This is a PDF report of the 'ST2195' module's Coursework Project

In this report we are analysing the flight arrival and departure details for all commercial flights on major carriers within the USA from 2006 to 2007.

This dataset was completed by the 2009 ASA Statistical Computing and Graphics Data Expo, and can be downloaded from the Harvard Dataverse website.

A subset of two consecutive years was selected to complete this report and answer specific questions.

Before we start looking at the questions, let's take a look at the data we are dealing with

In [3]:
```
first_year
```

Out[3]:

	Year	Month	DayofMonth	DayOfWeek	DepTime	CRSDepTime	ArrTime	CRSArrTi
0	2006	1	11	3	743.0	745	1024.0	1
1	2006	1	11	3	1053.0	1053	1313.0	1
2	2006	1	11	3	1915.0	1915	2110.0	2
3	2006	1	11	3	1753.0	1755	1925.0	19
4	2006	1	11	3	824.0	832	1015.0	1
...	...	...	...	...	...	...	...	
7141917	2006	12	29	5	1246.0	1249	1452.0	14
7141918	2006	12	29	5	1225.0	1155	2033.0	1
7141919	2006	12	29	5	2118.0	2115	2254.0	2
7141920	2006	12	29	5	2122.0	2127	2209.0	21
7141921	2006	12	29	5	2042.0	2045	2213.0	21

7141922 rows × 29 columns

In [4]:
```
second_year
```

Out[4]:

	Year	Month	DayofMonth	DayOfWeek	DepTime	CRSDepTime	ArrTime	CRSArrT
0	2007	1	1	1	1232.0	1225	1341.0	1
1	2007	1	1	1	1918.0	1905	2043.0	2
2	2007	1	1	1	2206.0	2130	2334.0	2
3	2007	1	1	1	1230.0	1200	1356.0	1
4	2007	1	1	1	831.0	830	957.0	1
...	...	...	...	...	...	...	...	
7453210	2007	12	15	6	1558.0	1605	1749.0	1
7453211	2007	12	15	6	1902.0	1851	2110.0	2
7453212	2007	12	15	6	1024.0	1025	1750.0	1
7453213	2007	12	15	6	1353.0	1315	1658.0	1
7453214	2007	12	15	6	1824.0	1800	2001.0	1

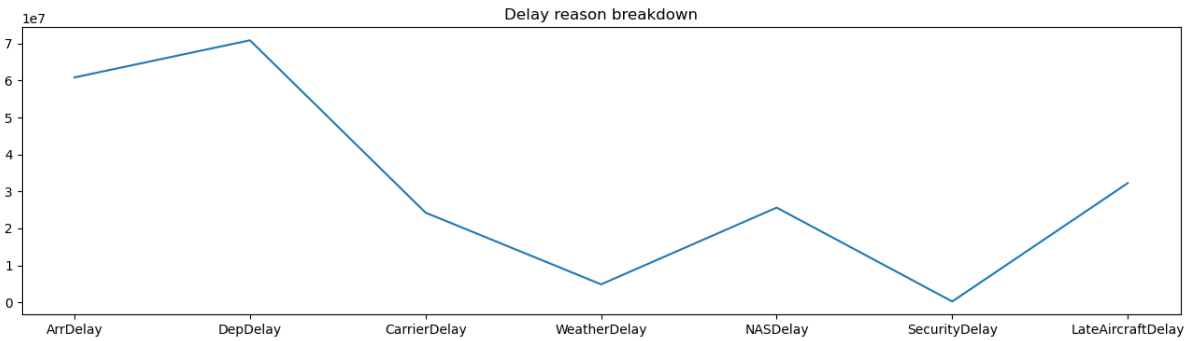
7453215 rows × 29 columns

For both years we have over seven million rows that depict each day of the year, and 29 columns that depict the different elements can be used for analysis.

As interesting bit, let's see the total delay in 2006 in minutes and the breakdown of the different delay reasons.

In [32]:
```
breakdown_delay_first_year.plot(figsize=(16,4), title="Delay reason breakdow
print(f"Total delay in the first year: {(total_delay_first_year)} minutes")
```

```
Total delay in the first year: 218857180.0 minutes
```



Question 1 - When is the best time of day, day of the week, and time of year to fly to minimise delays?

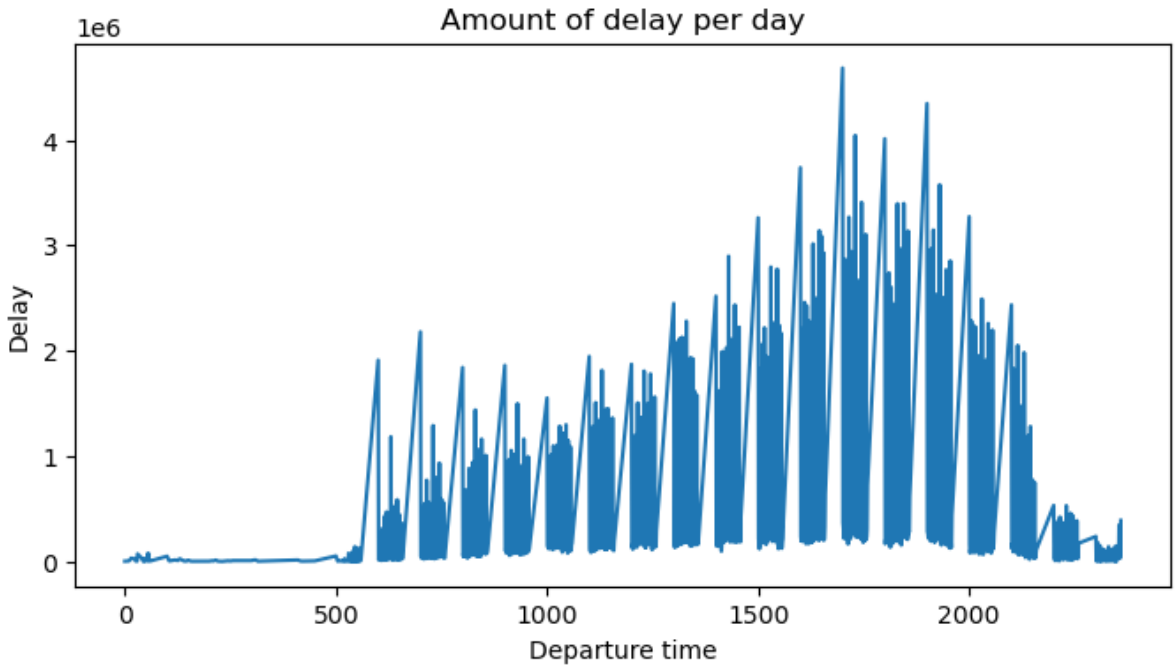
Note - For Questions 1, 2 it made sense to combine the two years to create the plots, as they were showing more or less the same results individually too.

Question 1 / part 1 - When is the best time of day to fly to minimise delays?

As the plot shows, the best time of day to minimise delays would be 5am to 12pm. The sum of delays is the highest between 5pm and 7pm. The very low number of delays between 12am and 5am can be attributed to the very low number of flights in that time period. Pilots are also more likely to be able to make up for delays / arrive earlier than the predicted time, after 9pm.

In [35]:
```
# Plot the graph with the relevant information
total_delay_day.plot(figsize = (8,4), xlabel='Departure time', ylabel='Delay
```

Out[35]:
```
<AxesSubplot:title={'center':'Amount of delay per day'}, xlabel='Departure t
ime', ylabel='Delay'>
```

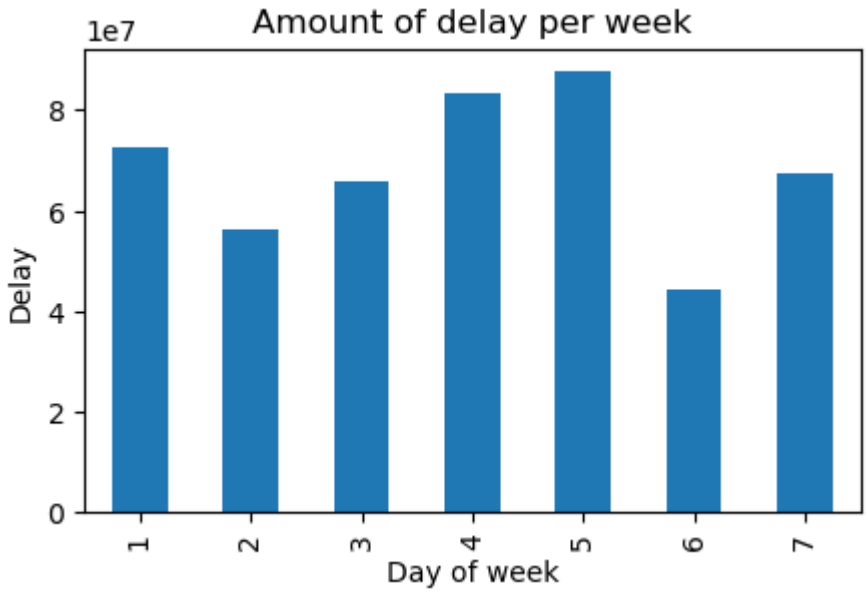


Question 1 / part 2 - The best day of the week to fly to minimise delays?

The best day to fly to minimise delays is Saturday, followed by Tuesday and Wednesday. Not flying on Thursdays and Fridays is advised to avoid delays.

In [37]:
```
# Plot the graph with the relevant information
total_delay_week.plot(kind = 'bar', xlabel='Day of week', ylabel='Delay', ti
```

Out[37]:
```
<AxesSubplot:title={'center':'Amount of delay per week'}, xlabel='Day of wee
k', ylabel='Delay'>
```

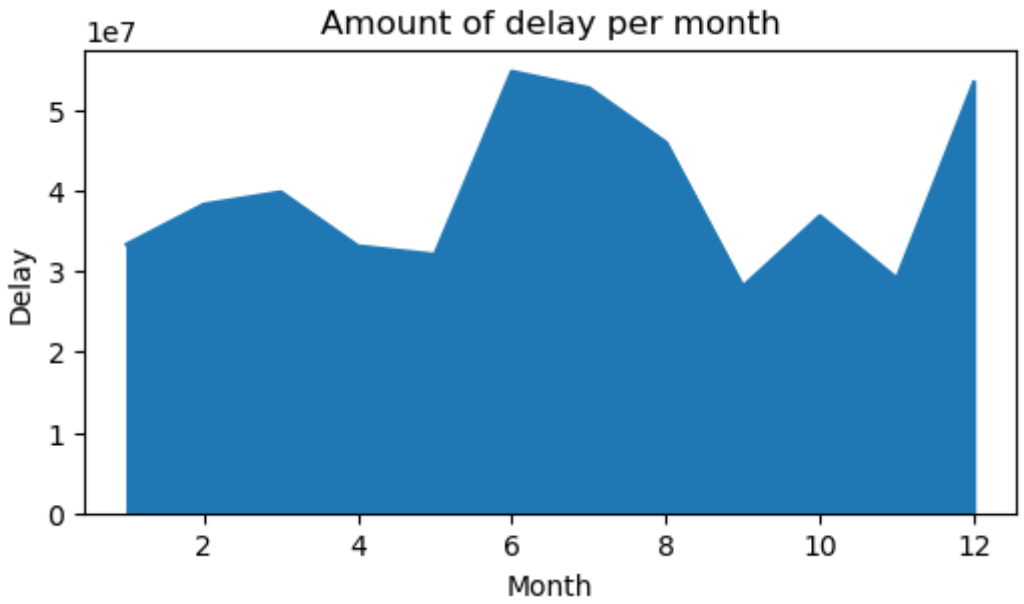


Question 1 / part 3 - The best time of year to fly to minimise delays?

We can see that the most likely months to avoid delays are September, November, May and January. The periods with the most delays are the summer and December.

In [40]:
```
# Plot the graph with the relevant information
total_delay_month.plot.area(xlabel='Month', ylabel='Delay', title="Amount of
```

Out[40]:
```
<AxesSubplot:title={'center':'Amount of delay per month'}, xlabel='Month', y
label='Delay'>
```



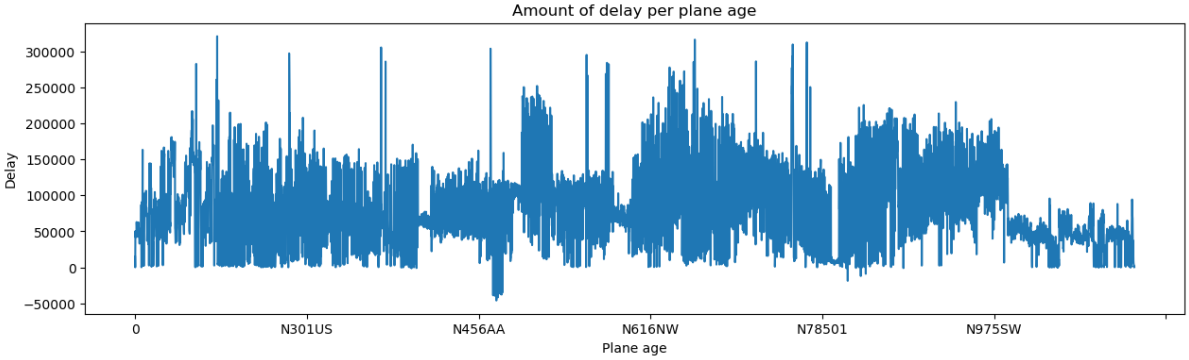
Question 2 - Do older planes suffer more delays?

Extrapolating from tail numbers corresponding to aircraft age in ascending order - smaller number older plane, higher number younger plane - we have no sufficient evidence to claim that older planes suffer more delays. There is, however, a drop in

delays with newer planes, although the number of flights is also lower, which is likely to cause the drop.

In [42]:
```
# Plot graph
total_delay_age.plot(figsize = (15,4), xlabel='Plane age', ylabel='Delay', t
```

Out[42]:
```
<AxesSubplot:title={'center':'Amount of delay per plane age'}, xlabel='Plane
age', ylabel='Delay'>
```



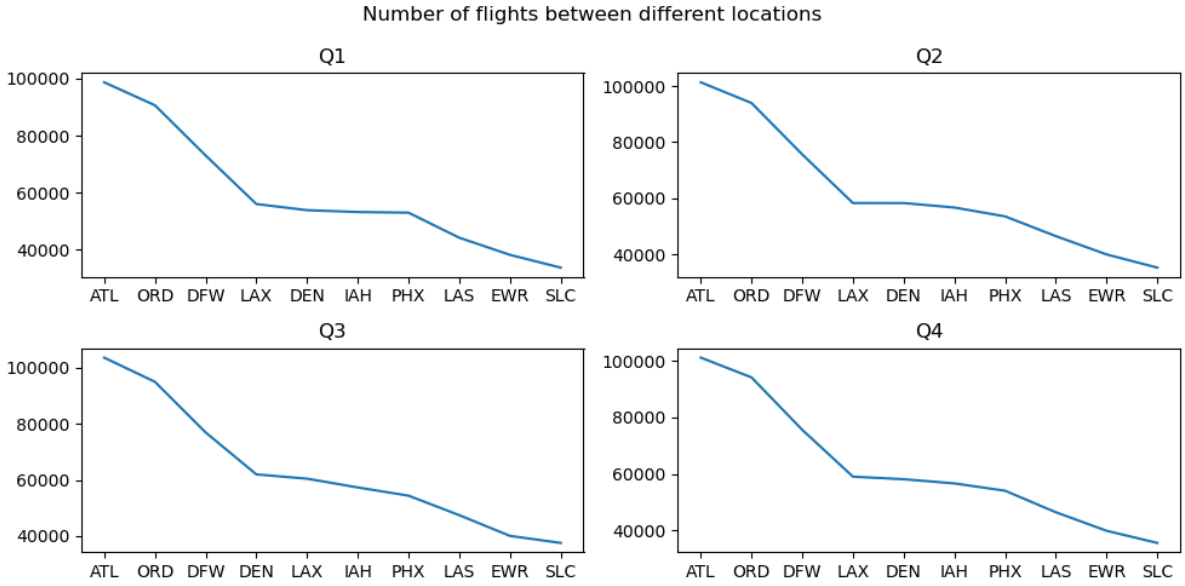
Question 3 - How does the number of people flying between different locations change over time?

The number of people flying between different locations is quite consistent. We look at the quarterly results of the top 10 destinations. The top 3 are Atlanta, GA, Chicago, IL O'Hare, Dallas, TX Dallas/Ft Worth Intl. In Q3 Denver, CO International takes the 4th place, in every other quarter Los Angeles, CA is the 4th most popluar destination. Salt Lake City, UT is the number 10 destination year-round.

We can also see that Atlanta, GA has the highest number of incoming flights in Q3, and the lowest in Q1.

In [44]:
```
fig, axs = plt.subplots(2,2, figsize=(10,5))
fig.set_xlabel=('Destination')
fig.set_ylabel=('Number of flights')
fig.suptitle('Number of flights between different locations')
axs[0,0].plot(q1)
axs[0,1].plot(q2)
axs[1,0].plot(q3)
axs[1,1].plot(q4)
for i in range(4):
    plt.subplot(2,2,i+1).set_title(f'Q{i+1}')

fig.tight_layout()
```



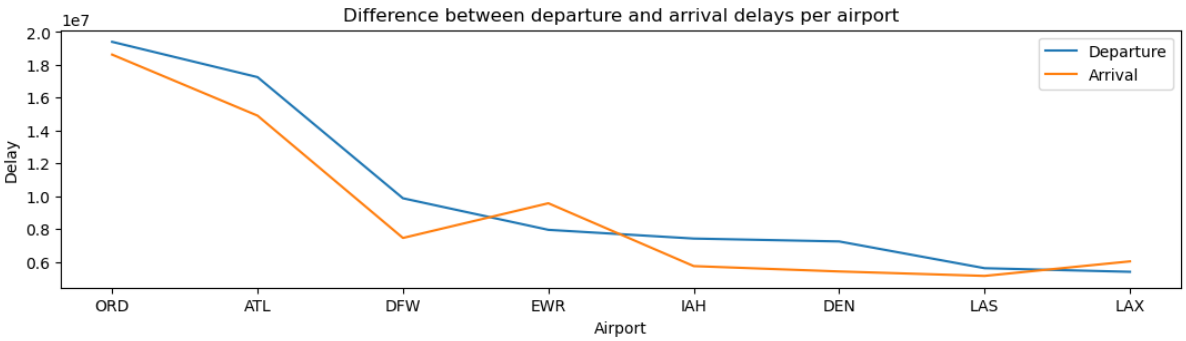
Question 4 - Can you detect cascading failures as delays in one airport create delays in others?

To answer this question, we take the top 10 airports and group them by departure delays and arrival delays. Then we plot the results of the two variables on a chart to see their relationship.

Departure and arrival delays seem to be happening on the same scale meaning that arrival delays contribute to departure delays, with the exception of Newark, NJ and Los Angeles, CA. Here they create the impression of being able to make up for the arrival delays.

In [30]:
```
# Create a dataframe from the with two variables - top 10 departure and arri
df1 = pd.concat([total_delay_from, total_delay_to], axis=1).dropna()
df1.plot(xlabel='Airport', ylabel='Delay', title='Difference between departu
plt.legend(('Departure','Arrival'))
```

Out[30]:
```
<matplotlib.legend.Legend at 0x7fe7d57682e0>
```



Question 5

Use the available variables to construct a model that predicts delays.

Model 1 - Trying to find the likelihood of weather delays on different days. We train the model to find the days when weather delays happen and flights are diverted or cancelled.

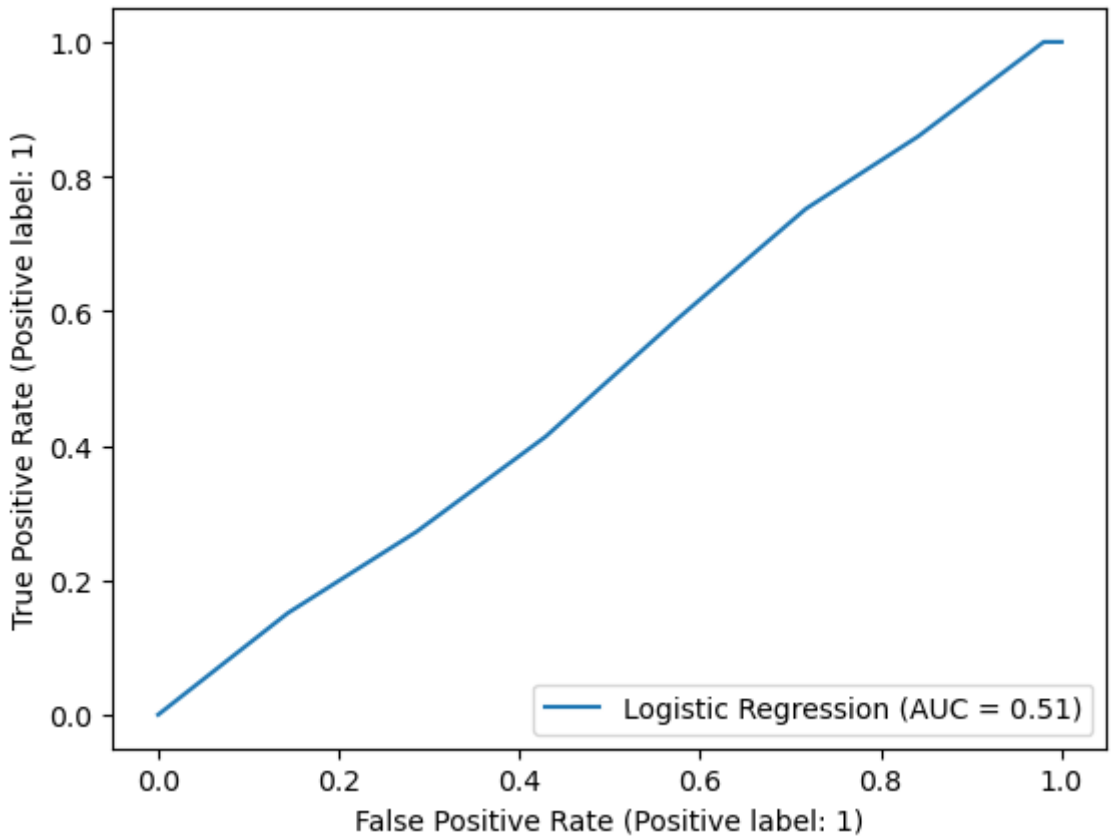
In [17]:
```
print(f'Training Accuracy - {(w_delay.score(X_train, y_train) * 100).round(2
```

```
Training Accuracy - 98.41%
```

In [18]:
```
ax = plt.gca()
plot_roc_curve(w_delay, X_test, y_test, ax=ax, name='Logistic Regression')
```

```
/Users/WorkAccount/opt/anaconda3/lib/python3.9/site-packages/sklearn/utils/d
eprecation.py:87: FutureWarning: Function plot_roc_curve is deprecated; Func
tion :func:`plot_roc_curve` is deprecated in 1.0 and will be removed in 1.2.
Use one of the class methods: :meth:`sklearn.metric.RocCurveDisplay.from_pre
dictions` or :meth:`sklearn.metric.RocCurveDisplay.from_estimator`.
  warnings.warn(msg, category=FutureWarning)
```

Out[18]:
```
<sklearn.metrics._plot.roc_curve.RocCurveDisplay at 0x7feccc882760>
```



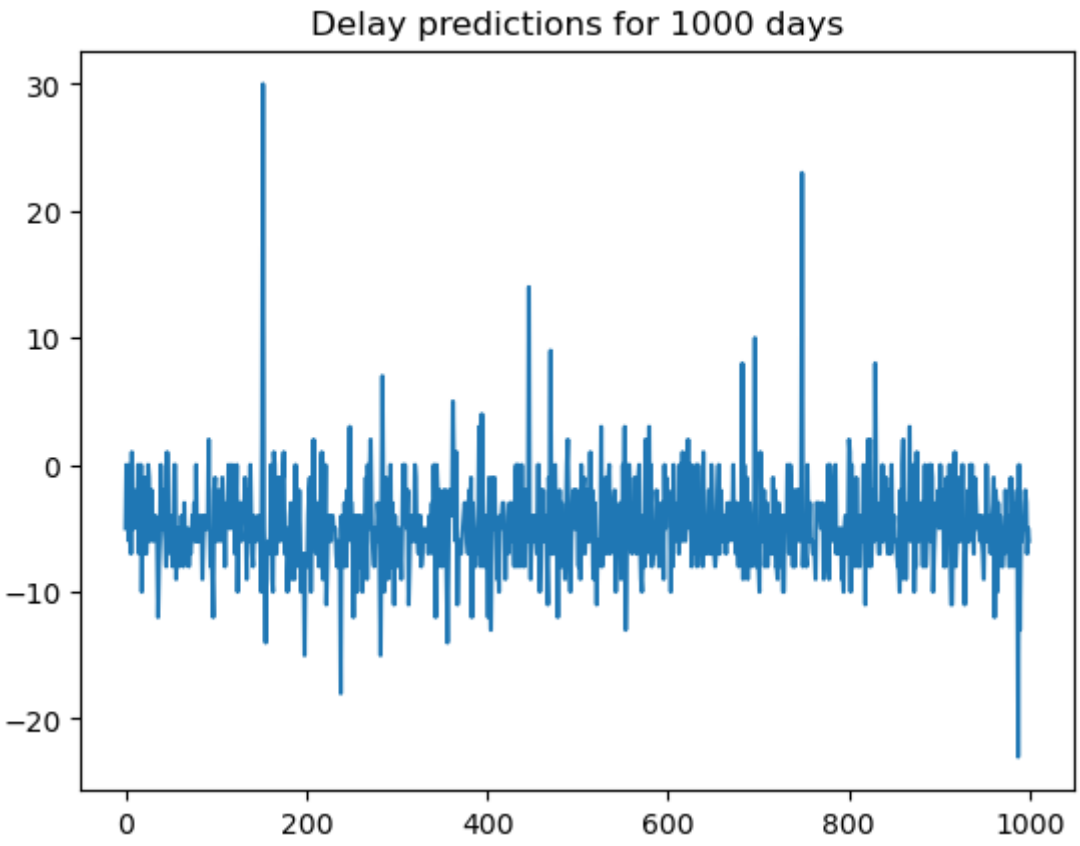
An AUC of 0.51 is not a great result for a test, therefore we can conclude that the test needs better alignment

Model 2 - We want to predict the arrival delays by looking at the total time it takes to flights to actually complete a route on different days.

In [25]:
```
print(f"Making a prediction of mean delay time for a period of 1000 days: wi
plt.plot(a_delay_model.predict(X_a_test[1000:2000]))
plt.title("Delay predictions for 1000 days")
```

```
Making a prediction of mean delay time for a period of 1000 days: with 3.28%
accuracy
```

Out[25]: `Text(0.5, 1.0, 'Delay predictions for 1000 days')`



An accuracy of 3.28% does not provide a reliable prediction. Further adjustments are needed for the model to gain better predictions.