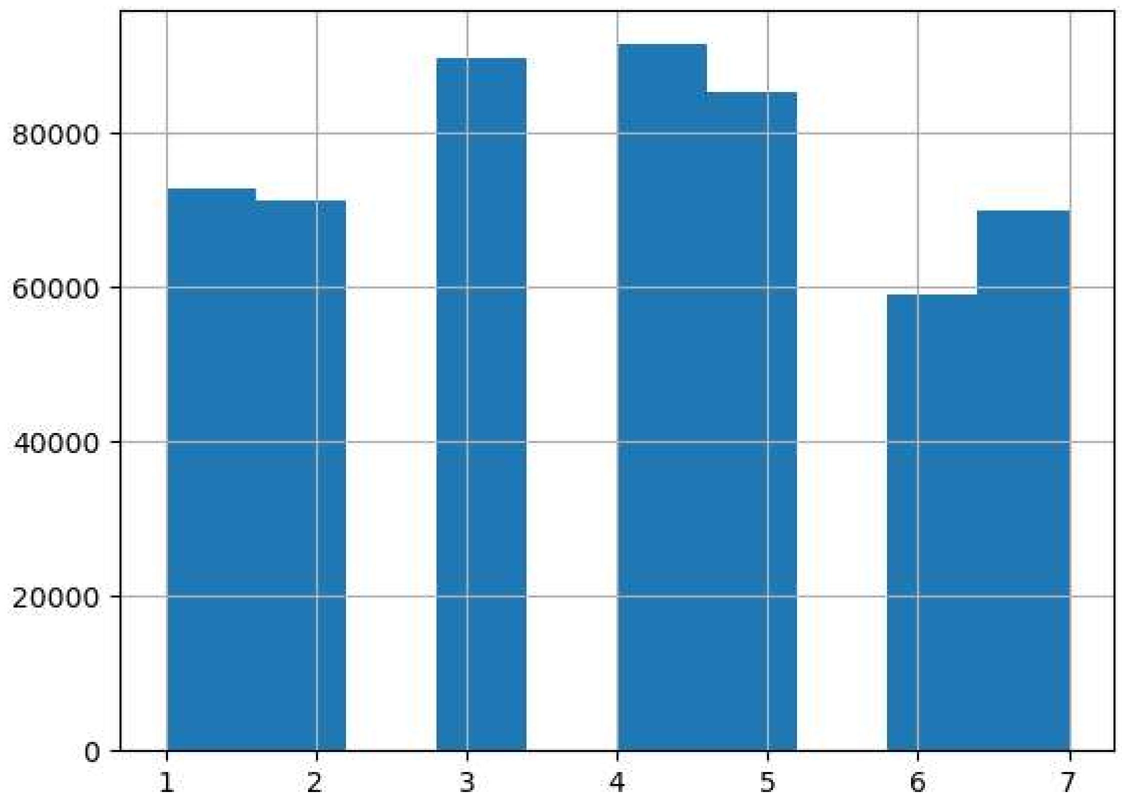
```python
#The histograms of each of the variables for flight delays:

Airline_delay.hist()
```

```
<AxesSubplot:title={'center':'Delay'}>]], dtype=object)
```



In [285]:
```
Airline_delay.DayOfWeek.hist()
```

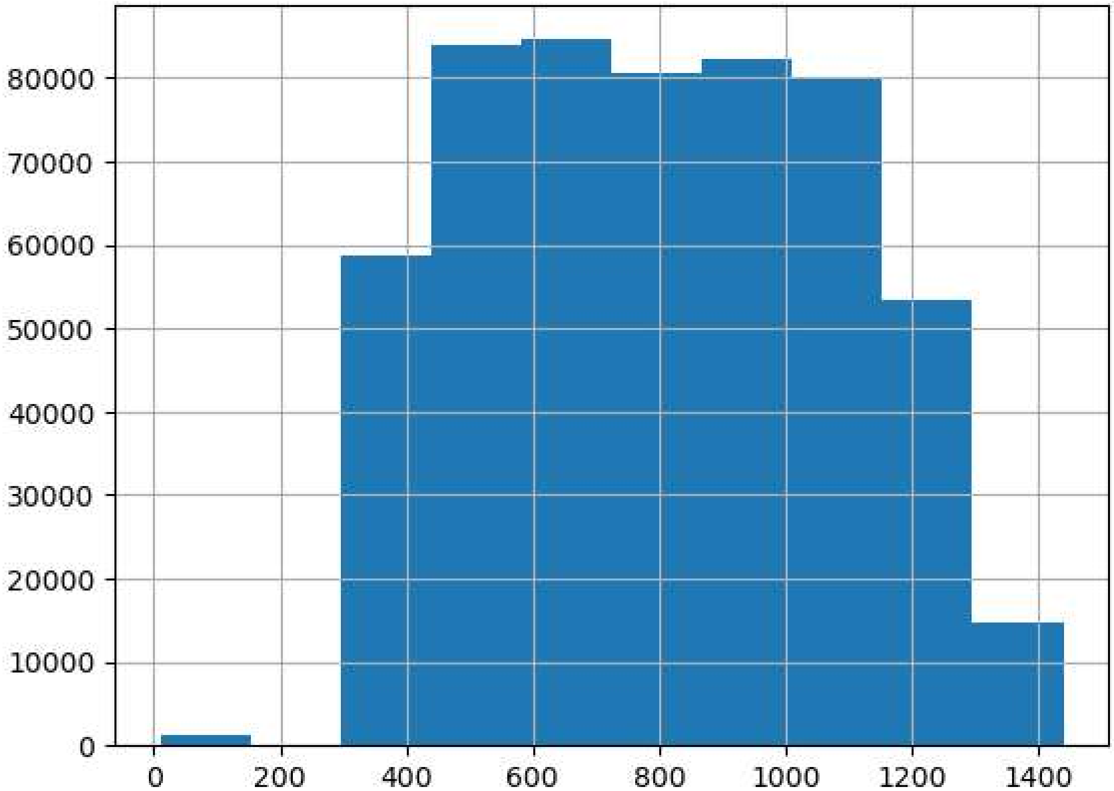
Out[285]: <AxesSubplot:>



In [286]:

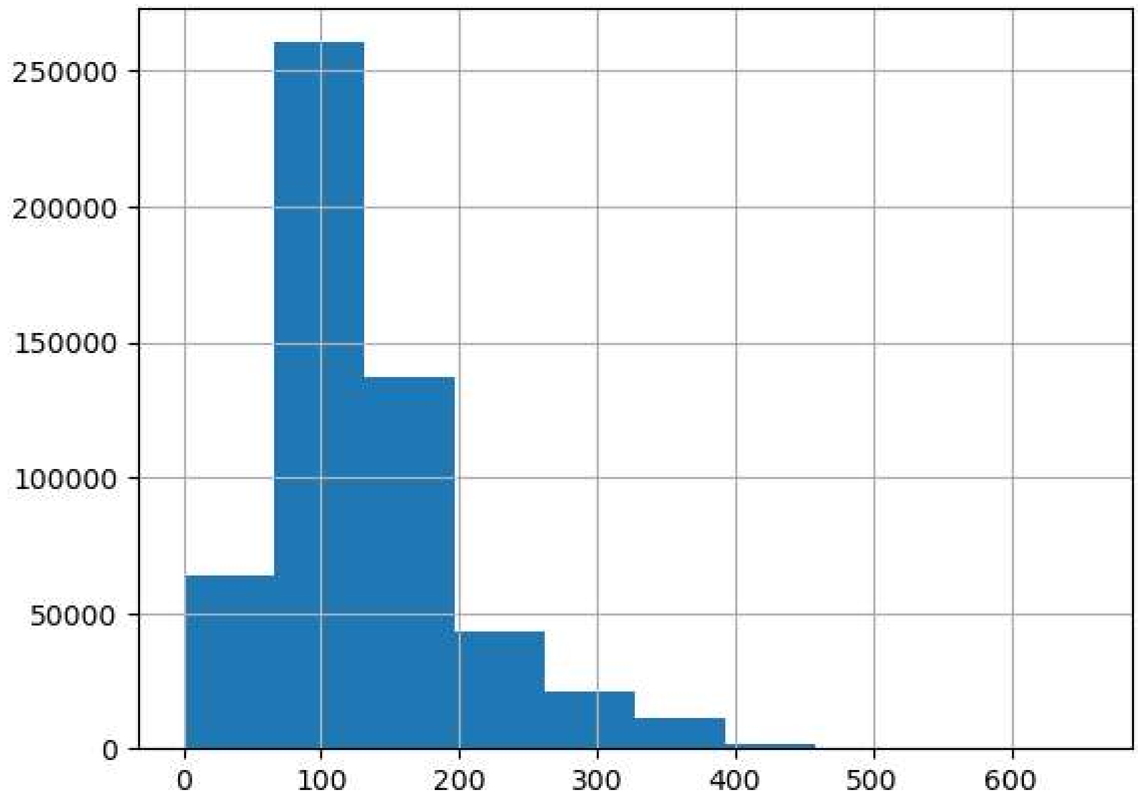
```
Airline_delay.Time.hist()
```

Out[286]: <AxesSubplot:>



In [287]:
```
Airline_delay.Length.hist()
```

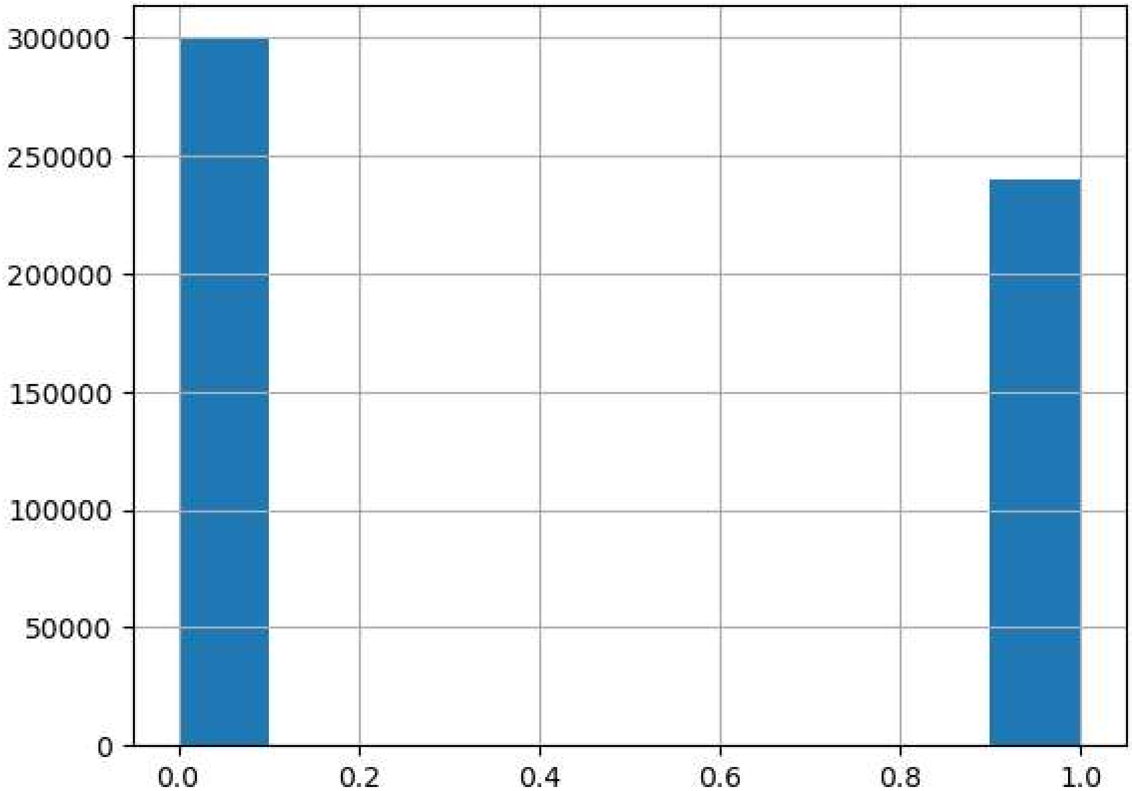
Out[287]: <AxesSubplot:>



In [288]:

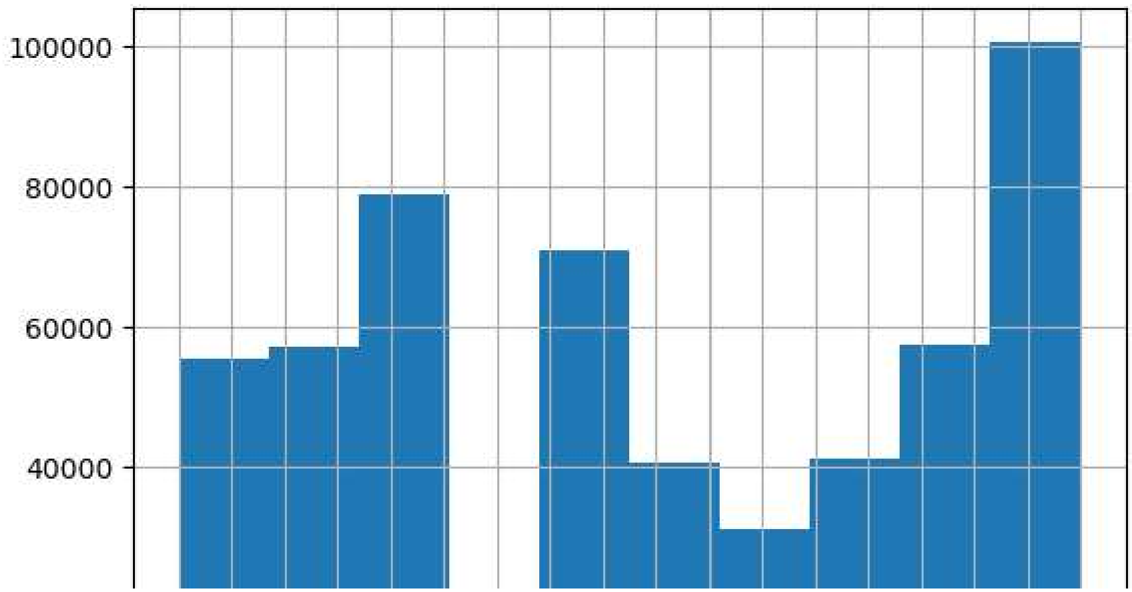
```
Airline_delay.Delay.hist()
```

Out[288]: <AxesSubplot:>



In [289]:
```
Airline_delay.Airline.hist()
```

Out[289]: <AxesSubplot:>



In [290]:
```
#There are two outliers in two of the histograms
#First - For time of delay, there are outliers for time of day for the earliest hours on the 24hr
#clock. Second - HA Hawaiian Airlines is the outlier for the airlines who reported or recorded delays
#From an outlier standpoint, we can assume there aren't many flights leaving or arriving during those
#times of the 24hr clock and we can use domain knowledge to acknowledge the fact that there aren't as
#many flights planned for that time of night. For Hawaiian Airlines, we can use domain knowedge and
#find out Hawaiian Airlines was the most in time airline in the country and therefore, is not common
#amongst airlines in terms of on time performance across all flights and airlines.
```

In [291]:
```
#The mean for length of delay is 132 min. Around an hour and half

mean = Airline_delay.Length.mean()
print(mean)
```

132.20200673732765

In [292]:
```
#The mean of day of week the delay occured approx. 3.9 -4.0 or between Weds and Thurs.

mean = Airline_delay.DayOfWeek.mean()
print(mean)
```

```
3.929667787082648
```

In [293]:
```
#The mean for time of delay is 802. Or on the 12 hr clock 8:02am

mean = Airline_delay.Time.mean()
print(mean)
```

```
802.7289625368245
```

In [294]:
```
var = Airline_delay.Length.var()
print(var)
```

```
4916.395876298452
```

In [295]:
```
var = Airline_delay.DayOfWeek.var()
print(var)
```

```
3.665939681295982
```

In [296]:
```
var = Airline_delay.Time.var()
print(var)
```

```
77309.52852193319
```

In [297]:
```
std = Airline_delay.Length.std()
print(std)
```

```
70.11701559748855
```

In [298]:

```
std = Airline_delay.DayOfWeek.std()
print(std)
```

1.9146643782386463

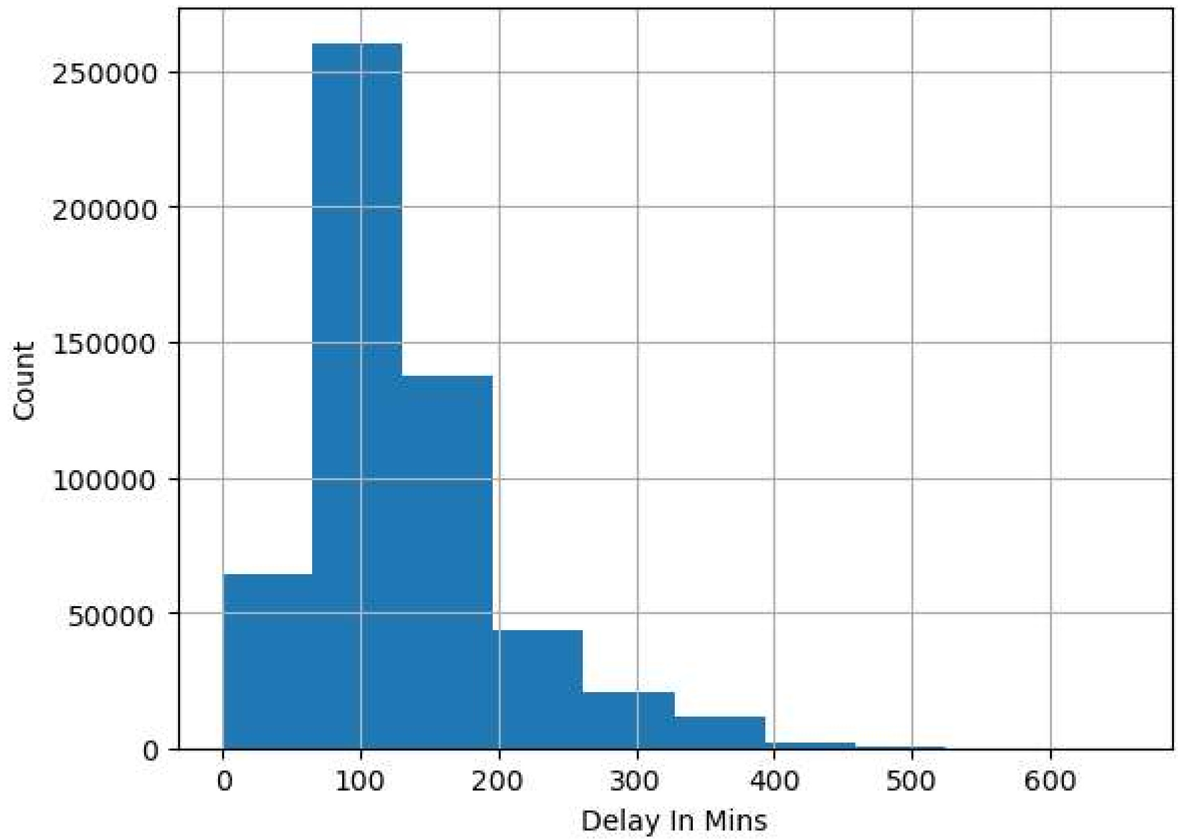
In [299]:

```
std = Airline_delay.Time.std()
print(std)
```

278.04591081678075

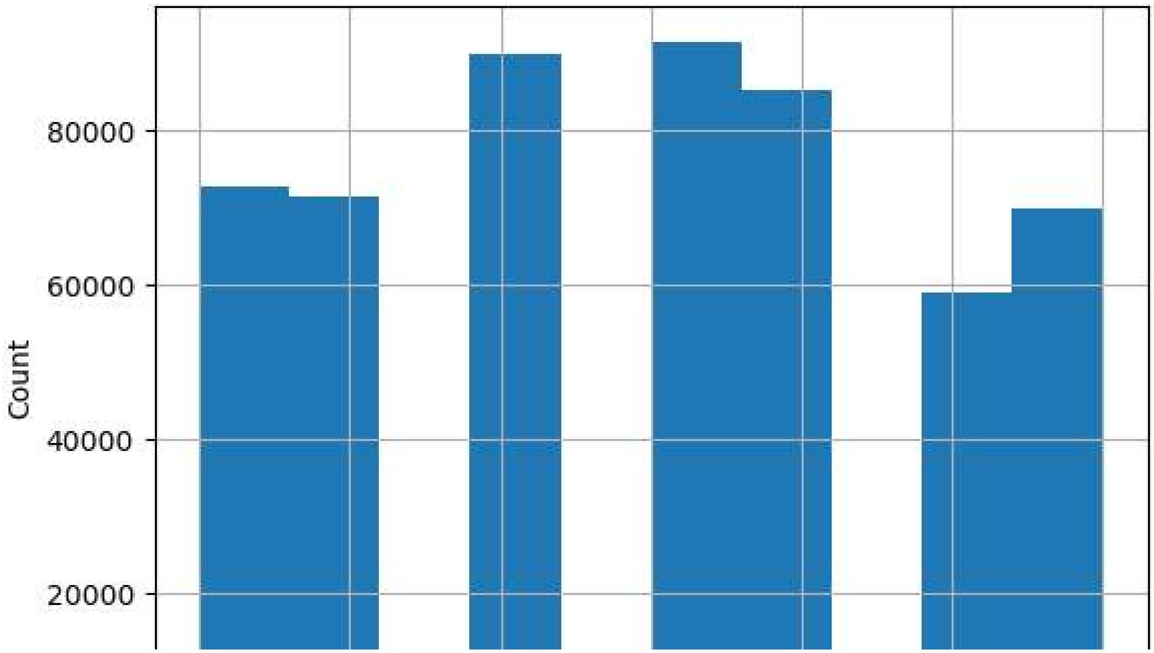
In [300]:

```
hist = Airline_delay.Length.hist()
thinkplot.Config(xlabel="Delay In Mins", ylabel="Count")
```



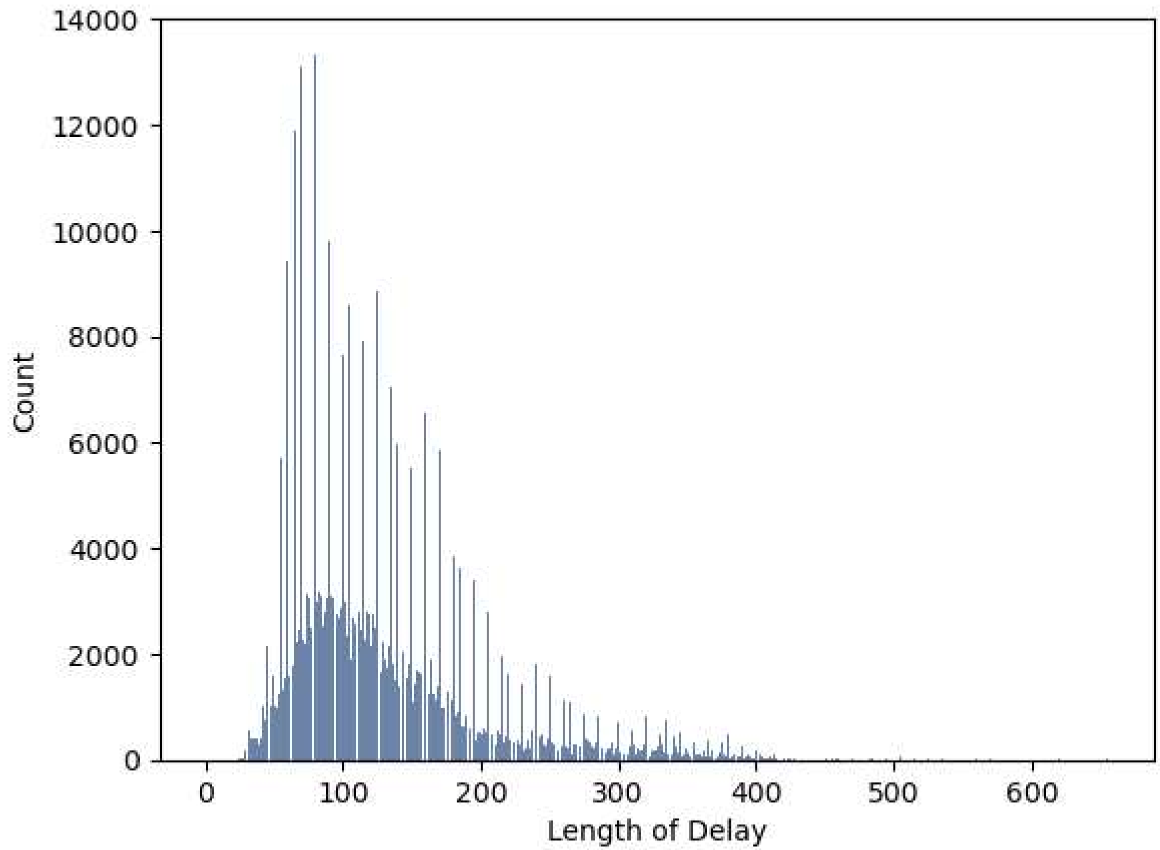
In [301]:

```
hist2 = Airline_delay.DayOfWeek.hist()
thinkplot.Config(xlabel="Day of Week Delay", ylabel="Count")
```



In [302]:

```
hist = thinkstats2.Hist(Airline_delay.Length, label="Ailrine Delays")
thinkplot.Hist(hist)
thinkplot.Config(xlabel="Length of Delay", ylabel="Count")
```

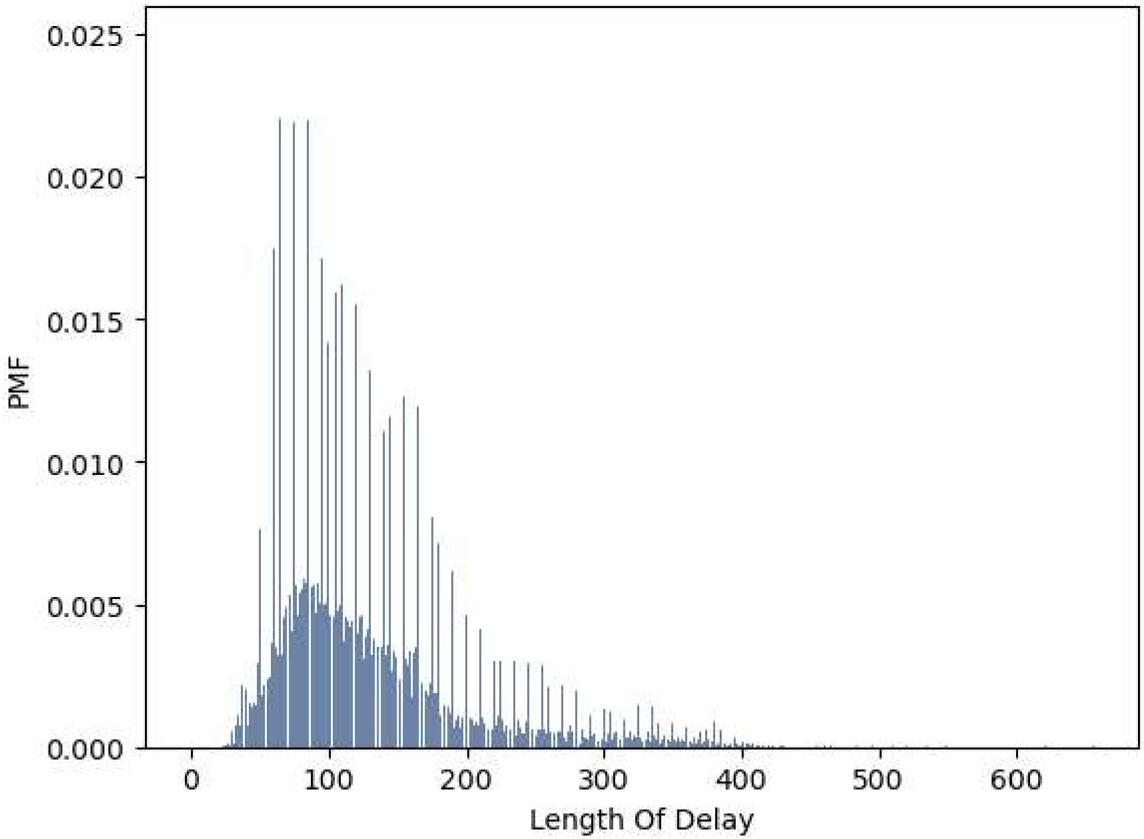


In [303]:

```
n = hist.Total()
pmf = hist.Copy()
for x, freq in hist.Items():
    pmf[x] = freq / n
```

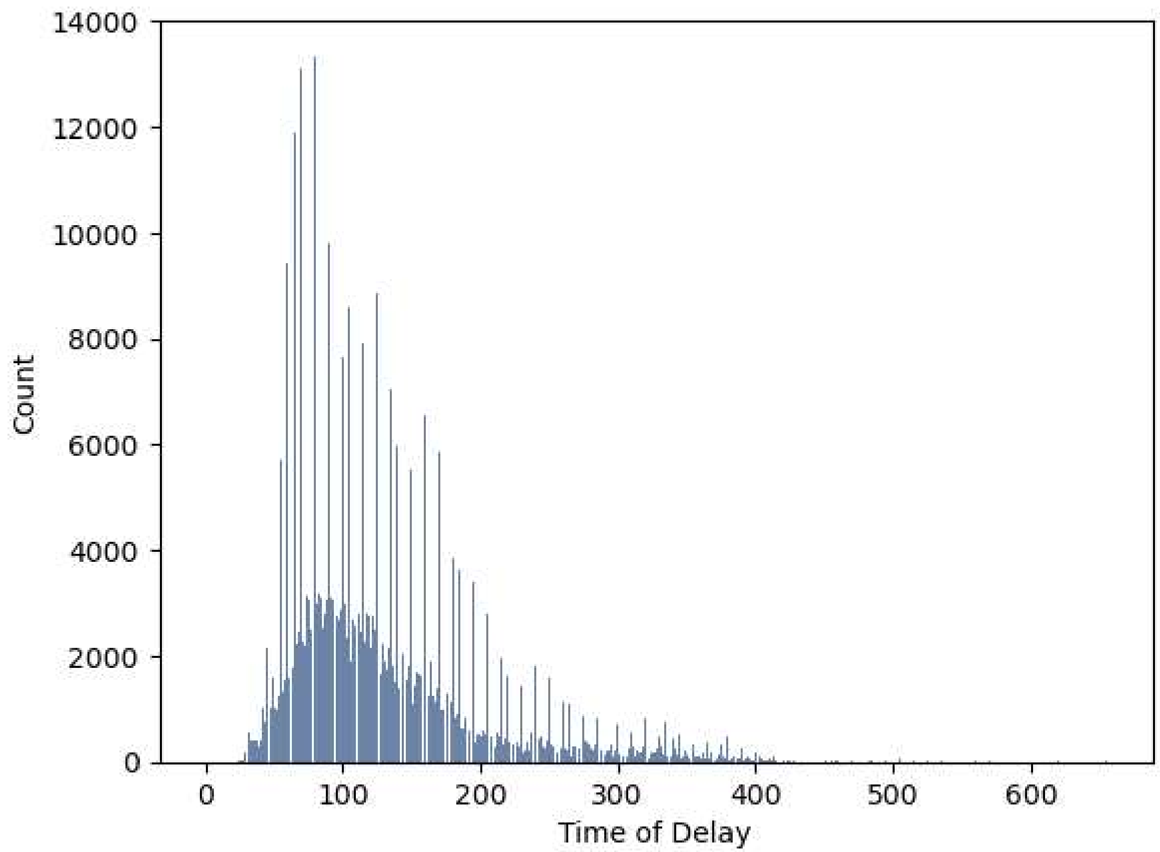
In [304]:

```
thinkplot.Hist(pmf)
thinkplot.Config(xlabel="Length Of Delay", ylabel="PMF")
```



In [305]:

```
hist2 = thinkstats2.Hist(Airline_delay.Time, label="Airline Delays")
thinkplot.Hist(hist)
thinkplot.Config(xlabel="Time of Delay", ylabel="Count")
```

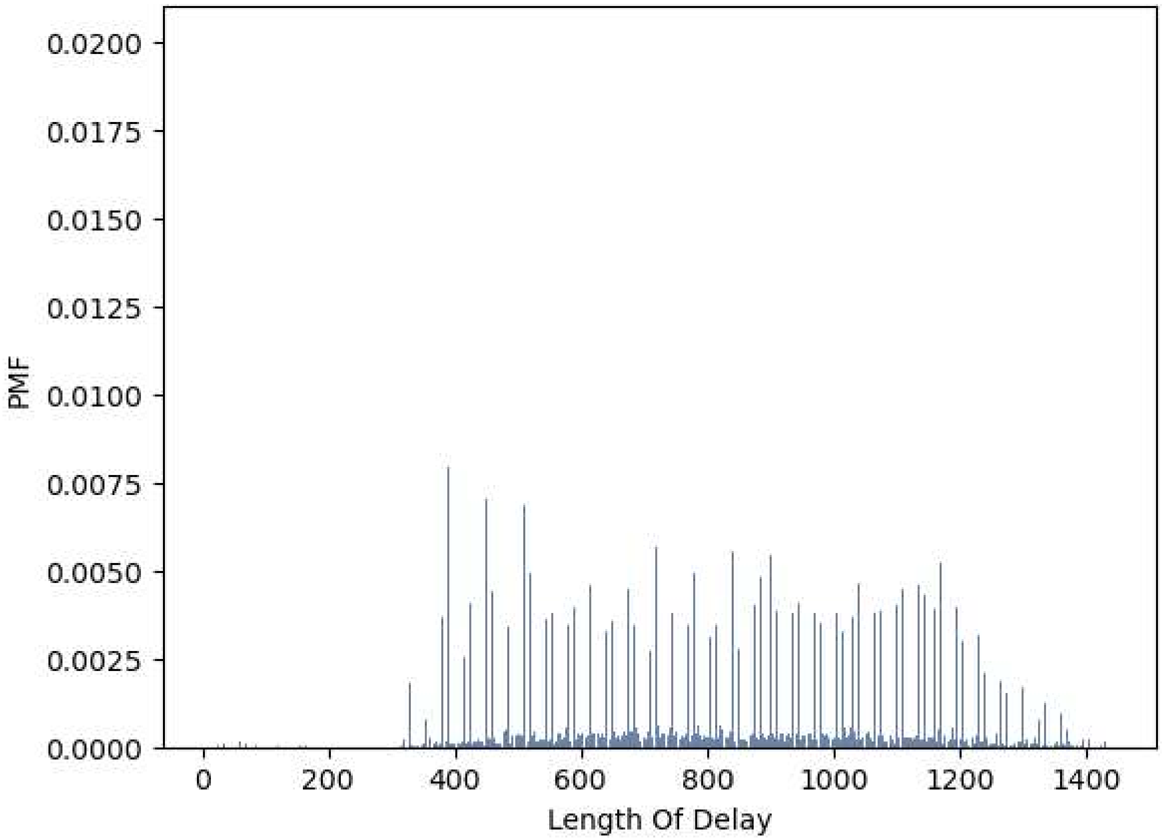


In [306]:

```
n2 = hist2.Total()
pmf2 = hist2.Copy()
for x, freq in hist2.Items():
    pmf2[x] = freq / n
```

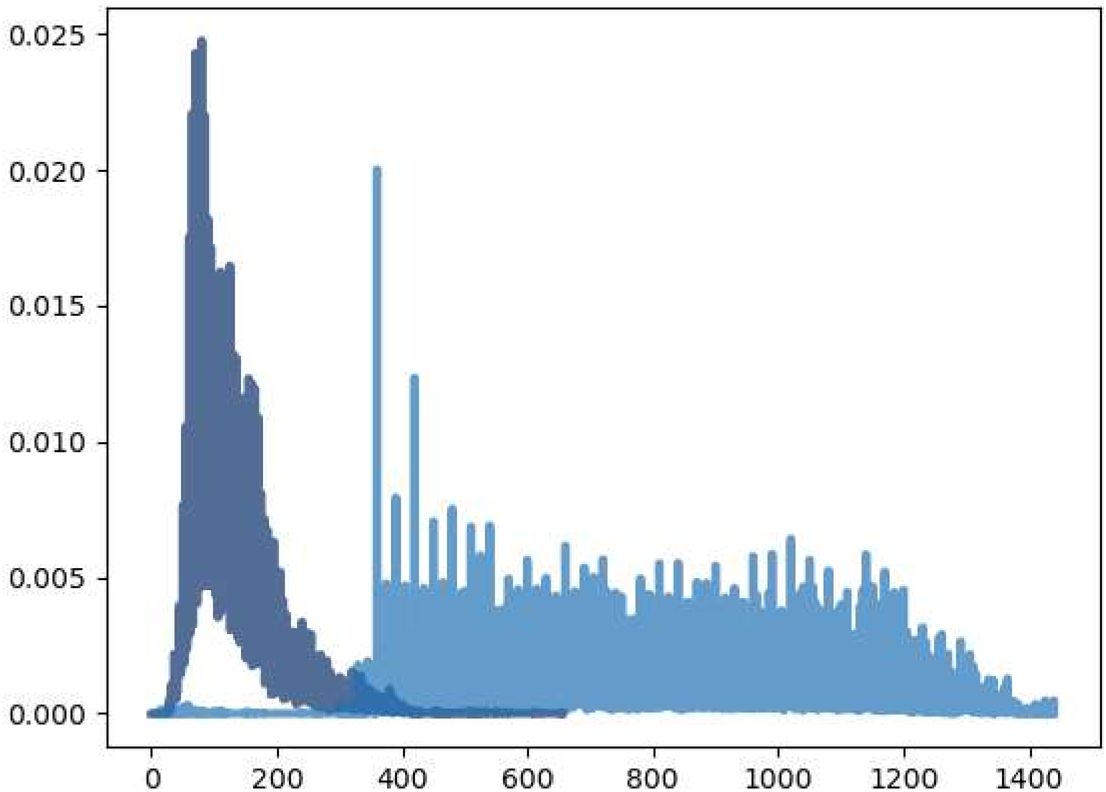
In [307]:

```
thinkplot.Hist(pmf2)
thinkplot.Config(xlabel="Length Of Delay", ylabel="PMF")
```



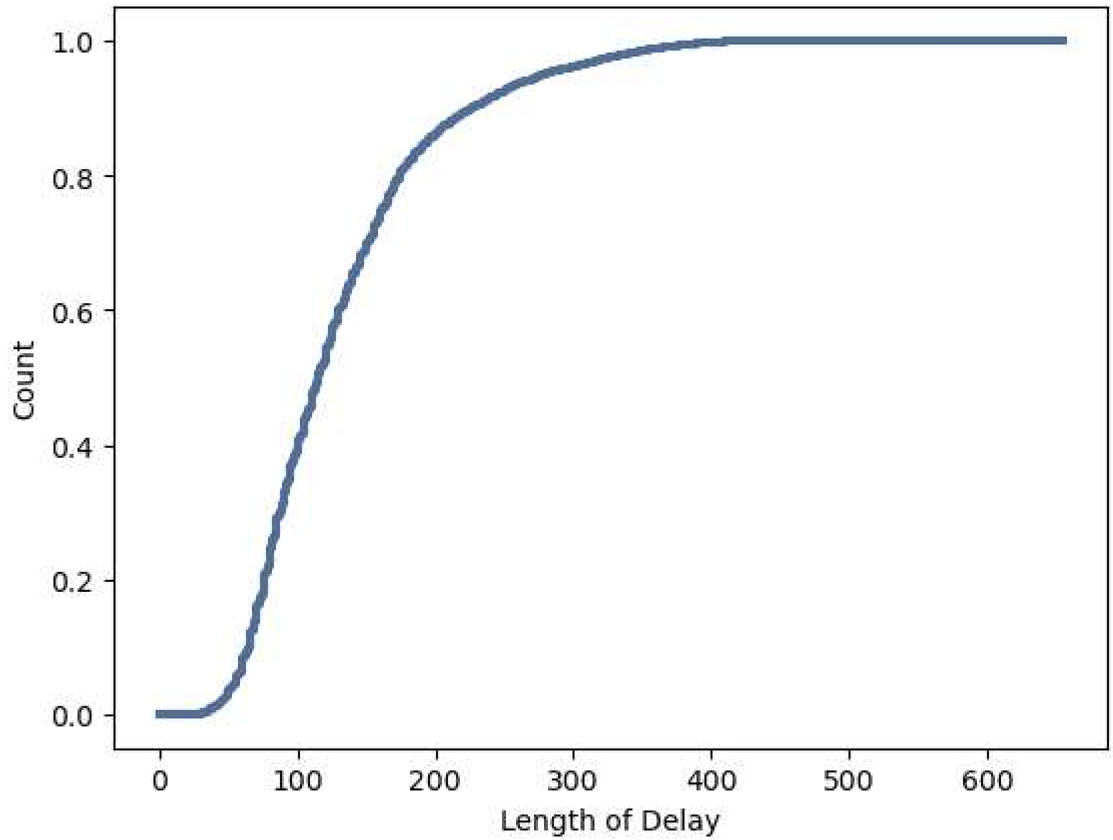
In [308]:

```
thinkplot.Pmfs([pmf, pmf2])
```

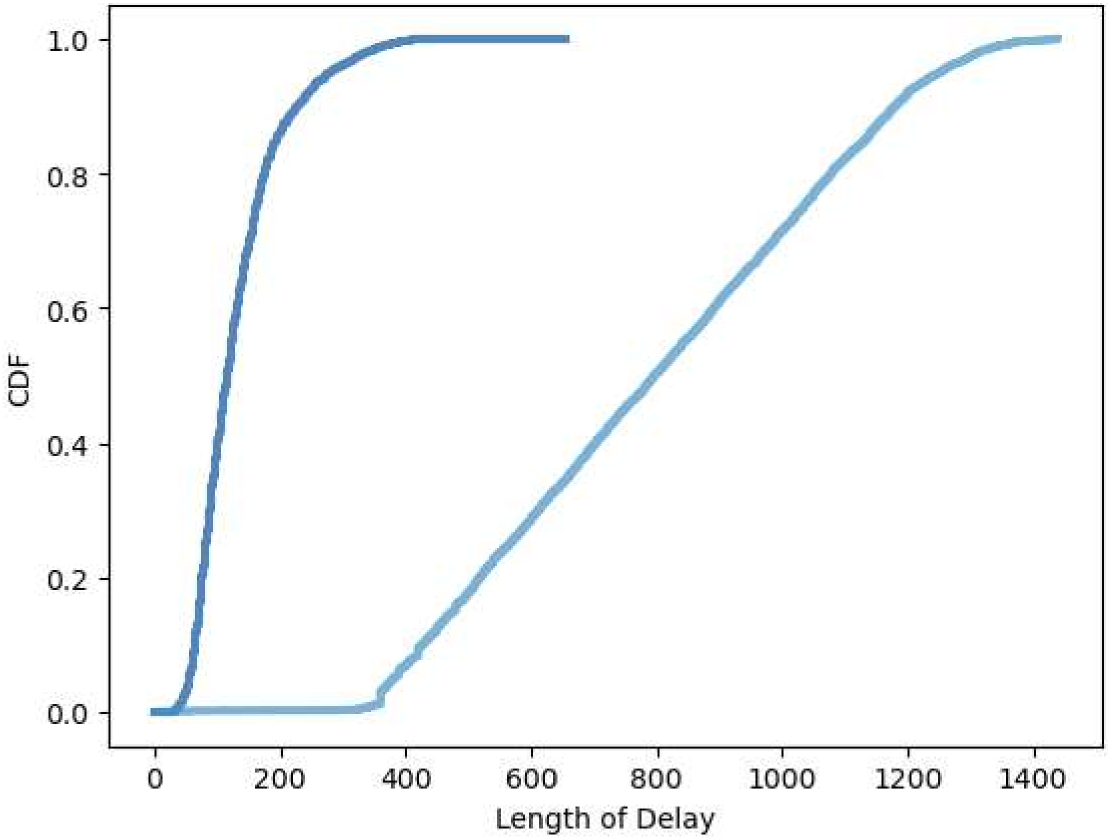


In [309]:

```
cdf = thinkstats2.Cdf(Airline_delay.Length, label="Ailrine Delays")
thinkplot.Cdf(cdf)
thinkplot.Config(xlabel="Length of Delay", ylabel="Count")
```

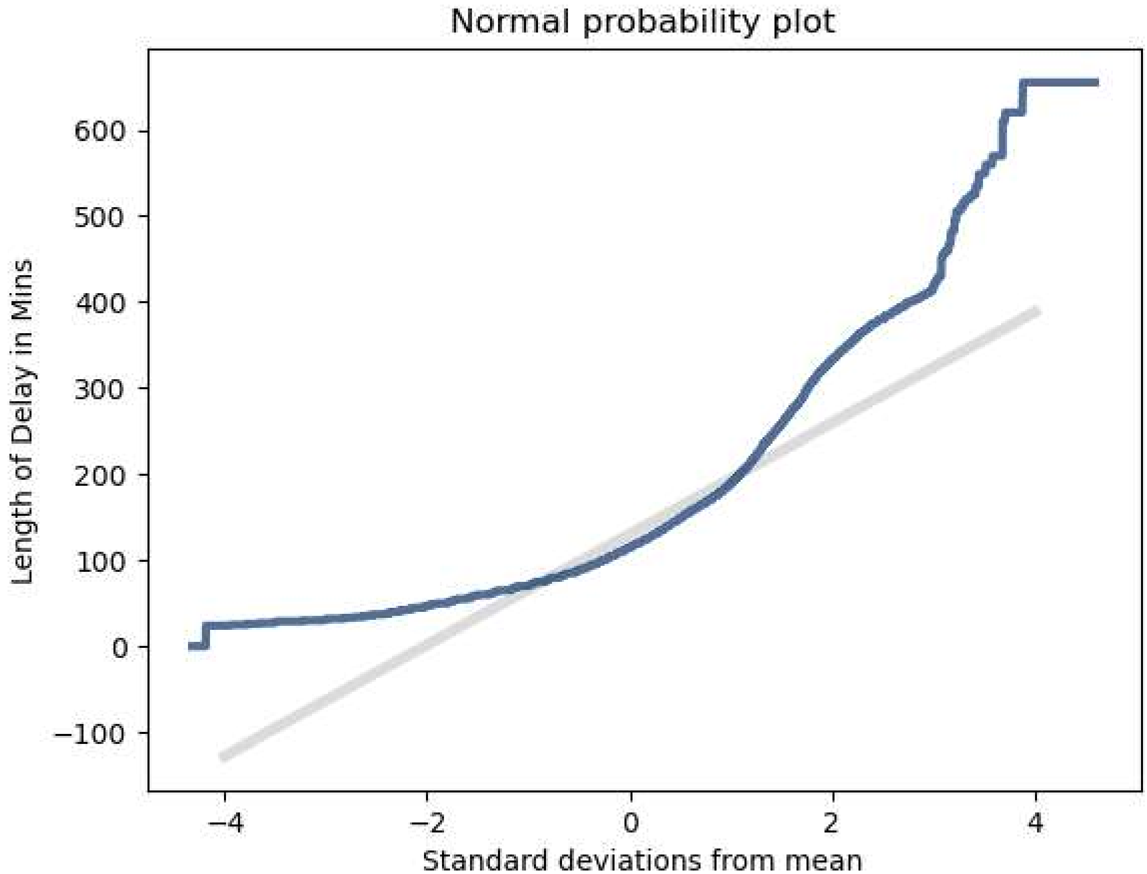


In [310]:
```
#The CDF below tells us that the early the flight on the 24hr clock, the less likeley
#someone is will experience a flight delay. Also, the early the fight the shorter the delay
#overall in terms of min.


first_cdf = thinkstats2.Cdf(Airline_delay.Length, label='Length')
other_cdf = thinkstats2.Cdf(Airline_delay.Time, label='Time Of Delay')

thinkplot.PrePlot(2)
thinkplot.Cdfs([first_cdf, other_cdf])
thinkplot.Config(xlabel='Length of Delay', ylabel='CDF')
```



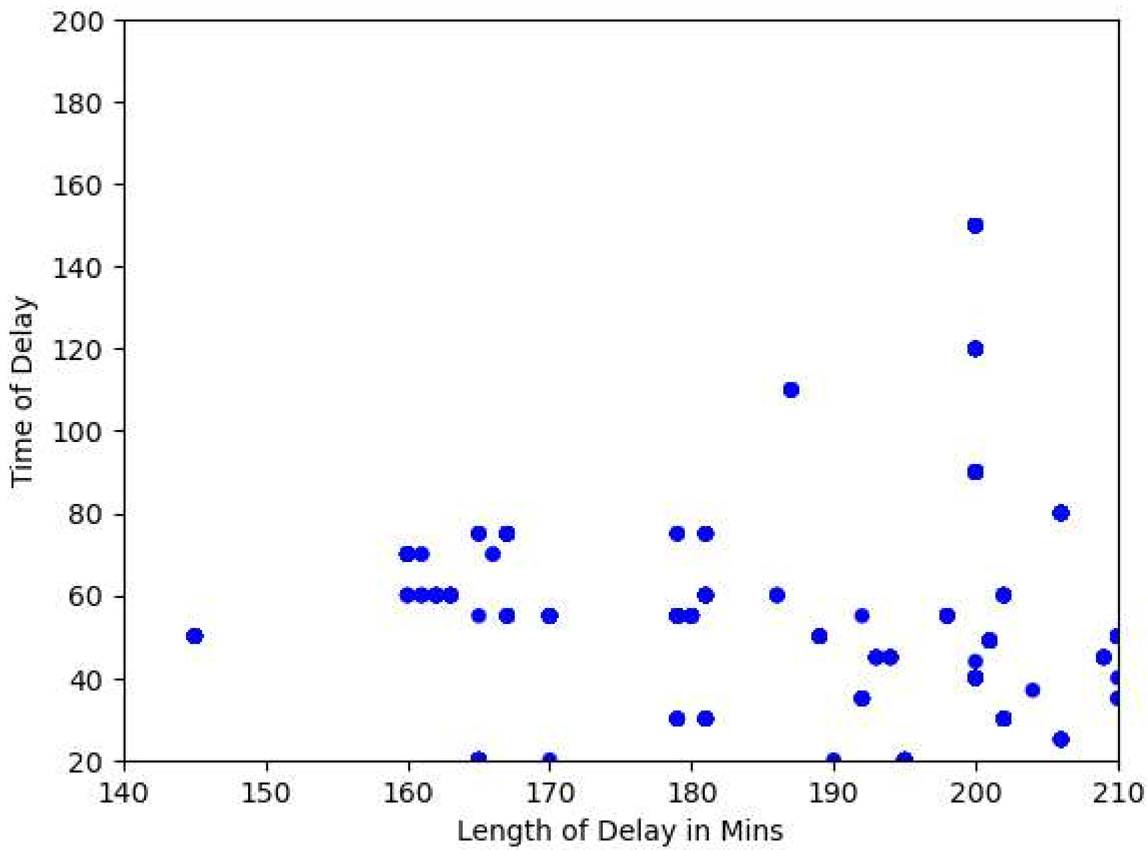
In [311]:

```
#I decided to use normal probability plot distribution. I felt this was the best model to use
#because we can see the amount of delays that deviate from the mean in a significant way.
#There are a fa amount of delays that fall on or slightly below when we look at the far left
#and far right of the graph. At the furthest left we can see the largest deviation from the mean.

mean, var = thinkstats2.TrimmedMeanVar(Airline_delay.Length, p=0.01)
std = np.sqrt(var)

xs = [-4, 4]
fxs, fys = thinkstats2.FitLine(xs, mean, std)
thinkplot.Plot(fxs, fys, linewidth=4, color="0.8")

xs, ys = thinkstats2.NormalProbability(Airline_delay.Length)
thinkplot.Plot(xs, ys, label="Flight Delays")

thinkplot.Config(
    title="Normal probability plot",
    xlabel="Standard deviations from mean",
    ylabel="Length of Delay in Mins",
)
```


Normal probability plot
Length of Delay in Mins
Standard deviations from mean
600
500
400
300
200
100
0
−100
−4
−2
0
2
4

In [312]:

```
thinkplot.Scatter(Airline_delay.Length, Airline_delay.Time, alpha=1)
thinkplot.Config(xlabel="Length of Delay in Mins",
                 ylabel='Time of Delay',
                 axis=[140, 210, 20, 200],
                 legend=False)
```



In [313]:

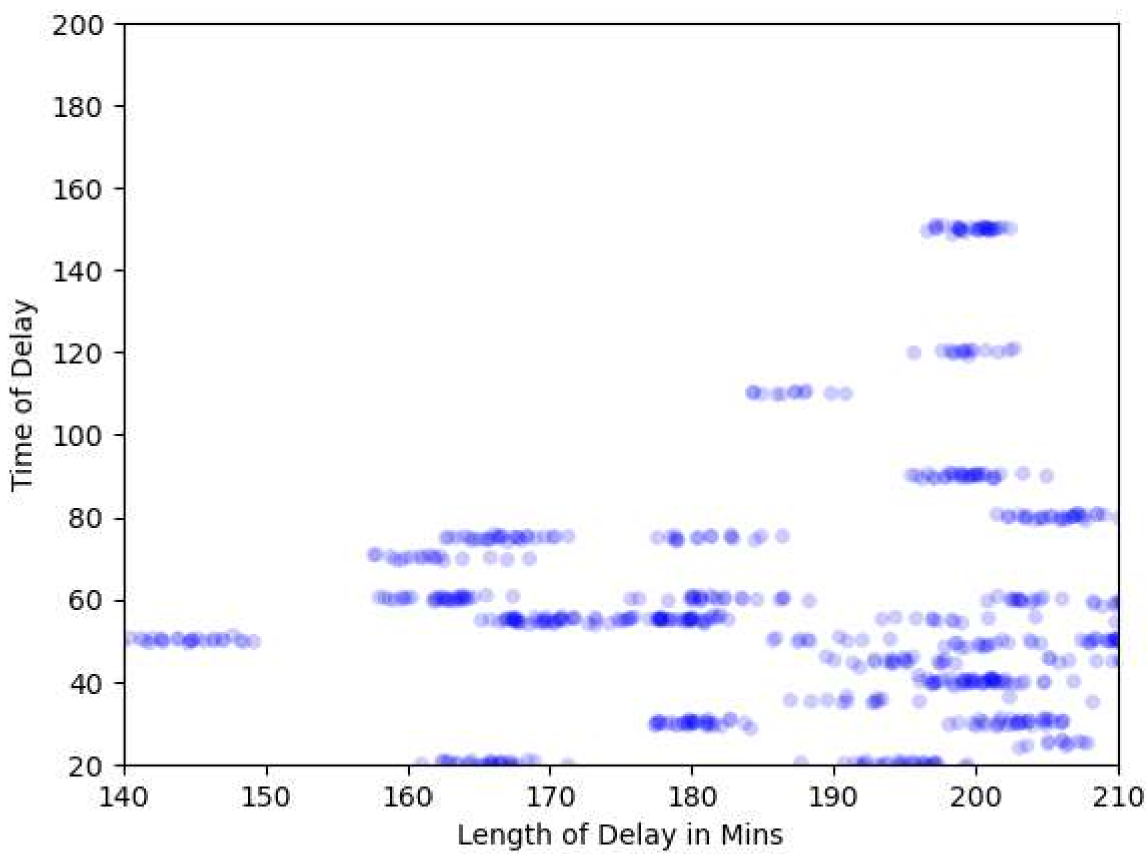
```
def Jitter(values, jitter=0.5):
    n = len(values)
    return np.random.normal(0, jitter, n) + values
```

In [314]:

```
Airline_delay.Length = Jitter(Airline_delay.Length, 2.0)
Airline_delay.Time = Jitter(Airline_delay.Time, 0.5)
```

In [315]:

```
#From the scatterplots, we can correlate the length of the dealy and the time being linear in nature
#we can see the correlation between time and lenght of delay have covariance becuase they vary
#together.

thinkplot.Scatter(Airline_delay.Length, Airline_delay.Time, alpha=0.2)
thinkplot.Config(xlabel='Length of Delay in Mins',
                 ylabel='Time of Delay',
                 axis=[140, 210, 20, 200],
                 legend=False)
```



In [316]:
```
class CorrelationPermute(thinkstats2.HypothesisTest):

    def TestStatistic(self, data):
        xs, ys = data
        test_stat = abs(thinkstats2.Corr(xs, ys))
        return test_stat

    def RunModel(self):
        xs, ys = self.data
        xs = np.random.permutation(xs)
        return xs, ys
```

In [317]:
```
import scipy.stats
```

In [318]:
```
scipy.stats.pearsonr(Airline_delay.Length, Airline_delay.Time)[0]    # Pearson's r
```

Out[318]:
```
-0.020678141667902366
```

In [319]:
```
scipy.stats.spearmanr(Airline_delay.Length, Airline_delay.Time)[0]   # Spearman's rho
```

Out[319]:
```
-0.04133274384402157
```

In [ ]:
```
#According to both Pearsons and Spearmans correlation coefficents, there is virtualy no linear
#relationship
```

In [346]:
```
Airline_delay.Length.head(10)
```

Out[346]:
```
0    204.703545
1    219.944415
2    167.789880
3    192.055376
4    202.519172
5    180.958850
6    222.943688
7    229.112258
8    216.660925
9    197.089766
Name: Length, dtype: float64
```

In [348]:
```
Airline_delay.Time.head(10)
```

Out[348]:
```
0    14.906158
1    15.264922
2    19.737849
3    19.329365
4    30.782883
5    30.235093
6    30.736221
7    30.136822
8    35.030230
9    39.662349
Name: Time, dtype: float64
```

In [406]:
```
#The following cells are showing a simple linear regression
#I first reshape the data into two seperate arays and assingned them to
#Variables dy1 and dy2.

dy1 = np.array(Airline_delay.Length).reshape((-1, 1))
dy2 = np.array(Airline_delay.Time).reshape((-1, 1))
```

In [407]:

```
dy1
```

Out[407]:

```
array([[204.70354516],
       [219.94441455],
       [167.78987977],
       ...,
       [257.94956239],
       [312.8424215 ],
       [298.50254687]])
```

In [408]:

```
dy2
```

Out[408]:

```
array([[  14.9061581 ],
       [  15.26492198],
       [  19.7378492 ],
       ...,
       [1438.59744448],
       [1439.02522239],
       [1438.04256716]])
```

In [409]:

```
#I fit the model to the data using the new arrays for dy1 and dy2

model = LinearRegression().fit(dy1, dy2)
```

In [414]:

```
r_sq = model.score(dy1, dy2)
print(f"coefficent of determination:{dy1+dy2}")
```

```
coefficent of determination:[[ 219.60970326]
 [ 235.20933653]
 [ 187.52772897]
 ...
 [1696.54700686]
 [1751.86764389]
 [1736.54511403]]
```

In [415]:
```
print(f"intercept: {model.intercept_}")
```

```
intercept: [813.56521706]
```

In [416]:
```
print(f"slope: {model.coef_}")
```

```
slope: [[-0.08196476]]
```

In [421]:
```
y_pred = model.predict(dy1 + dy2)
print(f"predicted response:\n{y_pred}")
```

```
predicted response:
[[795.56496041]
 [794.28634021]
 [798.19455174]
 ...
 [674.50814862]
 [669.97380588]
 [671.22971336]]
```

In [423]:
```
y_pred = model.intercept_ + model.coef_ * x
print(f"predicted response:\n{y_pred}")
```

```
predicted response:
[[810.53252093]]
```

In [424]:
```
x_new = np.arange(5).reshape((-1, 1))
x_new
```

Out[424]:
```
array([[0],
       [1],
       [2],
       [3],
       [4]])
```

In [425]:

```
y_new = model.predict(x_new)
y_new
```

Out[425]:

```
array([[813.56521706],
       [813.4832523 ],
       [813.40128754],
       [813.31932278],
       [813.23735802]])
```

In [ ]:

```
#Summary:

#Overall, the outcome of the EDA covering flight delays for 2018 shows several key
#predictive factors and delay information that could be useful for travelers who are
#looking to avoid delays. The variables show the likelihood of a delay being correlated
#to the flight time and the length of the delay sharing some level of
#relationship. Other variables that could have assisted in the analysis are the airport to and from
#the delay reported. The origin and destination of the delay could have played a critical factor in
#the delays. We know some airports are far busier than others which may result in delays
#from a traffic and passenger management perspective. Another variable that could have helped would be
# the time of year or season the delay was reported in when it occurred. Winter traveling is notorious fo
# experiencing delays due to weather conditions that make flying unsafe or impossible. One assumption
#I felt needed to be corrected was the honesty of the airlines reporting their delays promptly and accura
#The dataset does not consider industry standards or airline-specific practices for delays and time
#reporting. The section on the PMF was difficult because I needed to figure out how to run two different
#on the same variable. Instead, I chose two similar variables, time of delay vs. length.
#The last section for linear regression took some extra effort because I was working with a dataset that
#I chose, and I have been getting used to working with the datasets provided by the book. However,
#I was happy to get the chance to work with a dataset I had an interest in and an industry I worked in
#for a long time. Working with the delay information was insightful, and I learned a lot about the approa
#needed and methods used to present an overall picture of the data.
```