In [3]:
```
#Problem 1
## Create a pandas Series where the index labels are the even integers 0, 2, . . . , 5
import pandas as pd
import numpy as np

n = np.arange(0,51,2)
n

Entries = n*n-1
Entries

Series = pd.Series(Entries, index = n)
print(Series)
```

```
0       -1
2        3
4       15
6       35
8       63
10      99
12     143
14     195
16     255
18     323
20     399
22     483
24     575
26     675
28     783
30     899
32    1023
34    1155
36    1295
38    1443
40    1599
42    1763
44    1935
46    2115
48    2303
50    2499
dtype: int32
```

In [34]:
```
#Problem 2
## Load the data into a pandas DataFrame, using the column names in the file and the c
from matplotlib import pyplot as plt

crimedata = pd.read_csv("State_crime.csv", index_col = 'Year')
df = pd.DataFrame(crimedata)
print(crimedata.head(1))

##List out all column names (hint: call function colunms)
print( crimedata.columns )
```

```
        State  Data.Population  Data.Rates.Property.All  \
Year
1960  Alabama          3266740                   1035.4

      Data.Rates.Property.Burglary  Data.Rates.Property.Larceny  \
Year
1960                         355.9                        592.1

      Data.Rates.Property.Motor  Data.Rates.Violent.All  \
Year
1960                       87.3                   186.6

      Data.Rates.Violent.Assault  Data.Rates.Violent.Murder  \
Year
1960                       138.1                       12.4

      Data.Rates.Violent.Rape  Data.Rates.Violent.Robbery  \
Year
1960                      8.6                        27.5

      Data.Totals.Property.All  Data.Totals.Property.Burglary  \
Year
1960                     33823                          11626

      Data.Totals.Property.Larceny  Data.Totals.Property.Motor  \
Year
1960                         19344                        2853

      Data.Totals.Violent.All  Data.Totals.Violent.Assault  \
Year
1960                     6097                        4512

      Data.Totals.Violent.Murder  Data.Totals.Violent.Rape  \
Year
1960                         406                       281

      Data.Totals.Violent.Robbery
Year
1960                          898
Index(['State', 'Data.Population', 'Data.Rates.Property.All',
       'Data.Rates.Property.Burglary', 'Data.Rates.Property.Larceny',
       'Data.Rates.Property.Motor', 'Data.Rates.Violent.All',
       'Data.Rates.Violent.Assault', 'Data.Rates.Violent.Murder',
       'Data.Rates.Violent.Rape', 'Data.Rates.Violent.Robbery',
       'Data.Totals.Property.All', 'Data.Totals.Property.Burglary',
       'Data.Totals.Property.Larceny', 'Data.Totals.Property.Motor',
       'Data.Totals.Violent.All', 'Data.Totals.Violent.Assault',
       'Data.Totals.Violent.Murder', 'Data.Totals.Violent.Rape',
       'Data.Totals.Violent.Robbery'],
      dtype='object')
```

In [9]:

```
## Drop all columns with “Rates” as a part of the names
crimedata = pd.read_csv("State_crime.csv", index_col = 'Year')
df = pd.DataFrame(crimedata)
df.drop (['Data.Rates.Property.All', 'Data.Rates.Property.Burglary',
          'Data.Rates.Property.Larceny','Data.Rates.Property.Motor' ,
          'Data.Rates.Violent.All','Data.Rates.Violent.Assault','Data.Rates.Violent.Mu
          'Data.Rates.Violent.Rape', 'Data.Rates.Violent.Robbery'], axis = 1)
```

Out[9]:

Year	State	Data.Population	Data.Totals.Property.All	Data.Totals.Property.Burglary	Data.Totals.Pr
1960	Alabama	3266740	33823	11626	
1961	Alabama	3302000	32541	11205	
1962	Alabama	3358000	35829	11722	
1963	Alabama	3347000	38521	12614	
1964	Alabama	3407000	46290	15898	
...	...	...	...	...	
2015	Wyoming	586107	1054	11151	
2016	Wyoming	585501	11460	1771	
2017	Wyoming	579315	10604	1593	
2018	Wyoming	577737	10313	1525	
2019	Wyoming	578759	9093	1396	

3115 rows × 11 columns

In [45]:
```
## Rename “Data.Population” to “Population.” Rename the rest columns by removing “Data

df = df.iloc[ : , -9:]

df.rename(columns =  {
 "Data.Population" : "Population",
 "Data.Totals.Property.All" : "Property",
 "Data.Totals.Property.Burglary" : "Burglary",
 "Data.Totals.Property.Larceny" : "Larceny",
 "Data.Totals.Property.Motor" : "Motor",
 "Data.Totals.Violent.All" : "Violent.All",
 "Data.Totals.Violent.Assault" : "Assault",
 "Data.Totals.Violent.Murder" : "Murder",
 "Data.Totals.Violent.Rape" : "Rape",
 "Data.Totals.Violent.Robbery" : "Robbery",
 })
```

Out[45]:

Year	Property	Burglary	Larceny	Motor	Violent.All	Assault	Murder	Rape	Robbery
1960	33823	11626	19344	2853	6097	4512	406	281	898
1961	32541	11205	18801	2535	5564	4255	427	252	630
1962	35829	11722	21306	2801	5283	3995	316	218	754
1963	38521	12614	22874	3033	6115	4755	340	192	828
1964	46290	15898	26713	3679	7260	5555	316	397	992
...	...	...	...	...	...	...	...	...	...
2015	1054	11151	1762	8797	1302	59	16	173	125
2016	11460	1771	8889	800	1430	1146	20	205	59
2017	10604	1593	8232	779	1376	1022	15	263	76
2018	10313	1525	7949	839	1226	870	13	243	100
2019	9093	1396	6984	713	1258	854	13	324	67

3115 rows × 9 columns

In [10]:

```
##Insert a new column into the data frame that contains the property crime rate by yea
data = pd.read_csv("state_crime.csv", index_col = "Year")
df1 = pd.DataFrame(data)

df1.rename(columns = {
    "Data.Population" : "Poplulation", "Data.Totals.Property.All" : "Property.All"
})


new_col = df1['Data.Totals.Property.All'] / df1['Data.Population']
df1['Property.Crime.Per.Year'] = new_col
print(df1)
```

Year	State	Data.Population	Data.Rates.Property.All
1960	Alabama	3266740	1035.4
1961	Alabama	3302000	985.5
1962	Alabama	3358000	1067.0
1963	Alabama	3347000	1150.9
1964	Alabama	3407000	1358.7
...	...	...	...
2015	Wyoming	586107	179.8
2016	Wyoming	585501	1957.3
2017	Wyoming	579315	1830.4
2018	Wyoming	577737	1785.1
2019	Wyoming	578759	1571.1

Year	Data.Rates.Property.Burglary	Data.Rates.Property.Larceny
1960	355.9	592.1
1961	339.3	569.4
1962	349.1	634.5
1963	376.9	683.4
1964	466.6	784.1
...	...	...
2015	1902.6	300.6
2016	302.5	1518.2
2017	275.0	1421.0
2018	264.0	1375.9
2019	241.2	1206.7

Year	Data.Rates.Property.Motor	Data.Rates.Violent.All
1960	87.3	186.6
1961	76.8	168.5
1962	83.4	157.3
1963	90.6	182.7
1964	108.0	213.1
...	...	...
2015	1500.9	222.1
2016	136.6	244.2
2017	134.5	237.5
2018	145.2	212.2
2019	123.2	217.4

Year	Data.Rates.Violent.Assault	Data.Rates.Violent.Murder
1960	138.1	12.4
1961	128.9	12.9
1962	119.0	9.4
1963	142.1	10.2
1964	163.0	9.3
...	...	...
2015	10.1	2.7
2016	195.7	3.4
2017	176.4	2.6
2018	150.6	2.3
2019	147.6	2.2

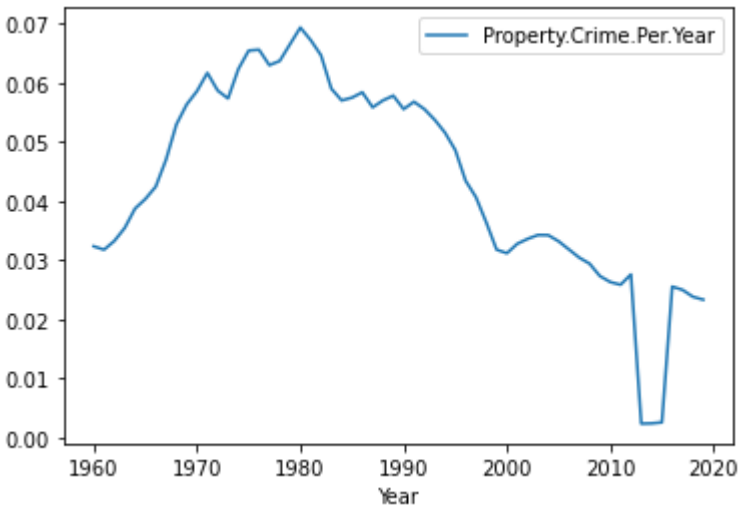
Year	Data.Rates.Violent.Rape	...	Data.Totals.Property.All
1960	8.6	...	33823
1961	7.6	...	32541

```
1962                    6.5  ...                    35829
1963                    5.7  ...                    38521
1964                   11.7  ...                    46290
...                     ...  ...                      ...
2015                   29.5  ...                     1054
2016                   35.0  ...                    11460
2017                   45.4  ...                    10604
2018                   42.1  ...                    10313
2019                   56.0  ...                     9093

      Data.Totals.Property.Burglary  Data.Totals.Property.Larceny  \
Year
1960                          11626                         19344
1961                          11205                         18801
1962                          11722                         21306
1963                          12614                         22874
1964                          15898                         26713
...                             ...                           ...
2015                          11151                          1762
2016                           1771                          8889
2017                           1593                          8232
2018                           1525                          7949
2019                           1396                          6984

      Data.Totals.Property.Motor  Data.Totals.Violent.All  \
Year
1960                        2853                     6097
1961                        2535                     5564
1962                        2801                     5283
1963                        3033                     6115
1964                        3679                     7260
...                          ...                      ...
2015                        8797                     1302
2016                         800                     1430
2017                         779                     1376
2018                         839                     1226
2019                         713                     1258

      Data.Totals.Violent.Assault  Data.Totals.Violent.Murder  \
Year
1960                         4512                        406
1961                         4255                        427
1962                         3995                        316
1963                         4755                        340
1964                         5555                        316
...                           ...                        ...
2015                           59                         16
2016                         1146                         20
2017                         1022                         15
2018                          870                         13
2019                          854                         13

      Data.Totals.Violent.Rape  Data.Totals.Violent.Robbery  \
Year
1960                       281                         898
1961                       252                         630
1962                       218                         754
1963                       192                         828
1964                       397                         992
...                        ...                         ...
```

```
2015                      173                       125
2016                      205                        59
2017                      263                        76
2018                      243                       100
2019                      324                        67

      Property.Crime.Per.Year
Year
1960                 0.010354
1961                 0.009855
1962                 0.010670
1963                 0.011509
1964                 0.013587
...                       ...
2015                 0.001798
2016                 0.019573
2017                 0.018304
2018                 0.017851
2019                 0.015711

[3115 rows x 21 columns]
```

In [42]:
```
##  Plot a line for the property crime rate of California as a function of the year.
df2 = df1.reset_index()
df3 = df2.set_index('State')

CA_crime = df3.loc['California']
CA_crime = CA_crime.set_index('Year')

CA_crime.plot(y= "Property.Crime.Per.Year")
```

Out[42]:
```
<AxesSubplot:xlabel='Year'>
```



In [48]:
```
##  List the 5 years with the highest property crime rate in descending order
df5 = df2.drop (['Data.Population','Data.Rates.Property.All','Data.Rates.Property.Burg
                 'Data.Rates.Property.Motor', 'Data.Rates.Violent.All', 'Data.Rates.Vio
                 'Data.Totals.Property.All', 'Data.Totals.Property.Burglary', 'Data.Tot
                  'Data.Totals.Violent.Assault', 'Data.Totals.Violent.Murder', 'Data.To

df5.sort_values(by="Property.Crime.Per.Year", ascending=False).head()
```

Out[48]:

	Year	State	Property.Crime.Per.Year
515	1995	District of Columbia	0.095121
516	1996	District of Columbia	0.094269
513	1993	District of Columbia	0.088393
490	1970	District of Columbia	0.086566
512	1992	District of Columbia	0.085742

In [ ]:
```
##  Calculate the average number of burglary crimes by states.
```

In [54]:
```
### Problem 3

pip install pydataset
```

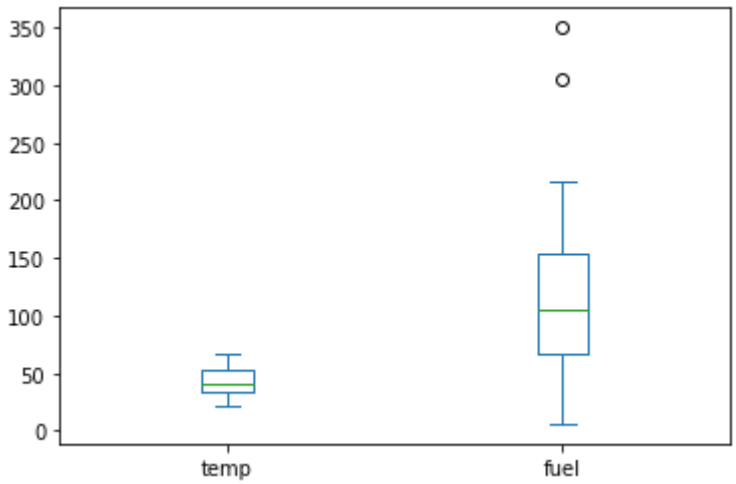
```
Collecting pydataset
  Downloading pydataset-0.2.0.tar.gz (15.9 MB)
Requirement already satisfied: pandas in c:\users\apple\onedrive\documents\python scr
ipts\lib\site-packages (from pydataset) (1.4.2)
Requirement already satisfied: pytz>=2020.1 in c:\users\apple\onedrive\documents\pyth
on scripts\lib\site-packages (from pandas->pydataset) (2021.3)
Requirement already satisfied: numpy>=1.18.5 in c:\users\apple\onedrive\documents\pyt
hon scripts\lib\site-packages (from pandas->pydataset) (1.21.5)
Requirement already satisfied: python-dateutil>=2.8.1 in c:\users\apple\onedrive\docu
ments\python scripts\lib\site-packages (from pandas->pydataset) (2.8.2)
Requirement already satisfied: six>=1.5 in c:\users\apple\onedrive\documents\python s
cripts\lib\site-packages (from python-dateutil>=2.8.1->pandas->pydataset) (1.16.0)
Building wheels for collected packages: pydataset
  Building wheel for pydataset (setup.py): started
  Building wheel for pydataset (setup.py): finished with status 'done'
  Created wheel for pydataset: filename=pydataset-0.2.0-py3-none-any.whl size=1593943
2 sha256=953adcf00973ff522c53bc1b0d8cb32ef33fcbcf866861db9c89e71be8452c73
  Stored in directory: c:\users\apple\appdata\local\pip\cache\wheels\6b\86\a7\f71cb84
c7bff804d83e293615a20c0531234397b796aee2645
Successfully built pydataset
Installing collected packages: pydataset
Successfully installed pydataset-0.2.0
Note: you may need to restart the kernel to use updated packages.
```

In [1]:
```
## Load dataset, "road", and show the data description. To see the information about '
from pydataset import data
data("road", show_doc=True)
```

```
road

PyDataset Documentation (adopted from R Documentation. The displayed examples are in
R)

##  Road Accident Deaths in US States

### Description

A data frame with the annual deaths in road accidents for half the US states.

### Usage

    road

### Format

Columns are:

`state`

name.

`deaths`

number of deaths.

`drivers`

number of drivers (in 10,000s).

`popden`

population density in people per square mile.

`rural`

length of rural roads, in 1000s of miles.

`temp`

average daily maximum temperature in January.

`fuel`

fuel consumption in 10,000,000 US gallons per year.

### Source

Imperial College, London M.Sc. exercise
```

In [5]:
```
 ##Generate box plots for temperature and fuel consumption. How many outliers are ther
from matplotlib import pyplot as plt

rd = data('road')
rd.head()

rd.plot(kind = "box", y = ["temp", "fuel"])
```

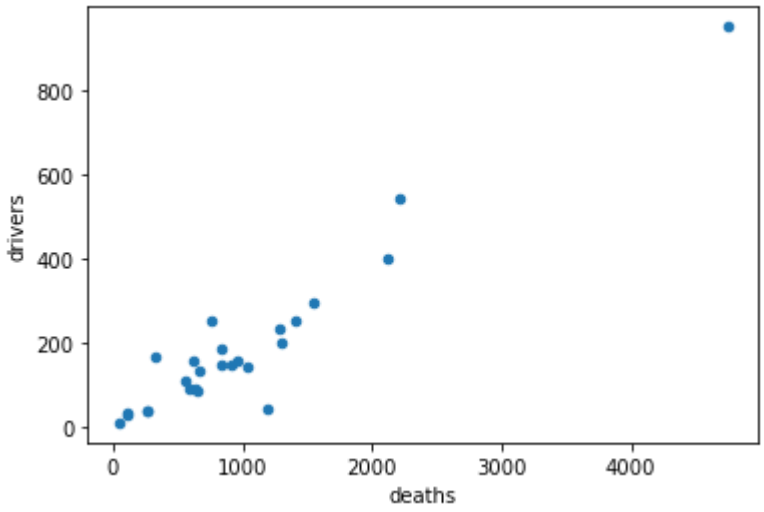
Out[5]:
```
<AxesSubplot:>
```



In [8]:
```
##  Provide a scatter plot to show the association between number of deaths and number
rd = data('road')
print(rd.head())

rd.plot(kind="scatter", x= "deaths", y= "drivers")
```

```
          deaths  drivers  popden  rural  temp   fuel
Alabama      968      158    64.0   66.0    62  119.0
Alaska        43       11     0.4    5.9    30    6.2
Arizona      588       91    12.0   33.0    64   65.0
Arkanas      640       92    34.0   73.0    51   74.0
Calif       4743      952   100.0  118.0    65  105.0
```

Out[8]:
```
<AxesSubplot:xlabel='deaths', ylabel='drivers'>
```



In [9]:
```
## Problem 4
data("titanic", show_doc= True)
```

```
titanic

PyDataset Documentation (adopted from R Documentation. The displayed examples are in
R)

## titanic

### Description

The data is an observation-based version of the 1912 Titanic passenger
survival log,

### Usage

    data(titanic)

### Format

A data frame with 1316 observations on the following 4 variables.

`class`

a factor with levels `1st class` `2nd class` `3rd class` `crew`

`age`

a factor with levels `child` `adults`

`sex`

a factor with levels `women` `man`

`survived`

a factor with levels `no` `yes`

### Details

titanic is saved as a data frame. Used to assess risk ratios

### Source

Found in many other texts

### References

Hilbe, Joseph M (2014), Modeling Count Data, Cambridge University Press Hilbe,
Joseph M (2007, 2011), Negative Binomial Regression, Cambridge University
Press Hilbe, Joseph M (2009), Logistic Regression Models, Chapman & Hall/CRC

### Examples

    data(titanic)
    titanic$survival <- titanic$survived == "yes"
    glmlr <- glm(survival ~ age + sex + factor(class), family=binomial, data=titanic)
    summary(glmlr)
```

In [20]:
```
# Create dummy variables of “survived”
```

```
import pandas as pd
td = data('titanic')


#td = pd.get_dummies(td, columns = ["survived"])
td.head()
```

Out[20]:

	class	age	sex	survived
1	1st class	adults	man	yes
2	1st class	adults	man	yes
3	1st class	adults	man	yes
4	1st class	adults	man	yes
5	1st class	adults	man	yes

In [19]:

```
## Create a pivot table using “sex” as the row, “class” as the column, and survival ra
td = data('titanic')

td.pivot_table(
    values=["survived" ],
    index = ["sex"],
    columns=["class"],
    aggfunc="max",
)
```

Out[19]:

	survived		
class	**1st class**	**2nd class**	**3rd class**
sex			
man	yes	yes	yes
women	yes	yes	yes

In [ ]: