In [1]:
```
import pandas as pd
dframe = pd.read_csv("aapl.csv")
```

In [2]:
```
dframe
```

Out[2]:

	Date	Open	High	Low	Close	Volume
0	7-Jul-17	142.90	144.75	142.90	144.18	19201712
1	6-Jul-17	143.02	143.50	142.41	142.73	24128782
2	5-Jul-17	143.69	144.79	142.72	144.09	21569557
3	3-Jul-17	144.88	145.30	143.10	143.50	14277848
4	30-Jun-17	144.45	144.96	143.78	144.02	23024107
...	...	...	...	...	...	...
246	15-Jul-16	98.92	99.30	98.50	98.78	30136990
247	14-Jul-16	97.39	98.99	97.32	98.79	38918997
248	13-Jul-16	97.41	97.67	96.84	96.87	25892171
249	12-Jul-16	97.17	97.70	97.12	97.42	24167463
250	11-Jul-16	96.75	97.65	96.73	96.98	23794945

251 rows × 6 columns

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

Out[3]:

	Date	Open	High	Low	Close	Volume
0	7-Jul-17	142.90	144.75	142.90	144.18	19201712
1	6-Jul-17	143.02	143.50	142.41	142.73	24128782
2	5-Jul-17	143.69	144.79	142.72	144.09	21569557
3	3-Jul-17	144.88	145.30	143.10	143.50	14277848
4	30-Jun-17	144.45	144.96	143.78	144.02	23024107

In [5]:
```
dframe.isna().sum()
```

Out[5]:
```
Date      0
Open      0
High      0
Low       0
Close     0
Volume    0
dtype: int64
```

In [6]:
```
dframe.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 251 entries, 0 to 250
Data columns (total 6 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   Date    251 non-null    object 
 1   Open    251 non-null    float64
 2   High    251 non-null    float64
 3   Low     251 non-null    float64
 4   Close   251 non-null    float64
 5   Volume  251 non-null    int64  
dtypes: float64(4), int64(1), object(1)
memory usage: 11.9+ KB
```

In [7]:
```
dframe = pd.read_csv("aapl.csv", parse_dates=["Date"])
```

```
C:\Users\bhanu\AppData\Local\Temp\ipykernel_33916\3836614857.py:1: UserWarning: Could not infer format, so each element will be parsed individually, falling back to `dateutil`. To ensure parsing is consistent and as-expected, please specify a format.
  dframe = pd.read_csv("aapl.csv", parse_dates=["Date"])
```

In [8]:
```
dframe
```

Out[8]:

	Date	Open	High	Low	Close	Volume
0	2017-07-07	142.90	144.75	142.90	144.18	19201712
1	2017-07-06	143.02	143.50	142.41	142.73	24128782
2	2017-07-05	143.69	144.79	142.72	144.09	21569557
3	2017-07-03	144.88	145.30	143.10	143.50	14277848
4	2017-06-30	144.45	144.96	143.78	144.02	23024107
...	...	...	...	...	...	...
246	2016-07-15	98.92	99.30	98.50	98.78	30136990
247	2016-07-14	97.39	98.99	97.32	98.79	38918997
248	2016-07-13	97.41	97.67	96.84	96.87	25892171
249	2016-07-12	97.17	97.70	97.12	97.42	24167463
250	2016-07-11	96.75	97.65	96.73	96.98	23794945

251 rows × 6 columns

In [9]:
```
dframe = pd.read_csv("aapl.csv", parse_dates=["Date"], index_col="Date")
```

```
C:\Users\bhanu\AppData\Local\Temp\ipykernel_33916\1913338630.py:1: UserWarning: Could not infer format, so each element will be parsed individually, falling back to `dateutil`. To ensure parsing is consistent and as-expected, please specify a format.
  dframe = pd.read_csv("aapl.csv", parse_dates=["Date"], index_col="Date")
```

In [10]:
```
dframe
```

Out[10]:

Date	Open	High	Low	Close	Volume
2017-07-07	142.90	144.75	142.90	144.18	19201712
2017-07-06	143.02	143.50	142.41	142.73	24128782
2017-07-05	143.69	144.79	142.72	144.09	21569557
2017-07-03	144.88	145.30	143.10	143.50	14277848
2017-06-30	144.45	144.96	143.78	144.02	23024107
...	...	...	...	...	...
2016-07-15	98.92	99.30	98.50	98.78	30136990
2016-07-14	97.39	98.99	97.32	98.79	38918997
2016-07-13	97.41	97.67	96.84	96.87	25892171
2016-07-12	97.17	97.70	97.12	97.42	24167463
2016-07-11	96.75	97.65	96.73	96.98	23794945

251 rows × 5 columns

In [11]:
```
dframe.loc["2017-01"]
```

Out[11]:

Date	Open	High	Low	Close	Volume
2017-01-31	121.15	121.39	120.62	121.35	49200993
2017-01-30	120.93	121.63	120.66	121.63	30377503
2017-01-27	122.14	122.35	121.60	121.95	20562944
2017-01-26	121.67	122.44	121.60	121.94	26337576
2017-01-25	120.42	122.10	120.28	121.88	32586673
2017-01-24	119.55	120.10	119.50	119.97	23211038
2017-01-23	120.00	120.81	119.77	120.08	22050218
2017-01-20	120.45	120.45	119.73	120.00	32597892
2017-01-19	119.40	120.09	119.37	119.78	25597291
2017-01-18	120.00	120.50	119.71	119.99	23712961
2017-01-17	118.34	120.24	118.22	120.00	34439843
2017-01-13	119.11	119.62	118.81	119.04	26111948
2017-01-12	118.90	119.30	118.21	119.25	27086220
2017-01-11	118.74	119.93	118.60	119.75	27588593
2017-01-10	118.77	119.38	118.30	119.11	24462051
2017-01-09	117.95	119.43	117.94	118.99	33561948
2017-01-06	116.78	118.16	116.47	117.91	31751900
2017-01-05	115.92	116.86	115.81	116.61	22193587
2017-01-04	115.85	116.51	115.75	116.02	21118116
2017-01-03	115.80	116.33	114.76	116.15	28781865

In [12]:
```
dframe.Close
```

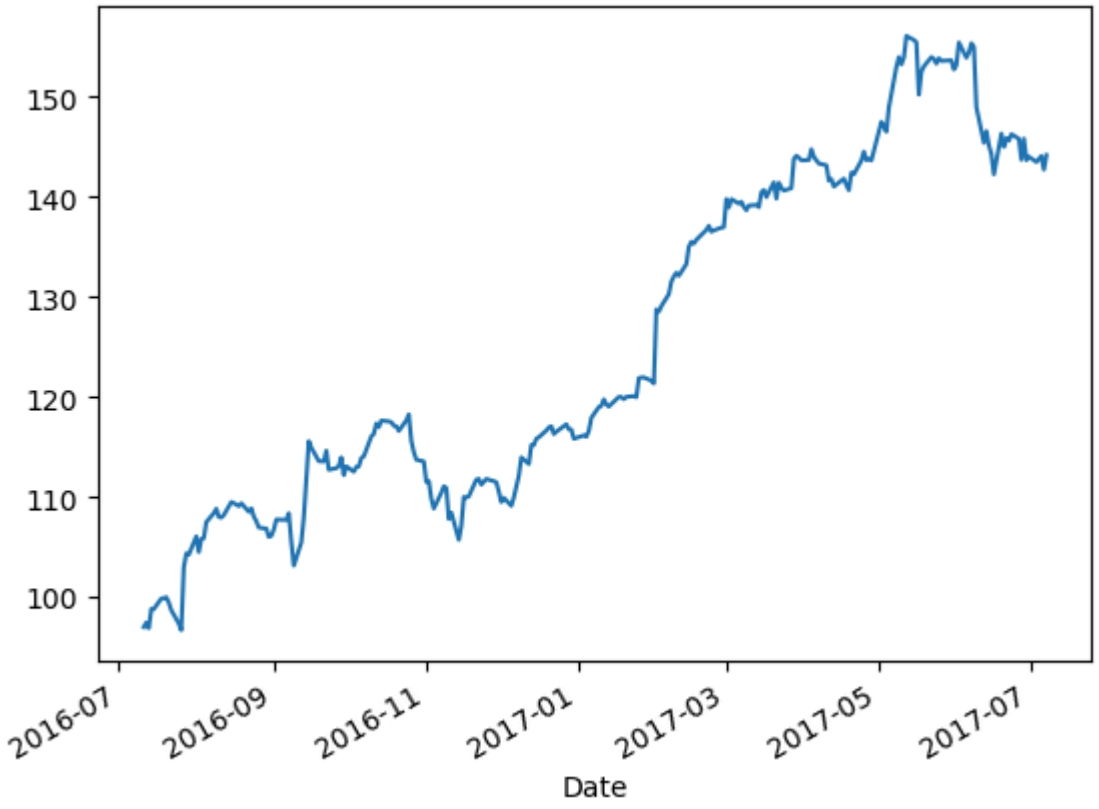
Out[12]:
```
Date
2017-07-07    144.18
2017-07-06    142.73
2017-07-05    144.09
2017-07-03    143.50
2017-06-30    144.02
               ...
2016-07-15     98.78
2016-07-14     98.79
2016-07-13     96.87
2016-07-12     97.42
2016-07-11     96.98
Name: Close, Length: 251, dtype: float64
```

In [14]:
```
dframe.Close.resample('M').mean()
```

Out[14]:
```
Date
2016-07-31     99.473333
2016-08-31    107.665217
2016-09-30    110.857143
2016-10-31    115.707143
2016-11-30    110.154286
2016-12-31    114.335714
2017-01-31    119.570000
2017-02-28    133.713684
2017-03-31    140.617826
2017-04-30    142.886842
2017-05-31    152.227727
2017-06-30    147.831364
2017-07-31    143.625000
Freq: M, Name: Close, dtype: float64
```

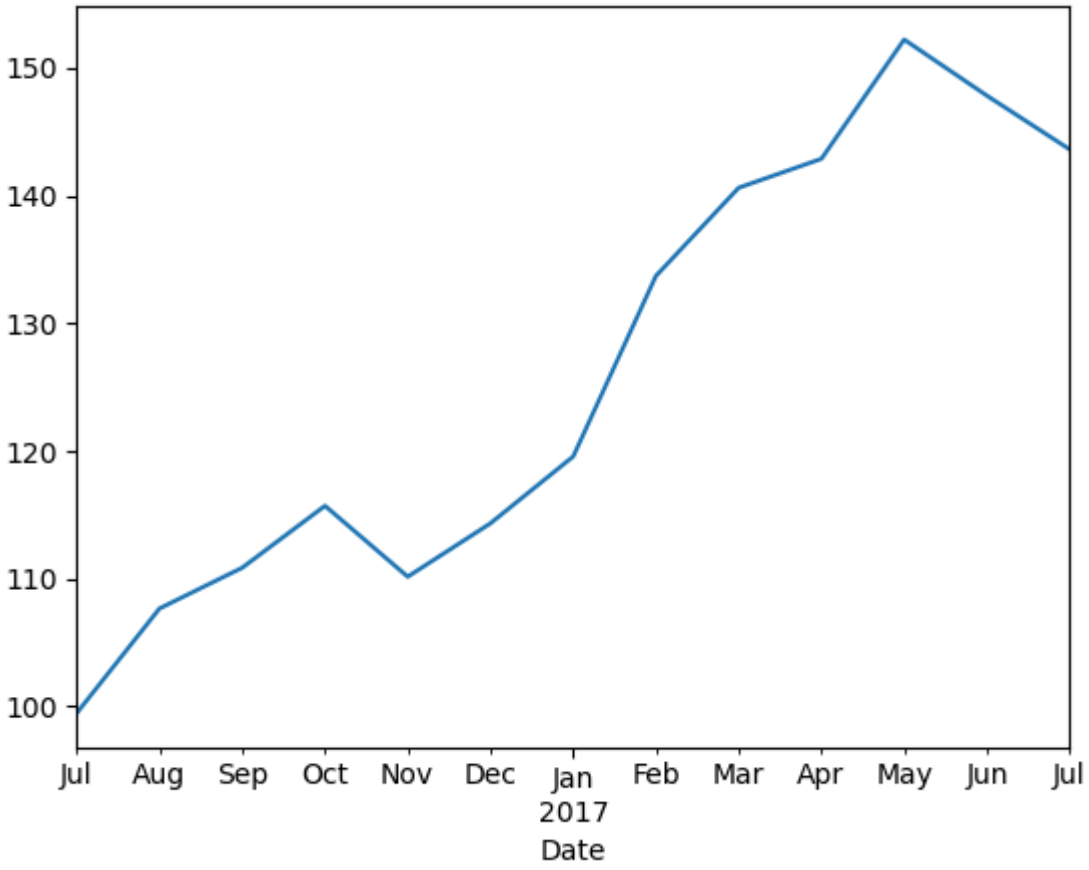
In [15]:
```
dframe.Close.plot()
```

Out[15]:
```
<Axes: xlabel='Date'>
```



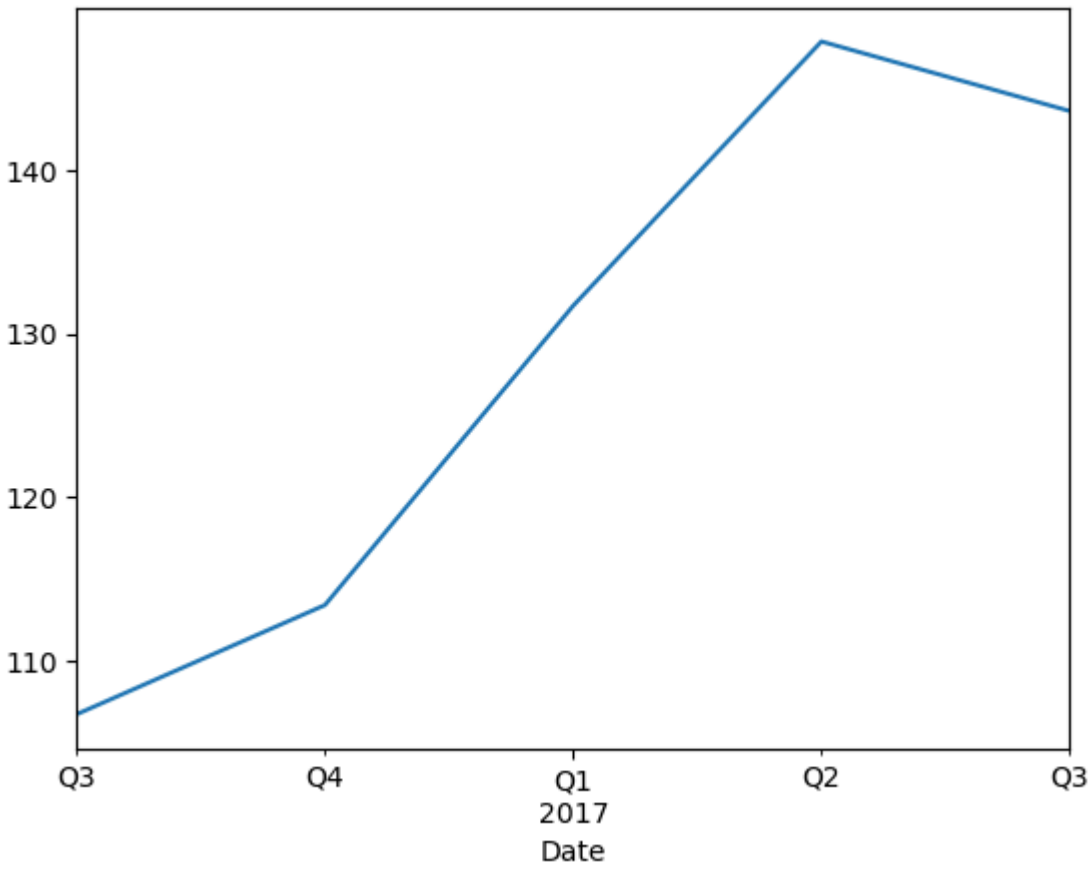
In [16]:
```
dframe.Close.resample('M').mean().plot()
```

Out[16]:
```
<Axes: xlabel='Date'>
```



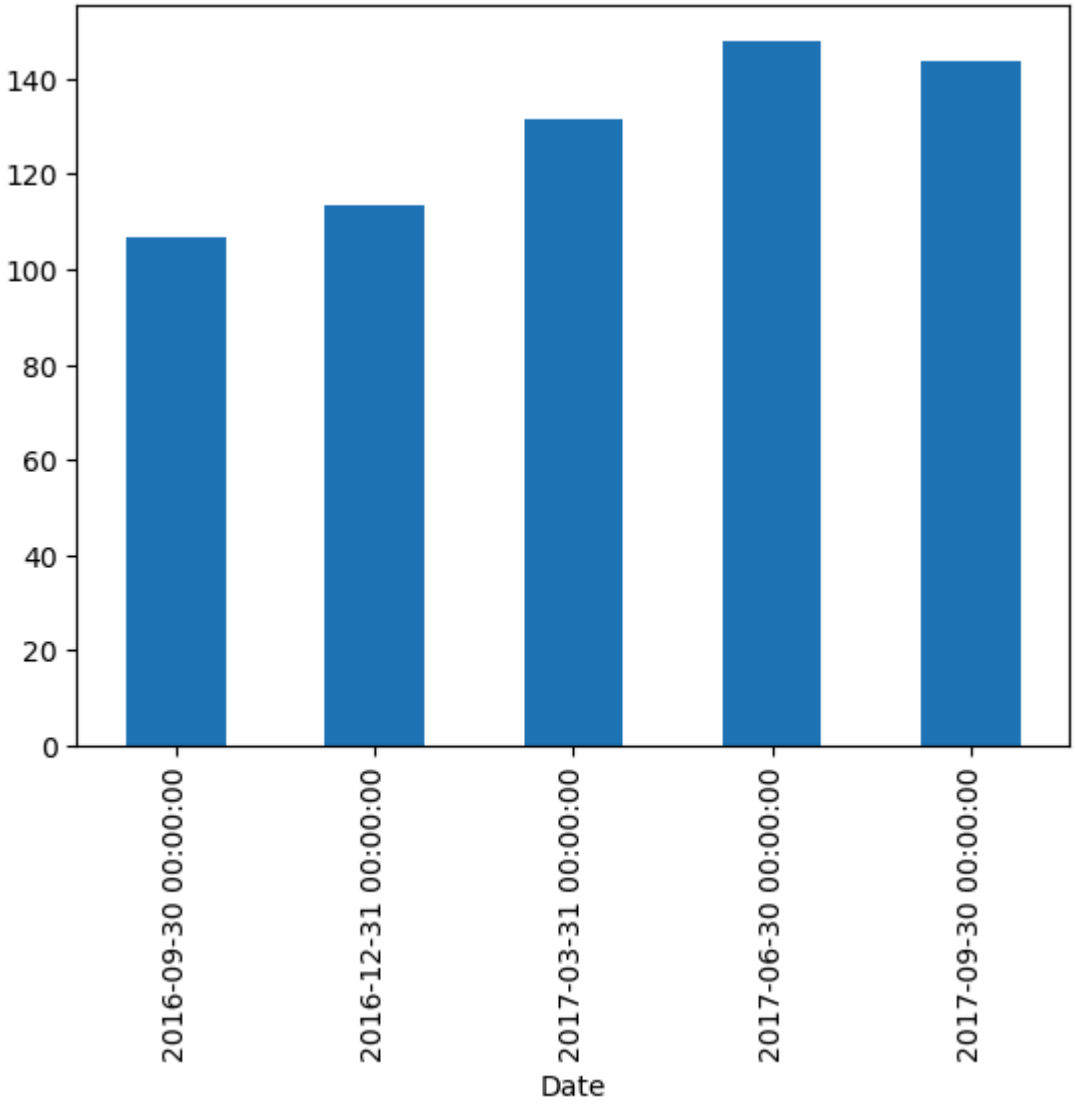
In [17]:
```
dframe.Close.resample('Q').mean().plot()
```

Out[17]: <Axes: xlabel='Date'>



In [18]:
```
dframe.Close.resample('Q').mean().plot(kind="bar")
```

Out[18]: <Axes: xlabel='Date'>



In [26]:
```
tdata = pd.read_csv("train.csv")
tdata
```

Out[26]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN
...	...	...	...	...	...	...	...	...	...	...	...
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.0000	NaN
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.0000	B42
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.4500	NaN
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.0000	C148
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.7500	NaN

891 rows × 12 columns

In [20]:
```
tdata.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   PassengerId  891 non-null    int64  
 1   Survived     891 non-null    int64  
 2   Pclass       891 non-null    int64  
 3   Name         891 non-null    object 
 4   Sex          891 non-null    object 
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64  
 7   Parch        891 non-null    int64  
 8   Ticket       891 non-null    object 
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object 
 11  Embarked     889 non-null    object 
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

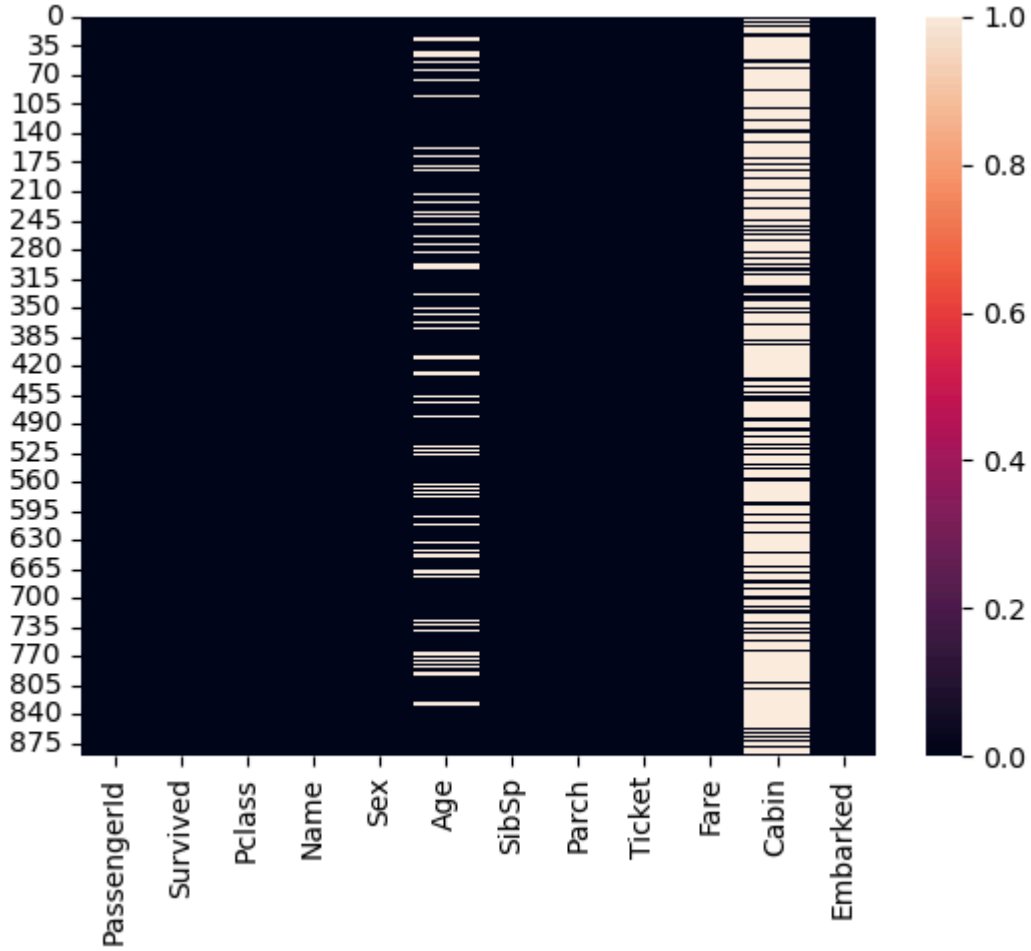
In [21]:
```
tdata.describe()
```

Out[21]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

In [22]:
```
import seaborn as sns
sns.heatmap(tdata.isnull())
```

Out[22]: <Axes: >

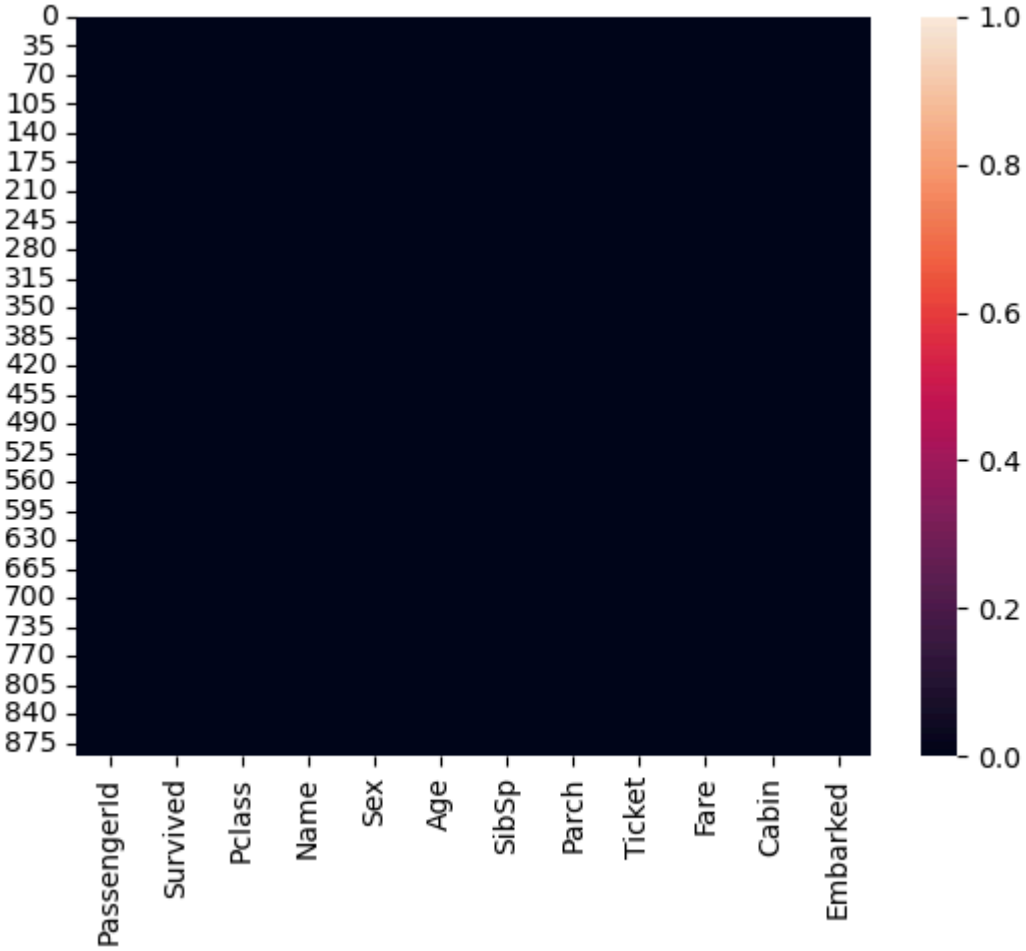


In [23]:
```
ffilldata = tdata.fillna(method="ffill")
```

```
C:\Users\bhanu\AppData\Local\Temp\ipykernel_33916\1215858901.py:1: FutureWarning: DataFrame.fill
na with 'method' is deprecated and will raise in a future version. Use obj.ffill() or obj.bfill
() instead.
  ffilldata = tdata.fillna(method="ffill")
```

In [24]:
```
sns.heatmap(ffilldata.isnull())
```

Out[24]:
```
<Axes: >
```



In [25]:
```
import matplotlib.pyplot as plt
```

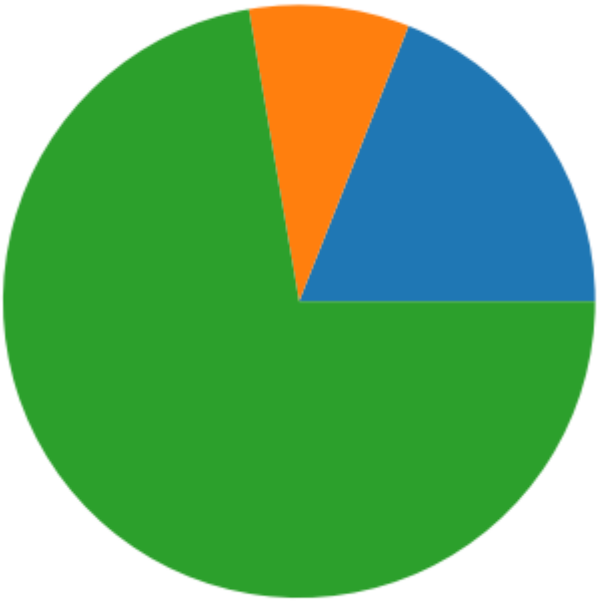
In [29]:
```
bc = ffilldata.groupby('Embarked')['Embarked'].count()
```

In [30]:
```
bc
```

Out[30]:
```
Embarked
C    169
Q     78
S    644
Name: Embarked, dtype: int64
```

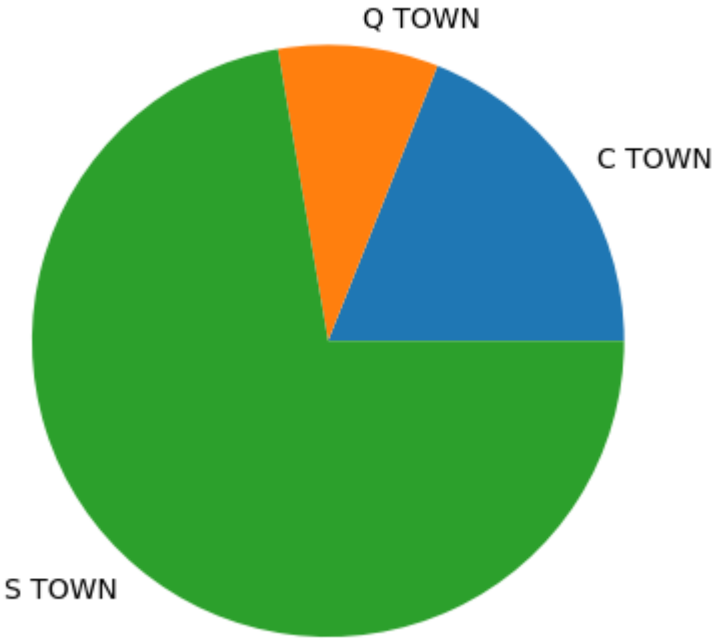
In [31]:
```
plt.figure()
plt.pie(bc.values)
```

Out[31]:
```
([<matplotlib.patches.Wedge at 0x242e28a7d90>,
  <matplotlib.patches.Wedge at 0x242e16d4b50>,
  <matplotlib.patches.Wedge at 0x242e16cad50>],
 [Text(0.9104203625508537, 0.6173611288806352, ''),
  Text(0.11420980728720928, 1.0940548980373055, ''),
  Text(-0.708550819151803, -0.8414010557868995, '')])
```



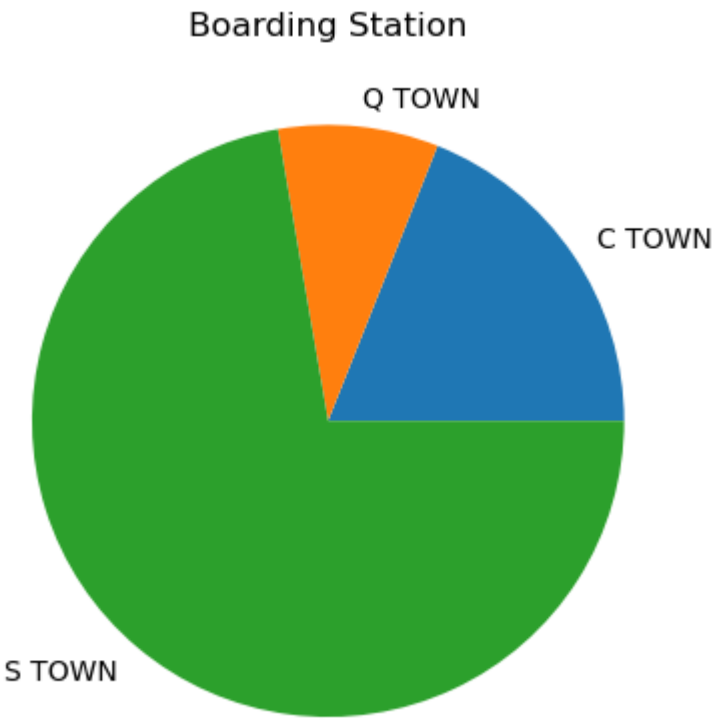
In [33]:
```
plt.pie(bc.values, labels=['C TOWN','Q TOWN','S TOWN'])
```

Out[33]:
```
([<matplotlib.patches.Wedge at 0x242e2d5a890>,
  <matplotlib.patches.Wedge at 0x242e156ded0>,
  <matplotlib.patches.Wedge at 0x242e34cf350>],
 [Text(0.9104203625508537, 0.6173611288806352, 'C TOWN'),
  Text(0.11420980728720928, 1.0940548980373055, 'Q TOWN'),
  Text(-0.708550819151803, -0.8414010557868995, 'S TOWN')])
```



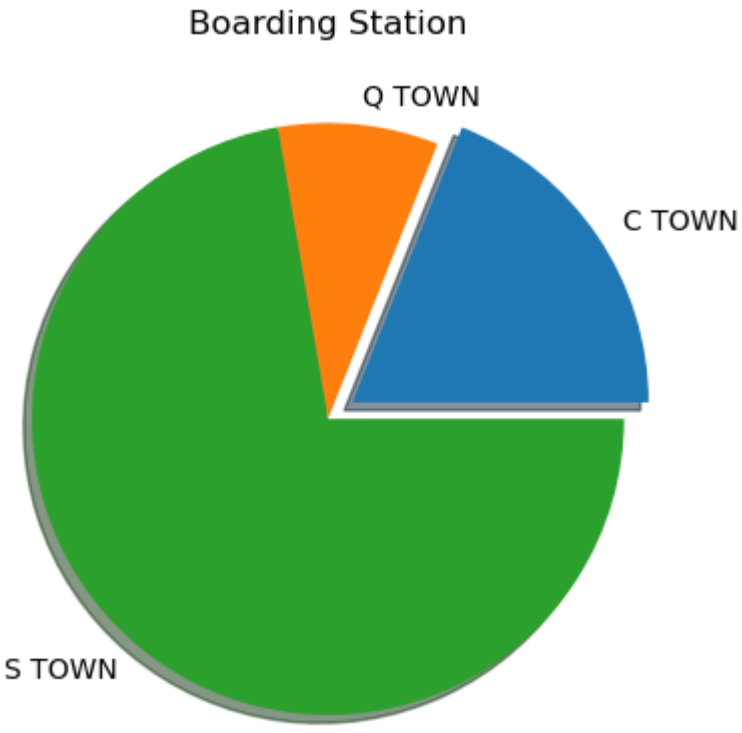
In [34]:
```
plt.title("Boarding Station")
plt.pie(bc.values, labels=['C TOWN','Q TOWN','S TOWN'])
```

Out[34]:
```
([<matplotlib.patches.Wedge at 0x242e355a9d0>,
  <matplotlib.patches.Wedge at 0x242e3562b50>,
  <matplotlib.patches.Wedge at 0x242e35732d0>],
 [Text(0.9104203625508537, 0.6173611288806352, 'C TOWN'),
  Text(0.11420980728720928, 1.0940548980373055, 'Q TOWN'),
  Text(-0.708550819151803, -0.8414010557868995, 'S TOWN')])
```



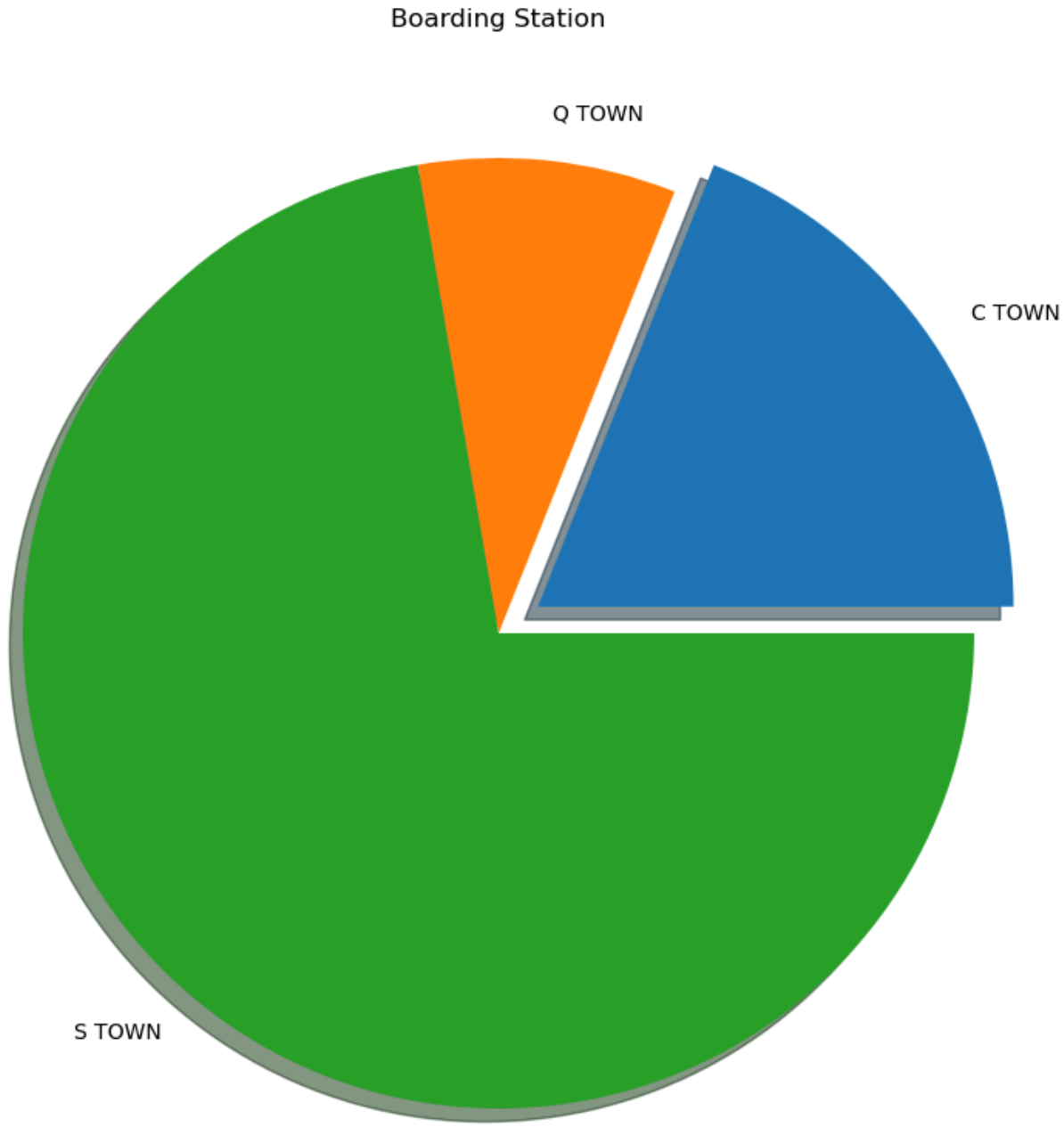
In [37]:
```
plt.title("Boarding Station")
exp = [0.1,0,0]
plt.pie(bc.values, labels=['C TOWN','Q TOWN','S TOWN'], explode=exp, shadow=True)
```

Out[37]:
```
([<matplotlib.patches.Wedge at 0x242e357ec90>,
  <matplotlib.patches.Wedge at 0x242e3642790>,
  <matplotlib.patches.Wedge at 0x242e3643f90>],
 [Text(0.9931858500554768, 0.6734848678697839, 'C TOWN'),
  Text(0.11420980728720928, 1.0940548980373055, 'Q TOWN'),
  Text(-0.708550819151803, -0.8414010557868995, 'S TOWN')])
```



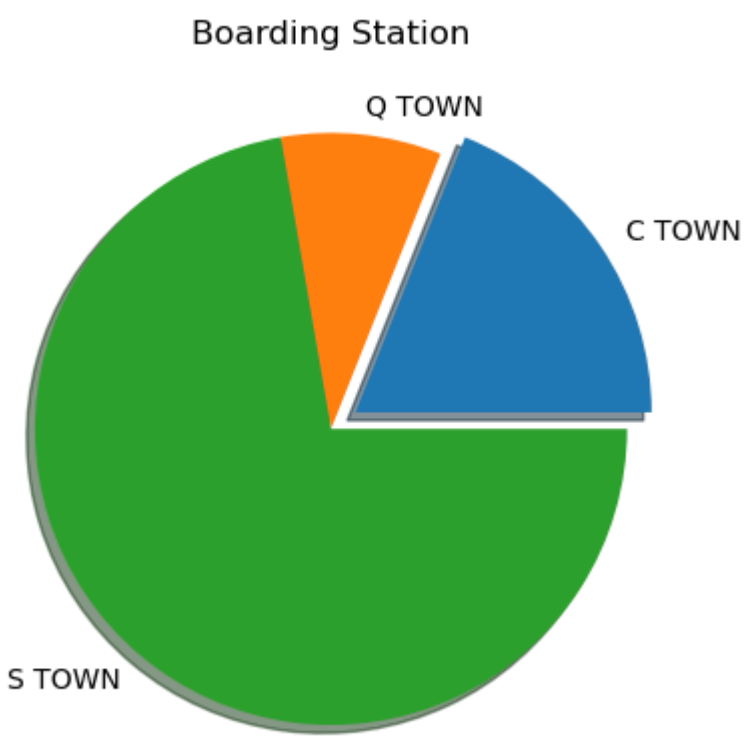
In [38]:
```
plt.figure(figsize=(10,10))
plt.title("Boarding Station")
exp = [0.1,0,0]
plt.pie(bc.values, labels=['C TOWN','Q TOWN','S TOWN'], explode=exp, shadow=True)
```

Out[38]:
```
([<matplotlib.patches.Wedge at 0x242e2d65f10>,
  <matplotlib.patches.Wedge at 0x242e3657790>,
  <matplotlib.patches.Wedge at 0x242e3698f10>],
 [Text(0.9931858500554768, 0.6734848678697839, 'C TOWN'),
  Text(0.11420980728720928, 1.0940548980373055, 'Q TOWN'),
  Text(-0.708550819151803, -0.8414010557868995, 'S TOWN')])
```



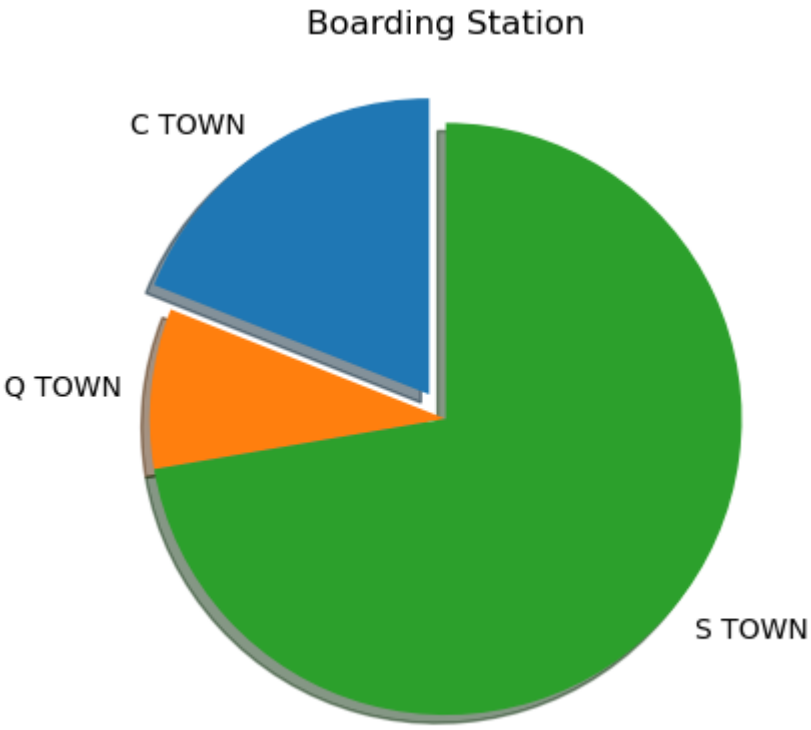
In [39]:
```
plt.title("Boarding Station")
exp = [0.1,0,0]
plt.pie(bc.values, labels=['C TOWN','Q TOWN','S TOWN'], explode=exp, shadow=True)
```

Out[39]:
```
([<matplotlib.patches.Wedge at 0x242e369a9d0>,
  <matplotlib.patches.Wedge at 0x242e36d7b90>,
  <matplotlib.patches.Wedge at 0x242e36e4750>],
 [Text(0.9931858500554768, 0.6734848678697839, 'C TOWN'),
  Text(0.11420980728720928, 1.0940548980373055, 'Q TOWN'),
  Text(-0.708550819151803, -0.8414010557868995, 'S TOWN')])
```



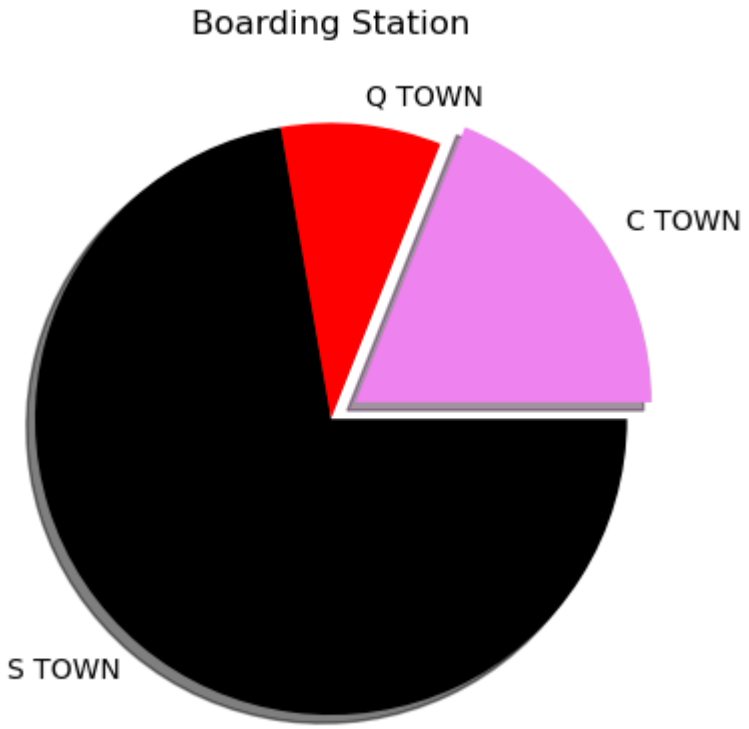
In [40]:
```
plt.title("Boarding Station")
exp = [0.1,0,0]
plt.pie(bc.values, labels=['C TOWN','Q TOWN','S TOWN'], explode=exp, shadow=True, startangle=90
```

Out[40]:
```
([<matplotlib.patches.Wedge at 0x242e3703c10>,
  <matplotlib.patches.Wedge at 0x242e36d65d0>,
  <matplotlib.patches.Wedge at 0x242e36ec750>],
 [Text(-0.6734848678697838, 0.9931858500554768, 'C TOWN'),
  Text(-1.0940548980373055, 0.11420980728720935, 'Q TOWN'),
  Text(0.8414010557868994, -0.7085508191518031, 'S TOWN')])
```



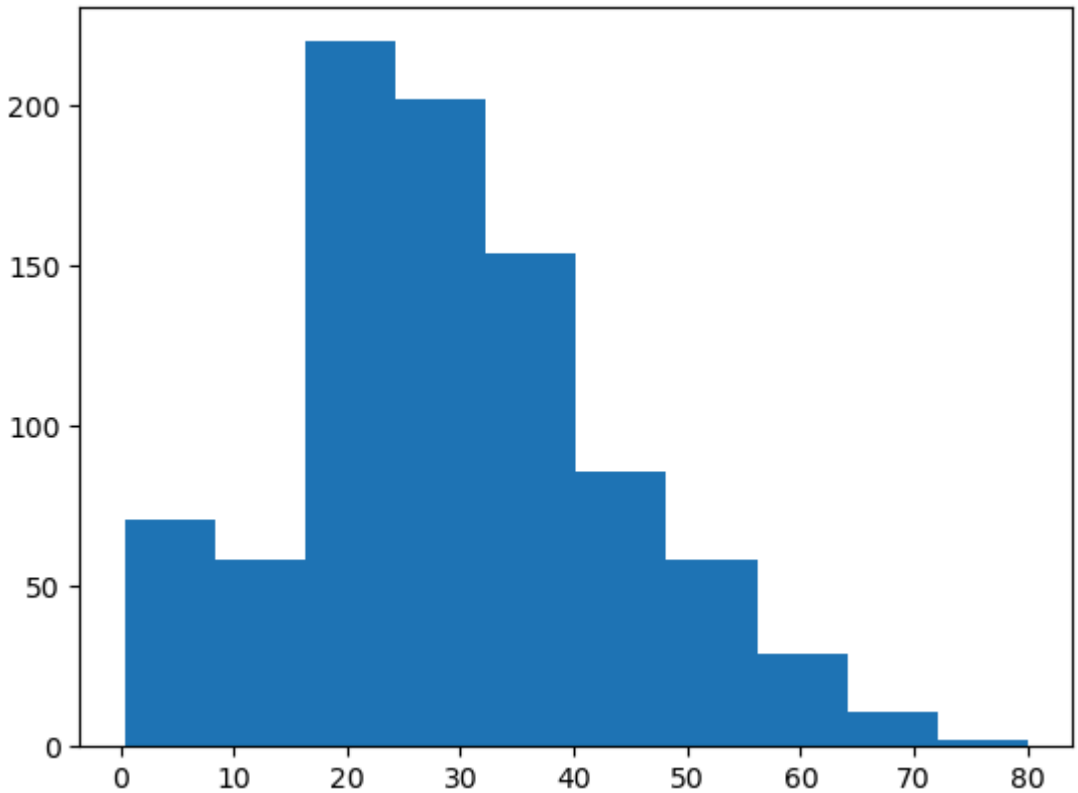
In [43]:
```
plt.title("Boarding Station")
exp = [0.1,0,0]
plt.pie(bc.values, labels=['C TOWN','Q TOWN','S TOWN'], explode=exp, shadow=True, colors=["Viol
```

Out[43]:
```
([<matplotlib.patches.Wedge at 0x242e8caa9d0>,
  <matplotlib.patches.Wedge at 0x242e8cb5a10>,
  <matplotlib.patches.Wedge at 0x242e8cb6e90>],
 [Text(0.9931858500554768, 0.6734848678697839, 'C TOWN'),
  Text(0.11420980728720928, 1.0940548980373055, 'Q TOWN'),
  Text(-0.708550819151803, -0.8414010557868995, 'S TOWN')])
```



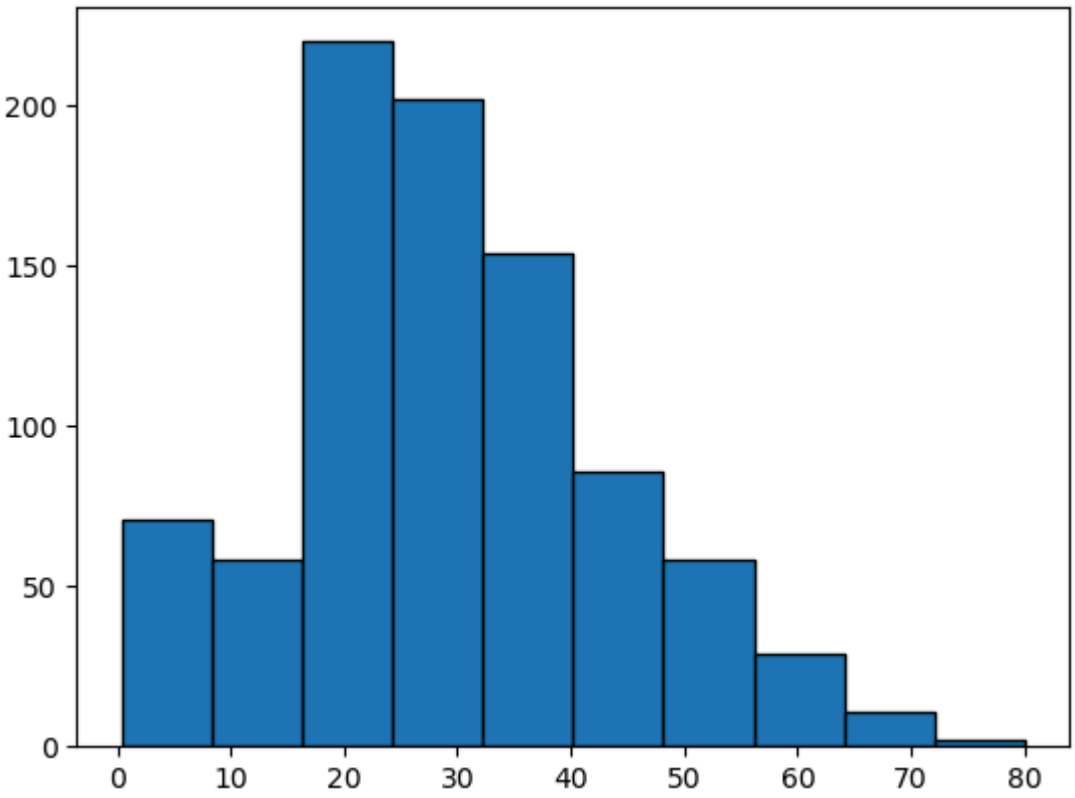
In [44]:
```
#Histogram - Range of values
plt.hist(ffilldata['Age'])
```

Out[44]:
```
(array([ 71.,  58., 220., 202., 154.,  86.,  58.,  29.,  11.,   2.]),
 array([ 0.42 ,  8.378, 16.336, 24.294, 32.252, 40.21 , 48.168, 56.126,
        64.084, 72.042, 80.   ]),
 <BarContainer object of 10 artists>)
```



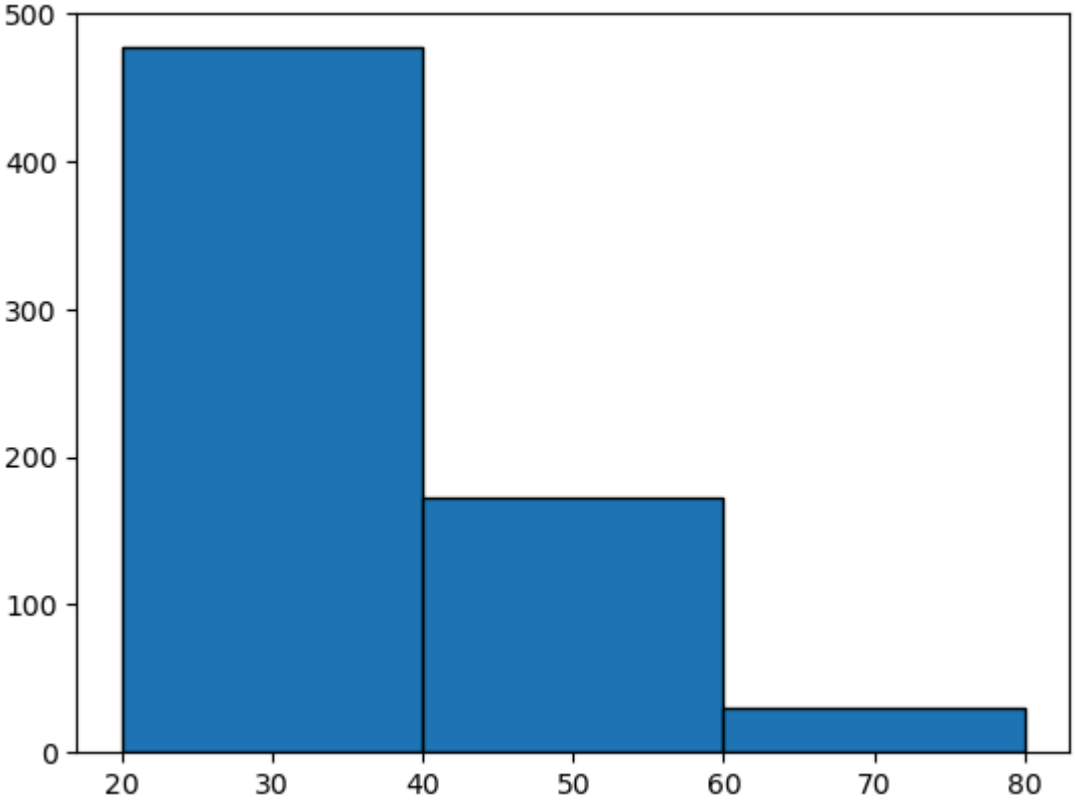
In [45]:
```
plt.hist(ffilldata['Age'], edgecolor='black')
```

Out[45]:
```
(array([ 71.,  58., 220., 202., 154.,  86.,  58.,  29.,  11.,   2.]),
 array([ 0.42 ,  8.378, 16.336, 24.294, 32.252, 40.21 , 48.168, 56.126,
        64.084, 72.042, 80.   ]),
 <BarContainer object of 10 artists>)
```



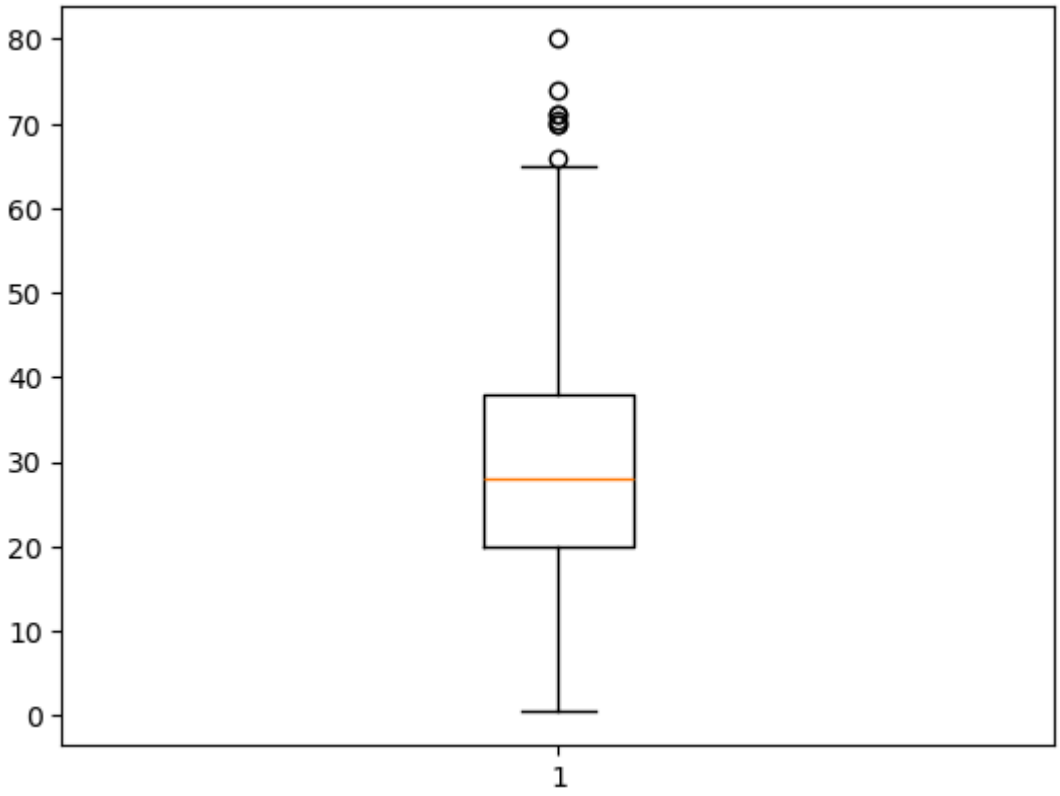
In [46]:
```
plt.hist(ffilldata['Age'], edgecolor='black', bins=[20,40,60,80])
```

Out[46]:
```
(array([477., 173.,  30.]),
 array([20., 40., 60., 80.]),
 <BarContainer object of 3 artists>)
```



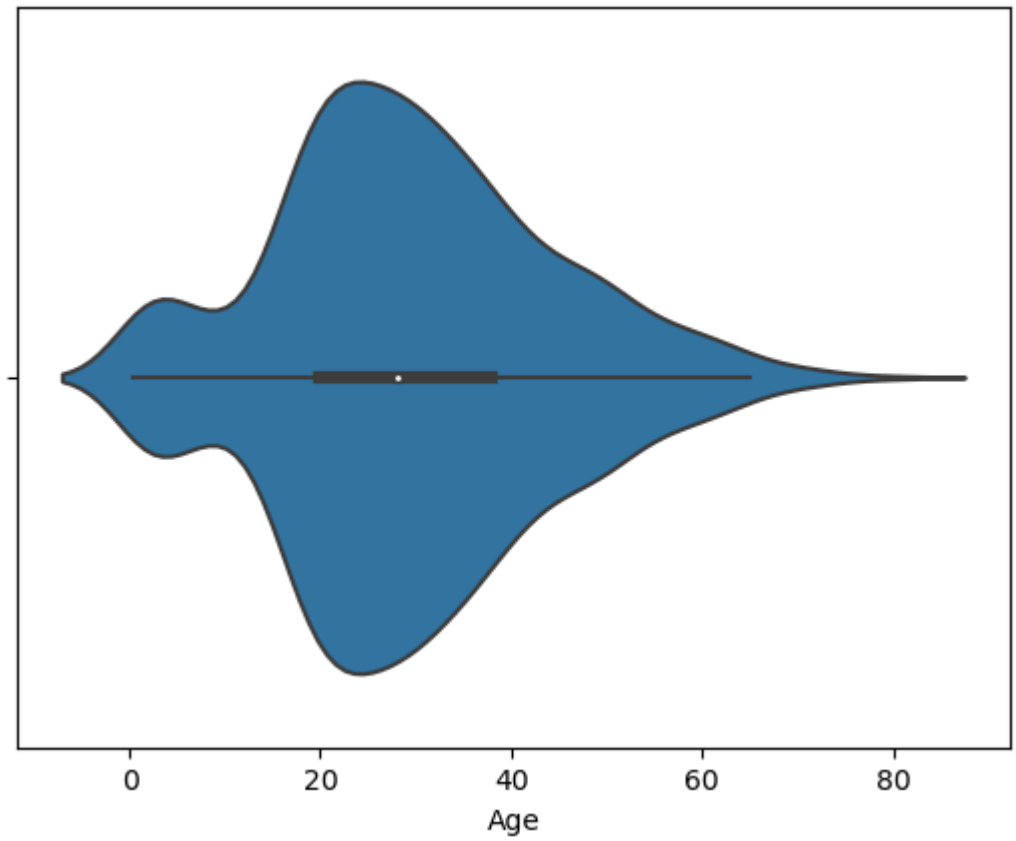
In [47]:
```
plt.boxplot(ffilldata['Age'])
```

Out[47]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x242e8e69c10>,
  <matplotlib.lines.Line2D at 0x242e8e6a650>],
 'caps': [<matplotlib.lines.Line2D at 0x242e8e6b010>,
  <matplotlib.lines.Line2D at 0x242e8e5de90>],
 'boxes': [<matplotlib.lines.Line2D at 0x242e8dae3d0>],
 'medians': [<matplotlib.lines.Line2D at 0x242e8e6be50>],
 'fliers': [<matplotlib.lines.Line2D at 0x242e8e7c650>],
 'means': []}
```



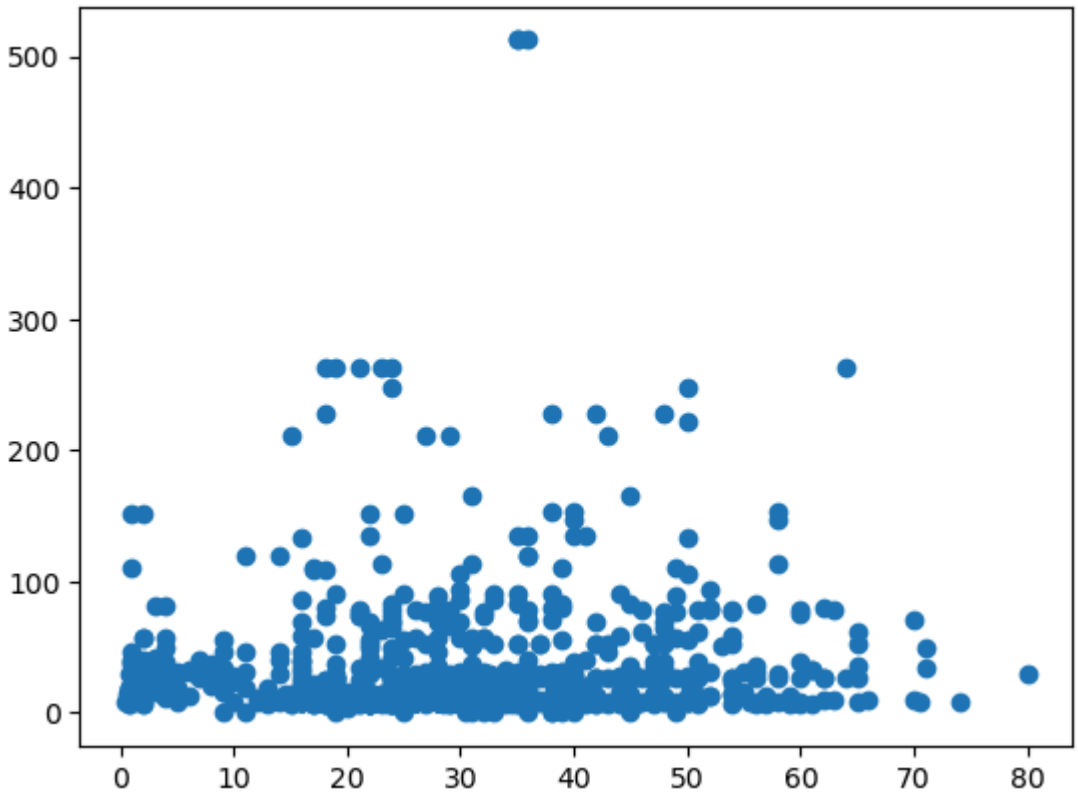
In [49]:
```
sns.violinplot(x=ffilldata['Age'])
```

Out[49]:
```
<Axes: xlabel='Age'>
```



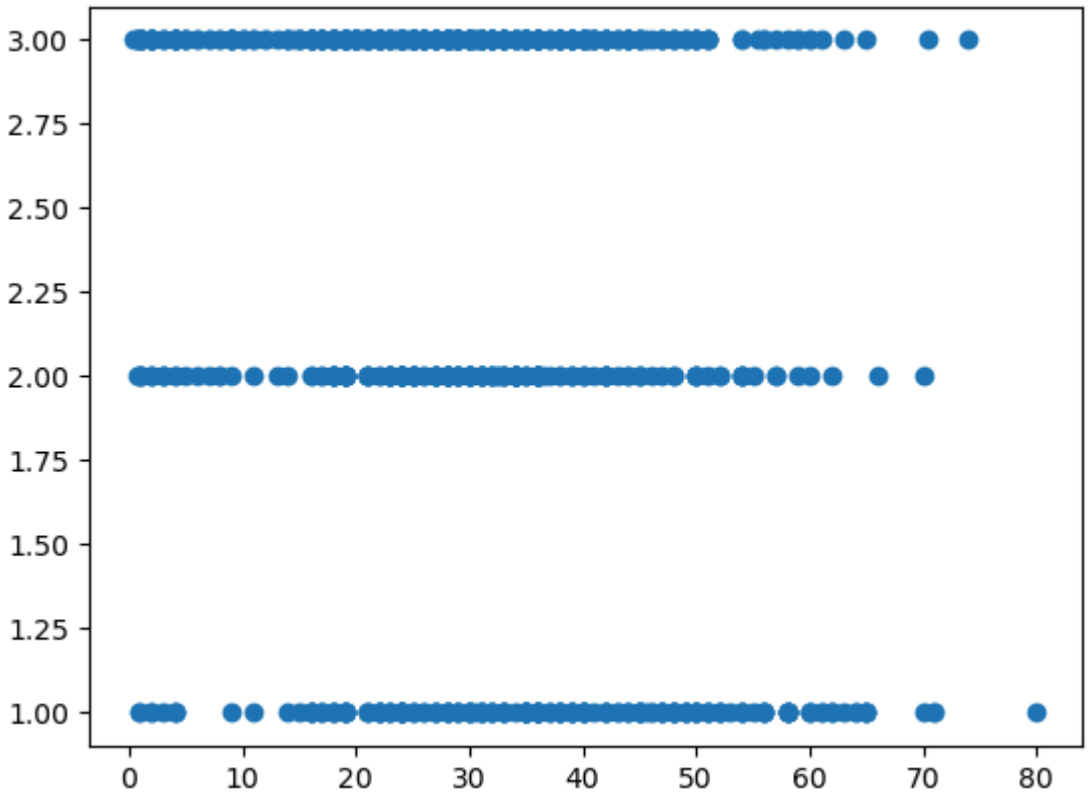
In [50]:
```
plt.scatter(ffilldata['Age'],ffilldata['Fare'])
```

Out[50]: <matplotlib.collections.PathCollection at 0x242e8f35e90>



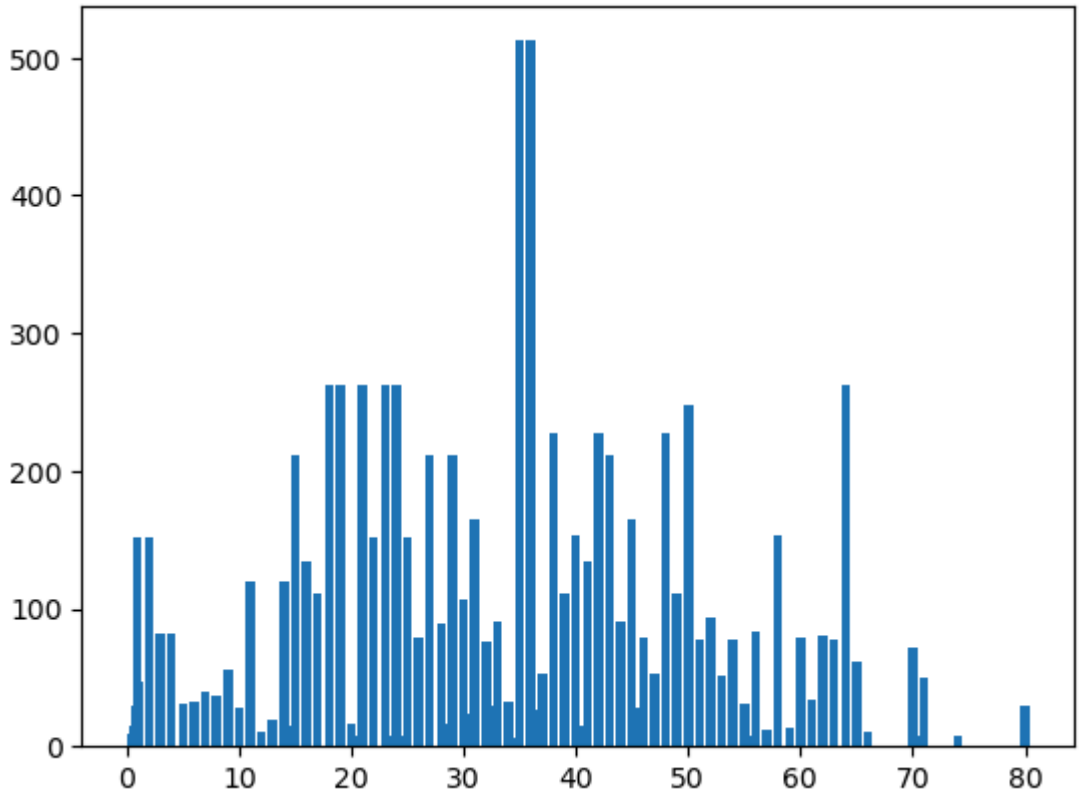
In [51]:
```
plt.scatter(ffilldata['Age'],ffilldata['Pclass'])
```

Out[51]: <matplotlib.collections.PathCollection at 0x242e97e83d0>



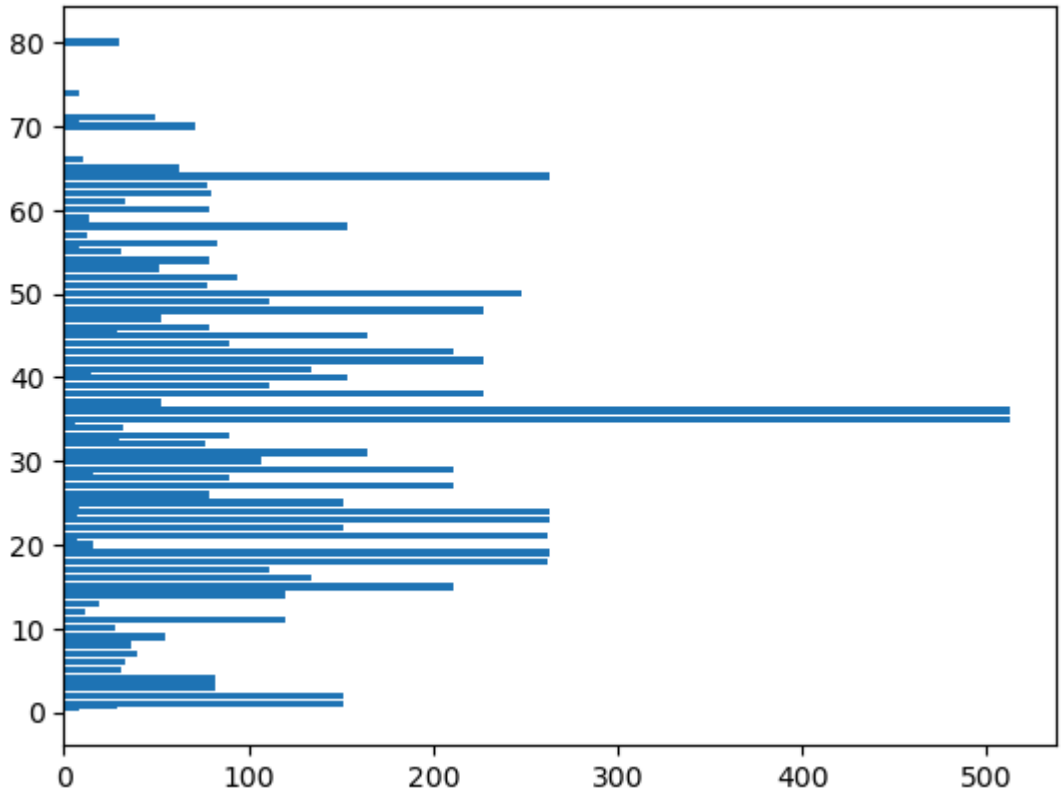
In [52]:
```
plt.bar(ffilldata['Age'],ffilldata['Fare'])
```

Out[52]: <BarContainer object of 891 artists>



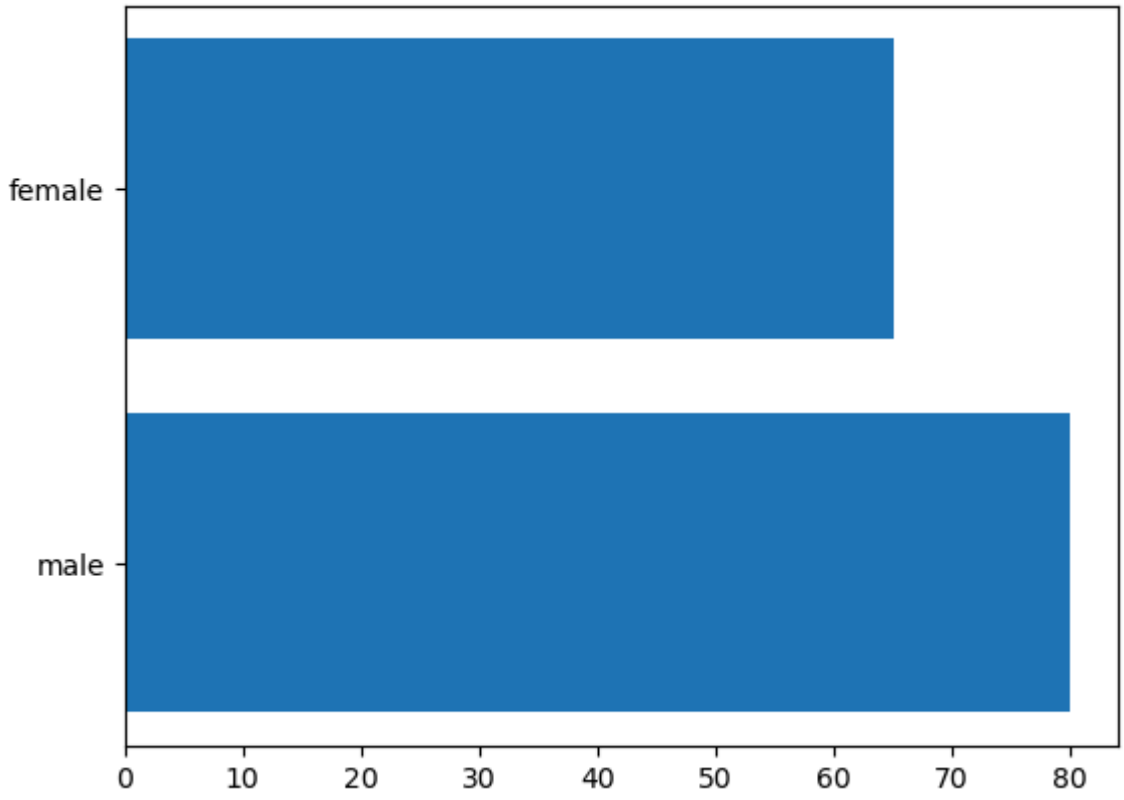
In [53]:
```
plt.barh(ffilldata['Age'],ffilldata['Fare'])
```

Out[53]: <BarContainer object of 891 artists>



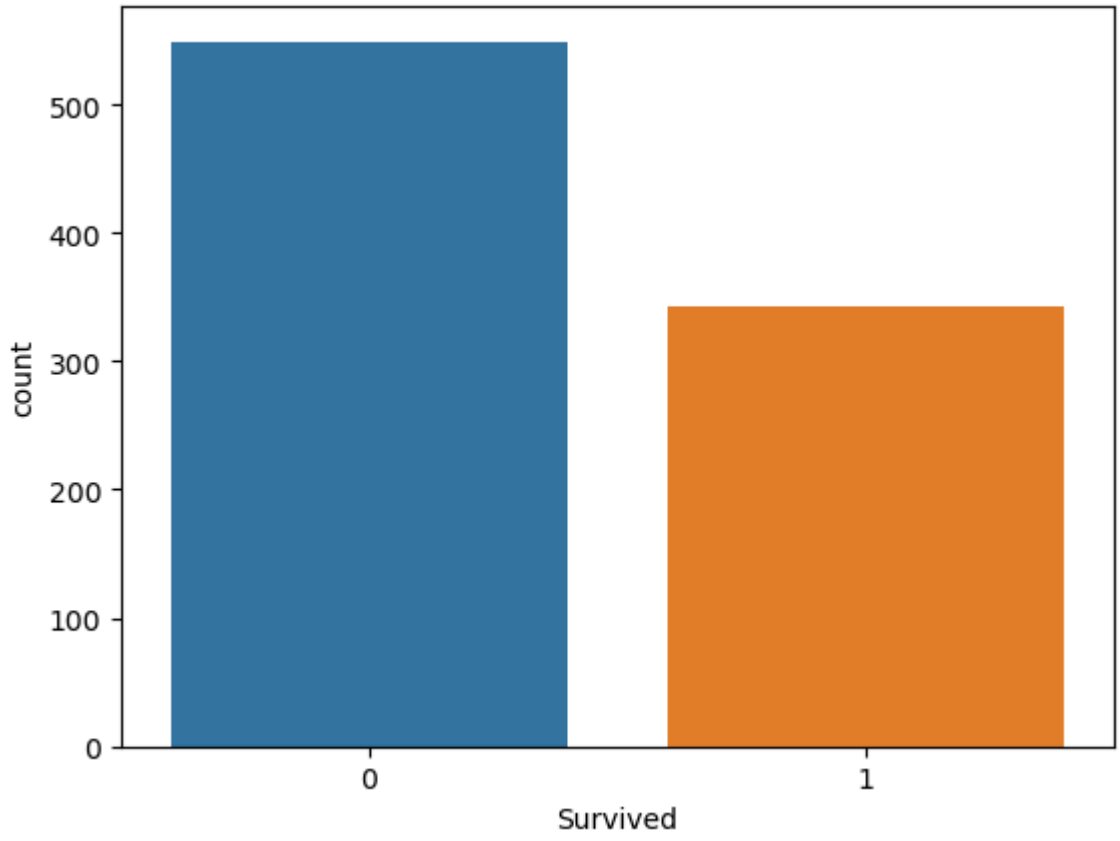
In [54]:
```
plt.barh(ffilldata['Sex'],ffilldata['Age'])
```

Out[54]: <BarContainer object of 891 artists>



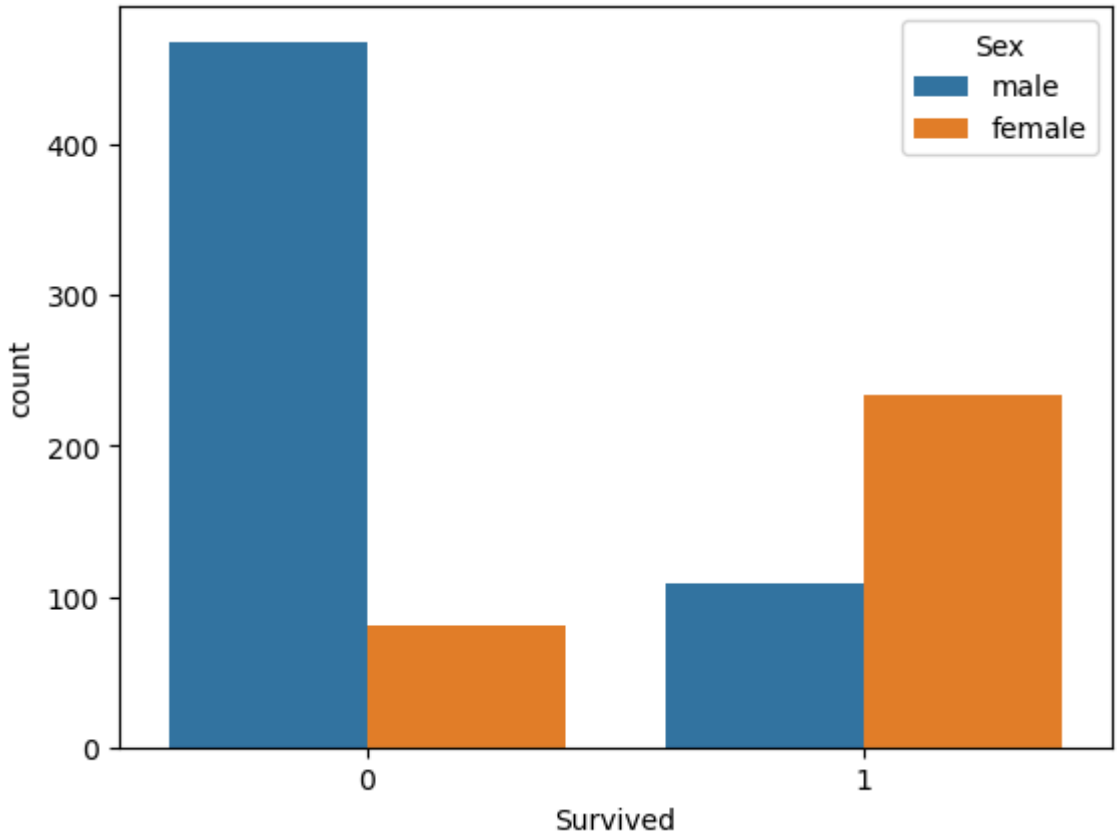
In [56]:
```
sns.countplot(x=ffilldata['Survived'],data=ffilldata)
```

Out[56]: <Axes: xlabel='Survived', ylabel='count'>



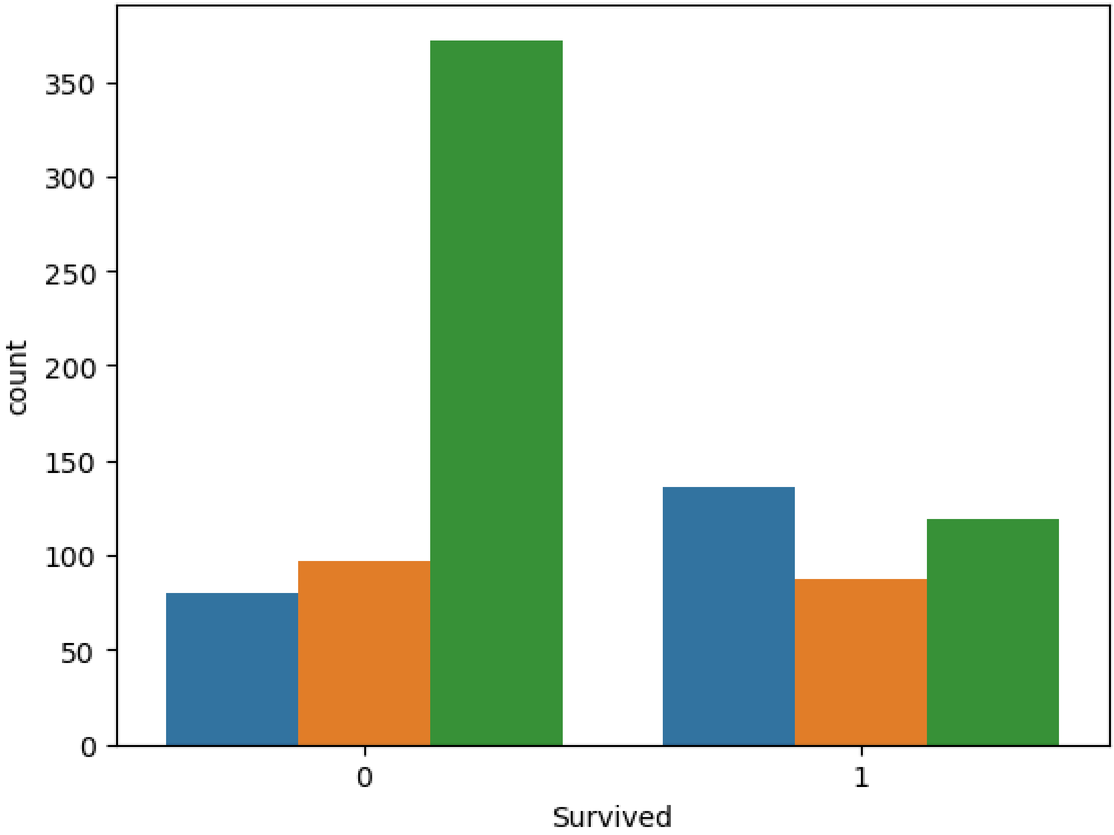
In [57]:
```
sns.countplot(x=ffilldata['Survived'],data=ffilldata, hue='Sex')
```

Out[57]: <Axes: xlabel='Survived', ylabel='count'>



In [58]:
```
sns.countplot(x=ffilldata['Survived'],data=ffilldata, hue='Pclass')
```

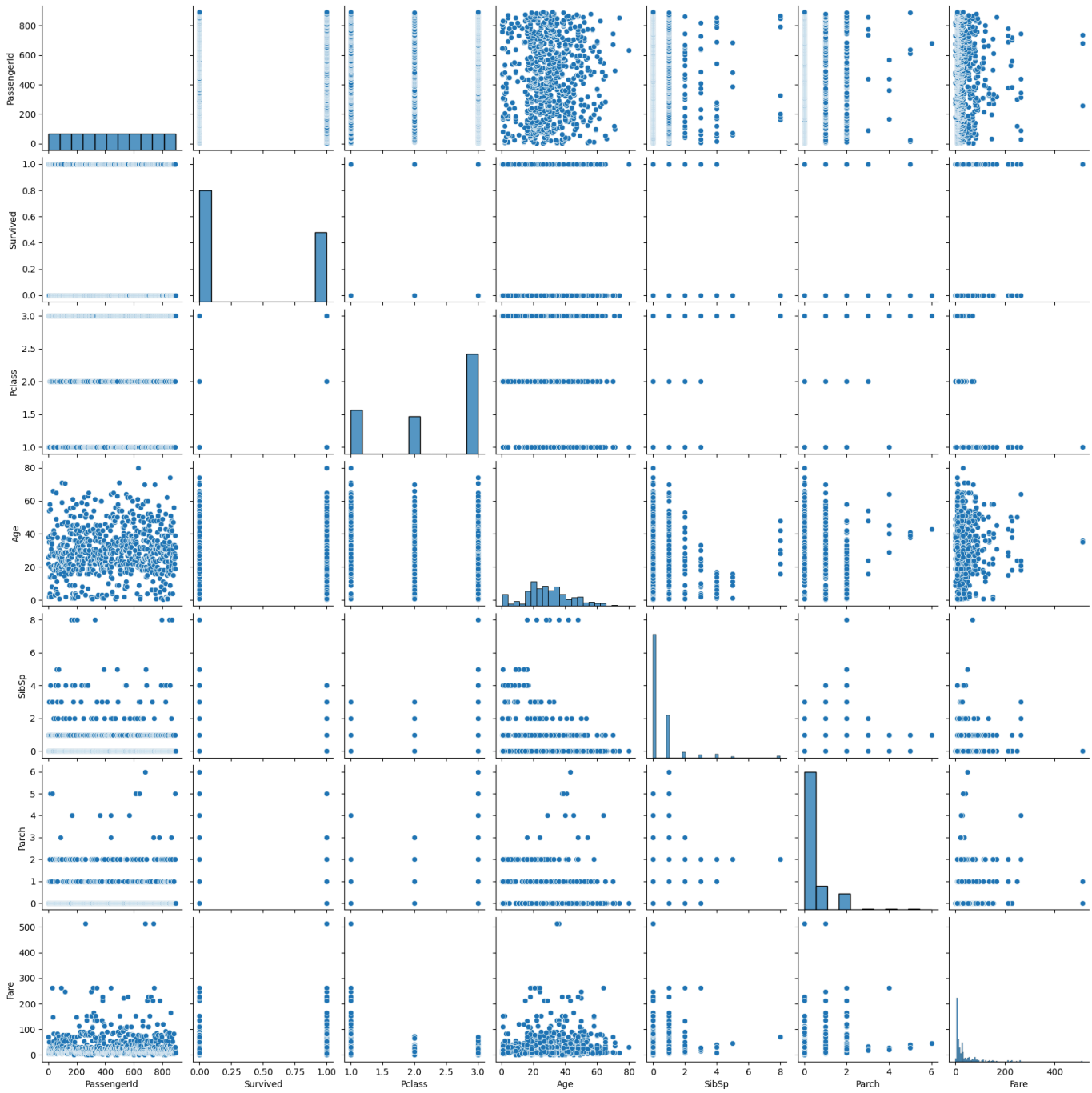
```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[58], line 1
----> 1 sns.countplot(x=ffilldata['Survived'],data=ffilldata, hue='Pclass')

File ~\anaconda3\Lib\site-packages\seaborn\categorical.py:2955, in countplot(data, x, y, hue, order, hue_order, orient, color, palette, saturation, width, dodge, ax, **kwargs)
   2952 if ax is None:
   2953     ax = plt.gca()
-> 2955 plotter.plot(ax, kwargs)
   2956 return ax

File ~\anaconda3\Lib\site-packages\seaborn\categorical.py:1587, in _BarPlotter.plot(self, ax, bar_kws)
   1585 """Make the plot."""
   1586 self.draw_bars(ax, bar_kws)
-> 1587 self.annotate_axes(ax)
   1588 if self.orient == "h":
   1589     ax.invert_yaxis()

File ~\anaconda3\Lib\site-packages\seaborn\categorical.py:767, in _CategoricalPlotter.annotate_axes(self, ax)
    764     ax.set_ylim(-.5, len(self.plot_data) - .5, auto=None)
    766 if self.hue_names is not None:
--> 767     ax.legend(loc="best", title=self.hue_title)

File ~\anaconda3\Lib\site-packages\matplotlib\axes\_axes.py:322, in Axes.legend(self, *args, **kwargs)
    204 @_docstring.dedent_interpd
    205 def legend(self, *args, **kwargs):
    206     """
    207     Place a legend on the Axes.
    208
   (...)
    320     .. plot:: gallery/text_labels_and_annotations/legend.py
    321     """
--> 322     handles, labels, kwargs = mlegend._parse_legend_args([self], *args, **kwargs)
    323     self.legend_ = mlegend.Legend(self, handles, labels, **kwargs)
    324     self.legend_._remove_method = self._remove_legend

File ~\anaconda3\Lib\site-packages\matplotlib\legend.py:1361, in _parse_legend_args(axs, handles, labels, *args, **kwargs)
   1357     handles = [handle for handle, label
   1358                in zip(_get_legend_handles(axs, handlers), labels)]
   1360 elif len(args) == 0:  # 0 args: automatically detect labels and handles.
-> 1361     handles, labels = _get_legend_handles_labels(axs, handlers)
   1362     if not handles:
   1363         log.warning(
   1364             "No artists with labels found to put in legend.  Note that "
   1365             "artists whose label start with an underscore are ignored "
   1366             "when legend() is called with no argument.")

File ~\anaconda3\Lib\site-packages\matplotlib\legend.py:1291, in _get_legend_handles_labels(axs, legend_handler_map)
   1289 for handle in _get_legend_handles(axs, legend_handler_map):
   1290     label = handle.get_label()
-> 1291     if label and not label.startswith('_'):
   1292         handles.append(handle)
   1293         labels.append(label)

AttributeError: 'numpy.int64' object has no attribute 'startswith'
```



In [59]:
```
sns.pairplot(ffilldata)
```

```
C:\Users\bhanu\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_n
a option is deprecated and will be removed in a future version. Convert inf values to NaN before
operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\bhanu\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_n
a option is deprecated and will be removed in a future version. Convert inf values to NaN before
operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\bhanu\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_n
a option is deprecated and will be removed in a future version. Convert inf values to NaN before
operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\bhanu\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_n
a option is deprecated and will be removed in a future version. Convert inf values to NaN before
operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\bhanu\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_n
a option is deprecated and will be removed in a future version. Convert inf values to NaN before
operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\bhanu\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_n
a option is deprecated and will be removed in a future version. Convert inf values to NaN before
operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\bhanu\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_n
a option is deprecated and will be removed in a future version. Convert inf values to NaN before
operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

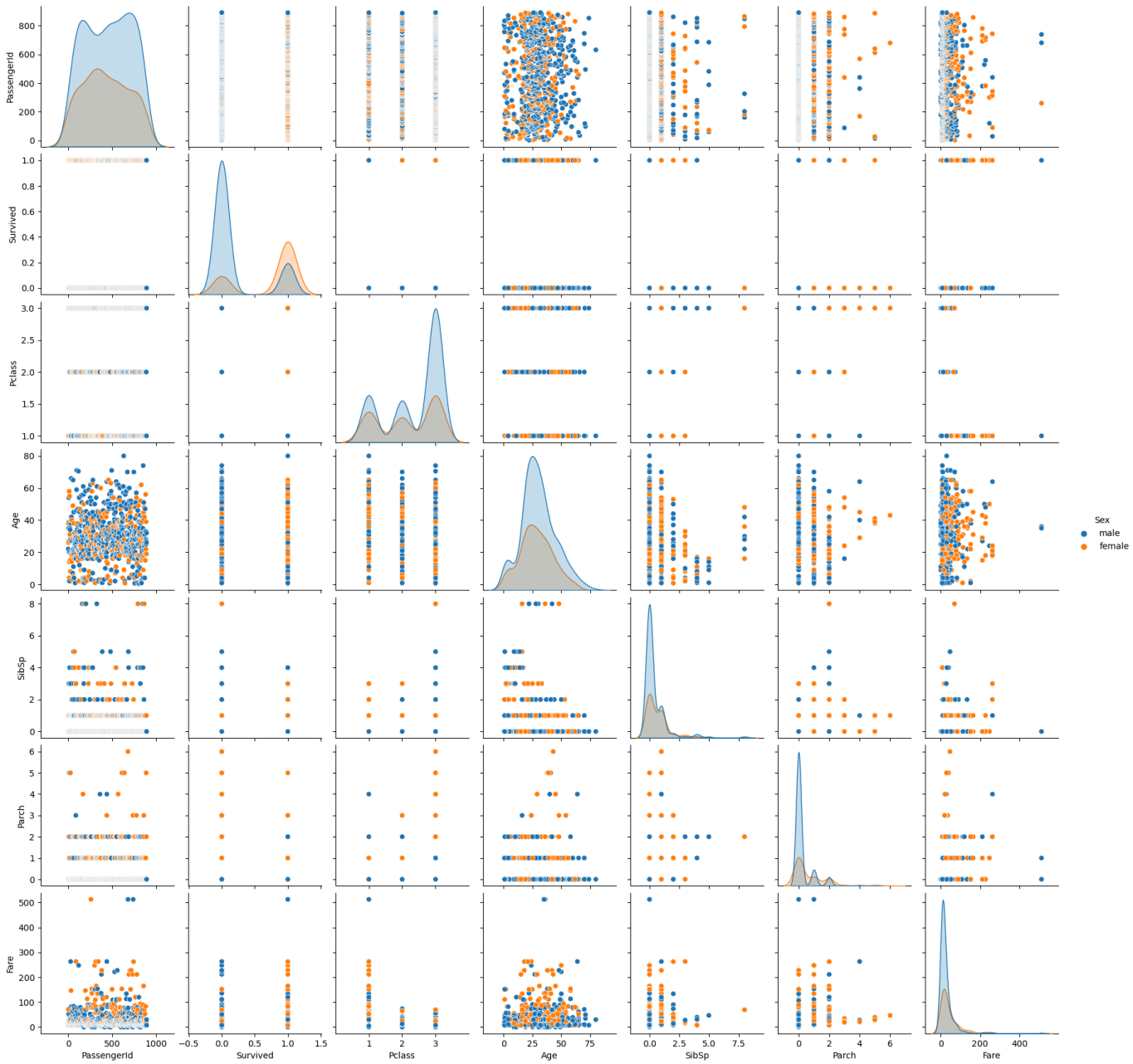
Out[59]:
```
<seaborn.axisgrid.PairGrid at 0x242ea53fe10>
```



In [60]:
```
sns.pairplot(ffilldata, hue='Sex')
```

```
C:\Users\bhanu\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_n
a option is deprecated and will be removed in a future version. Convert inf values to NaN before
operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\bhanu\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_n
a option is deprecated and will be removed in a future version. Convert inf values to NaN before
operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\bhanu\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_n
a option is deprecated and will be removed in a future version. Convert inf values to NaN before
operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\bhanu\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_n
a option is deprecated and will be removed in a future version. Convert inf values to NaN before
operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\bhanu\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_n
a option is deprecated and will be removed in a future version. Convert inf values to NaN before
operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\bhanu\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_n
a option is deprecated and will be removed in a future version. Convert inf values to NaN before
operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\bhanu\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_n
a option is deprecated and will be removed in a future version. Convert inf values to NaN before
operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

Out[60]:
```
<seaborn.axisgrid.PairGrid at 0x242efd86f90>
```


PassengerId
Survived
Pclass
Age
SibSp
Parch
Fare
Sex
male
female

In [ ]: