In [2]:
```
import pandas as pd
```

In [3]:
```
makers = pd.Series(['BMW','Toyota','Honda','Toyota'])
```

In [4]:
```
makers
```

Out[4]:
```
0       BMW
1    Toyota
2     Honda
3    Toyota
dtype: object
```

In [5]:
```
makers.value_counts()
```

Out[5]:
```
Toyota    2
BMW       1
Honda     1
dtype: int64
```

In [6]:
```
colours = pd.Series(['red','yellow','black'])
```

In [7]:
```
colours
```

Out[7]:
```
0       red
1    yellow
2     black
dtype: object
```

In [8]:
```
# for data frame which is 2Dimensional
cars_data_1 = pd.DataFrame({'car maker':makers,'colour':colours })
```

In [9]:
```
cars_data_1
```

Out[9]:

	car maker	colour
0	BMW	red
1	Toyota	yellow
2	Honda	black
3	Toyota	NaN

In [10]:
```
#import data
df1 = pd.read_csv('car-sales.csv')
```

In [11]:
```
df1
```

Out[11]:

	Make	Colour	Odometer (KM)	Doors	Price
0	Toyota	White	150043	4	$4,000.00
1	Honda	Red	87899	4	$5,000.00
2	Toyota	Blue	32549	3	$7,000.00
3	BMW	Black	11179	5	$22,000.00
4	Nissan	White	213095	4	$3,500.00
5	Toyota	Green	99213	4	$4,500.00
6	Honda	Blue	45698	4	$7,500.00
7	Honda	Blue	54738	4	$7,000.00
8	Toyota	White	60000	4	$6,250.00
9	Nissan	White	31600	4	$9,700.00

In [137…
```
df1.to_csv('exporteed_car_sales.csv',index=False)
```

In [138…
```
df2 = pd.read_csv("exporteed_car_sales.csv")
```

In [139…
```
df2
```

Out[139]:

	Make	Colour	Odometer (KM)	Doors	Price	Seats	fuel per 100KM	total_fuel_used	Passes road safety
0	toyota	White	150043	4	4000	5.0	7.5	11253.225	True
1	honda	Red	87899	4	5000	5.0	9.4	8262.506	True
2	toyota	Blue	32549	3	7000	5.0	6.5	2115.685	True
3	bmw	Black	11179	5	22000	5.0	8.4	939.036	True
4	nissan	White	213095	4	3500	5.0	4.9	10441.655	True
5	toyota	Green	99213	4	4500	5.0	7.2	7143.336	True
6	honda	Blue	45698	4	7500	5.0	9.3	4249.914	True
7	honda	Blue	54738	4	7000	5.0	8.6	4707.468	True
8	toyota	White	60000	4	6250	5.0	5.1	3060.000	True
9	nissan	White	31600	4	9700	5.0	8.1	2559.600	True

Describe Data

In [15]:
```
#Attributes
df1.dtypes
```

Out[15]:
```
Make              object
Colour            object
Odometer (KM)      int64
Doors              int64
Price             object
dtype: object
```

In [16]:
```
#functions
# df1.to_csv()
```

In [17]:
```
df1.columns
```

Out[17]:
```
Index(['Make', 'Colour', 'Odometer (KM)', 'Doors', 'Price'], dtype='object')
```

In [18]:
```
df1.index
```

Out[18]:
```
RangeIndex(start=0, stop=10, step=1)
```

In [19]:
```
df1.describe()
```

Out[19]:

	Odometer (KM)	Doors
count	10.000000	10.000000
mean	78601.400000	4.000000
std	61983.471735	0.471405
min	11179.000000	3.000000
25%	35836.250000	4.000000
50%	57369.000000	4.000000
75%	96384.500000	4.000000
max	213095.000000	5.000000

In [20]:
```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 5 columns):
 #   Column          Non-Null Count  Dtype 
---  ------          --------------  ----- 
 0   Make            10 non-null     object
 1   Colour          10 non-null     object
 2   Odometer (KM)   10 non-null     int64 
 3   Doors           10 non-null     int64 
 4   Price           10 non-null     object
dtypes: int64(2), object(3)
memory usage: 528.0+ bytes
```

In [21]:
```
df1.mean()
```

```
C:\Users\Hanu\AppData\Local\Temp\ipykernel_17608\2053335143.py:1: FutureWarning: The default value of numeric_only in DataFrame.
mean is deprecated. In a future version, it will default to False. In addition, specifying 'numeric_only=None' is deprecated. Se
lect only valid columns or specify the value of numeric_only to silence this warning.
  df1.mean()
```

Out[21]:
```
Odometer (KM)    78601.4
Doors                4.0
dtype: float64
```

In [22]:
```
    #mean of series
    car_series = pd.Series([1003,456,4789,4456,876])
```

In [23]:
```
car_series
```

Out[23]:
```
0    1003
1     456
2    4789
3    4456
4     876
dtype: int64
```

In [24]:
```
car_series.mean()
```

Out[24]:
```
2316.0
```

In [25]:
```
df1.sum()
```

Out[25]:
```
Make             ToyotaHondaToyotaBMWNissanToyotaHondaHondaToyo...
Colour               WhiteRedBlueBlackWhiteGreenBlueBlueWhiteWhite
Odometer (KM)                                               786014
Doors                                                           40
Price            $4,000.00$5,000.00$7,000.00$22,000.00$3,500.00...
dtype: object
```

In [26]:
```
df1["Odometer (KM)"].sum()
```

Out[26]:
```
786014
```

In [27]:
```
len(df1)
```

Out[27]:
```
10
```

In [28]:
```
#selecting and viewing data
```

In [29]:
```
df1.head(7)
```

Out[29]:

	Make	Colour	Odometer (KM)	Doors	Price
0	Toyota	White	150043	4	$4,000.00
1	Honda	Red	87899	4	$5,000.00
2	Toyota	Blue	32549	3	$7,000.00
3	BMW	Black	11179	5	$22,000.00
4	Nissan	White	213095	4	$3,500.00
5	Toyota	Green	99213	4	$4,500.00
6	Honda	Blue	45698	4	$7,500.00

In [30]:
```
df1.tail(6)
```

Out[30]:

	Make	Colour	Odometer (KM)	Doors	Price
4	Nissan	White	213095	4	$3,500.00
5	Toyota	Green	99213	4	$4,500.00
6	Honda	Blue	45698	4	$7,500.00
7	Honda	Blue	54738	4	$7,000.00
8	Toyota	White	60000	4	$6,250.00
9	Nissan	White	31600	4	$9,700.00

In [31]:
```
#.loc & .iloc
animals = pd.Series(['dog','cat','elephant','zebra','panda'])
```

In [32]:
```
animals
```

Out[32]:
```
0         dog
1         cat
2    elephant
3       zebra
4       panda
dtype: object
```

In [33]:
```
animals = pd.Series(['dog','cat','elephant','zebra','panda'], index=[8,40,5,7,40])
```

In [34]:
```
animals
```

Out[34]:
```
8          dog
40         cat
5     elephant
7        zebra
40       panda
dtype: object
```

In [35]:
```
animals.loc[40]
```

Out[35]:
```
40      cat
40    panda
dtype: object
```

In [36]:
```
animals.loc[5]
```

Out[36]:
```
'elephant'
```

In [37]:
```
df1.loc[3]
```

Out[37]:
```
Make                    BMW
Colour                Black
Odometer (KM)         11179
Doors                     5
Price           $22,000.00
Name: 3, dtype: object
```

In [38]:
```
#iloc
animals.iloc[0]
```

Out[38]:
```
'dog'
```

we can see that from above example that iloc refers to position no matter if you have already changed the index

In [39]:
```
df1.iloc[3]
```

Out[39]:
```
Make                  BMW
Colour              Black
Odometer (KM)       11179
Doors                   5
Price            $22,000.00
Name: 3, dtype: object
```

In [40]:
```
df1.iloc[:4]
```

Out[40]:

	Make	Colour	Odometer (KM)	Doors	Price
0	Toyota	White	150043	4	$4,000.00
1	Honda	Red	87899	4	$5,000.00
2	Toyota	Blue	32549	3	$7,000.00
3	BMW	Black	11179	5	$22,000.00

In [41]:
```
df1.loc[:4]
```

Out[41]:

	Make	Colour	Odometer (KM)	Doors	Price
0	Toyota	White	150043	4	$4,000.00
1	Honda	Red	87899	4	$5,000.00
2	Toyota	Blue	32549	3	$7,000.00
3	BMW	Black	11179	5	$22,000.00
4	Nissan	White	213095	4	$3,500.00

In [42]:
```
df1['Colour']
# or
df1.Colour
```

Out[42]:
```
0    White
1      Red
2     Blue
3    Black
4    White
5    Green
6     Blue
7     Blue
8    White
9    White
Name: Colour, dtype: object
```

In [43]:
```
df1['Make']
#or
df1.Make
```

Out[43]:
```
0    Toyota
1     Honda
2    Toyota
3       BMW
4    Nissan
5    Toyota
6     Honda
7     Honda
8    Toyota
9    Nissan
Name: Make, dtype: object
```

In [44]:
```
#df1.Odometer (KM) this will give error

df1['Odometer (KM)']
```

Out[44]:
```
0    150043
1     87899
2     32549
3     11179
4    213095
5     99213
6     45698
7     54738
8     60000
9     31600
Name: Odometer (KM), dtype: int64
```

In [45]:
```
df1[df1['Make'] == 'Honda']
```

Out[45]:

	Make	Colour	Odometer (KM)	Doors	Price
1	Honda	Red	87899	4	$5,000.00
6	Honda	Blue	45698	4	$7,500.00
7	Honda	Blue	54738	4	$7,000.00

In [46]:
```
df1[df1['Make']=='Toyota']
```

Out[46]:

	Make	Colour	Odometer (KM)	Doors	Price
0	Toyota	White	150043	4	$4,000.00
2	Toyota	Blue	32549	3	$7,000.00
5	Toyota	Green	99213	4	$4,500.00
8	Toyota	White	60000	4	$6,250.00

In [47]:
```
df1[df1['Odometer (KM)']<=60000]
```

Out[47]:

	Make	Colour	Odometer (KM)	Doors	Price
2	Toyota	Blue	32549	3	$7,000.00
3	BMW	Black	11179	5	$22,000.00
6	Honda	Blue	45698	4	$7,500.00
7	Honda	Blue	54738	4	$7,000.00
8	Toyota	White	60000	4	$6,250.00
9	Nissan	White	31600	4	$9,700.00

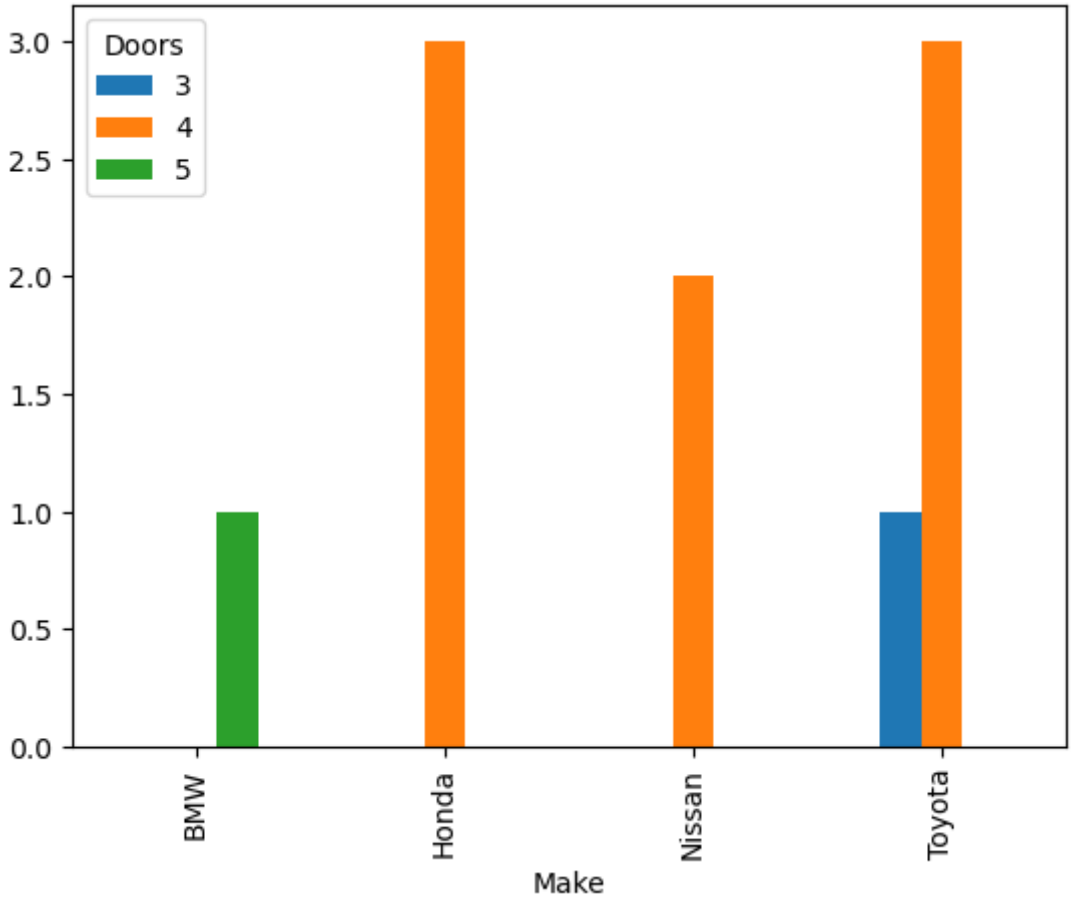
In [48]:
```
pd.crosstab(df1['Make'],df1['Doors'])
```

Out[48]:

Doors	3	4	5
Make			
BMW	0	0	1
Honda	0	3	0
Nissan	0	2	0
Toyota	1	3	0

In [49]:
```
pd.crosstab(df1['Make'],df1['Doors']).plot(kind='bar')
```

Out[49]:
```
<AxesSubplot: xlabel='Make'>
```



In [50]:
```
#group by
df1.groupby(['Make']).mean()
```

```
C:\Users\Hanu\AppData\Local\Temp\ipykernel_17608\3921110548.py:2: FutureWarning: The default value of numeric_only in DataFrameG
roupBy.mean is deprecated. In a future version, numeric_only will default to False. Either specify numeric_only or select only c
olumns which should be valid for the function.
  df1.groupby(['Make']).mean()
```

Out[50]:

Make	Odometer (KM)	Doors
BMW	11179.000000	5.00
Honda	62778.333333	4.00
Nissan	122347.500000	4.00
Toyota	85451.250000	3.75

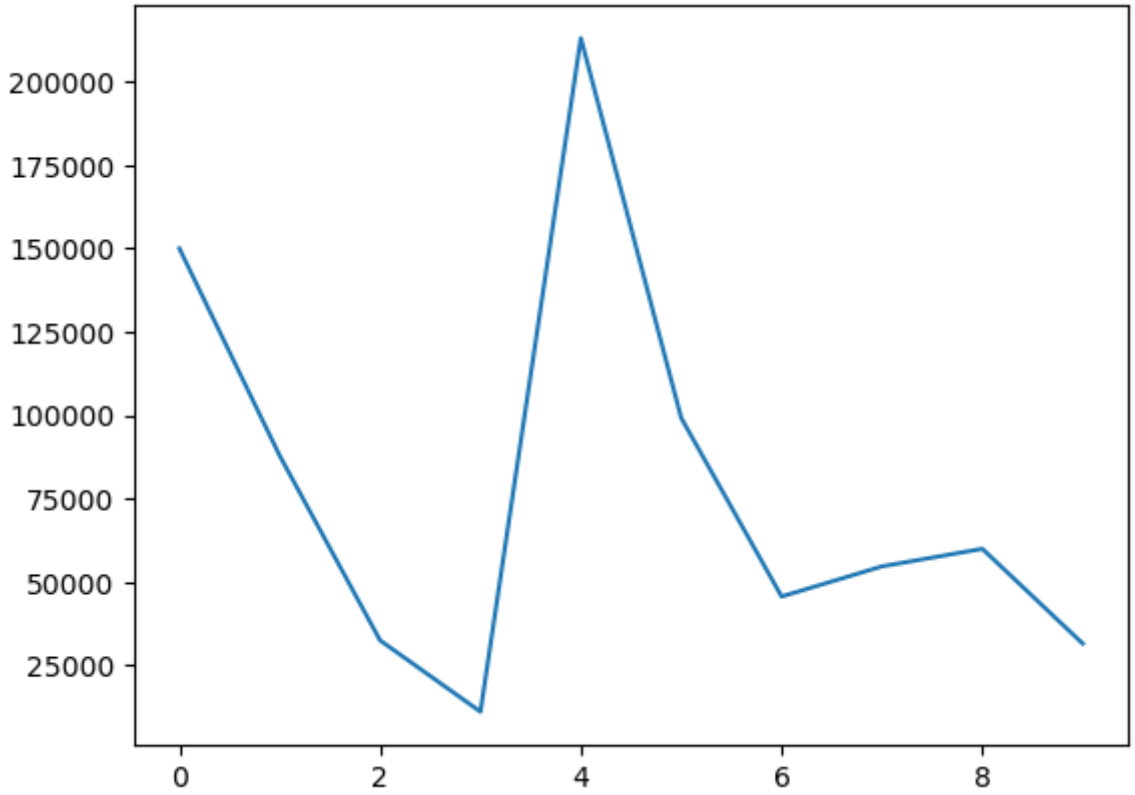
In [51]:
```
pd.crosstab(df1['Make'],df1['Colour'])
```

Out[51]:

Make	Black	Blue	Green	Red	White
BMW	1	0	0	0	0
Honda	0	2	0	1	0
Nissan	0	0	0	0	2
Toyota	0	1	1	0	2

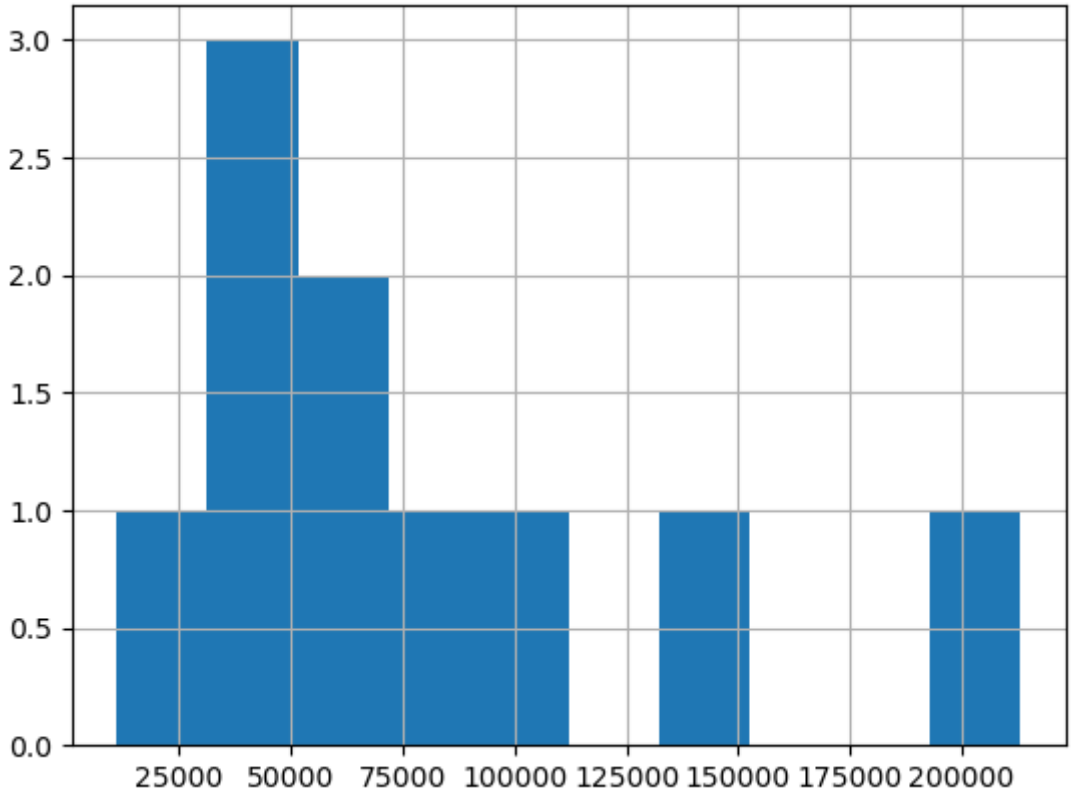
In [52]:
```
df1['Odometer (KM)'].plot()
```

Out[52]:
```
<AxesSubplot: >
```



In [53]:
```
df1['Odometer (KM)'].hist()
```

Out[53]:
```
<AxesSubplot: >
```



In [54]:
```
df1['Price'] = df1['Price'].replace('[\$\,\.]', '', regex=True).astype(int)
```

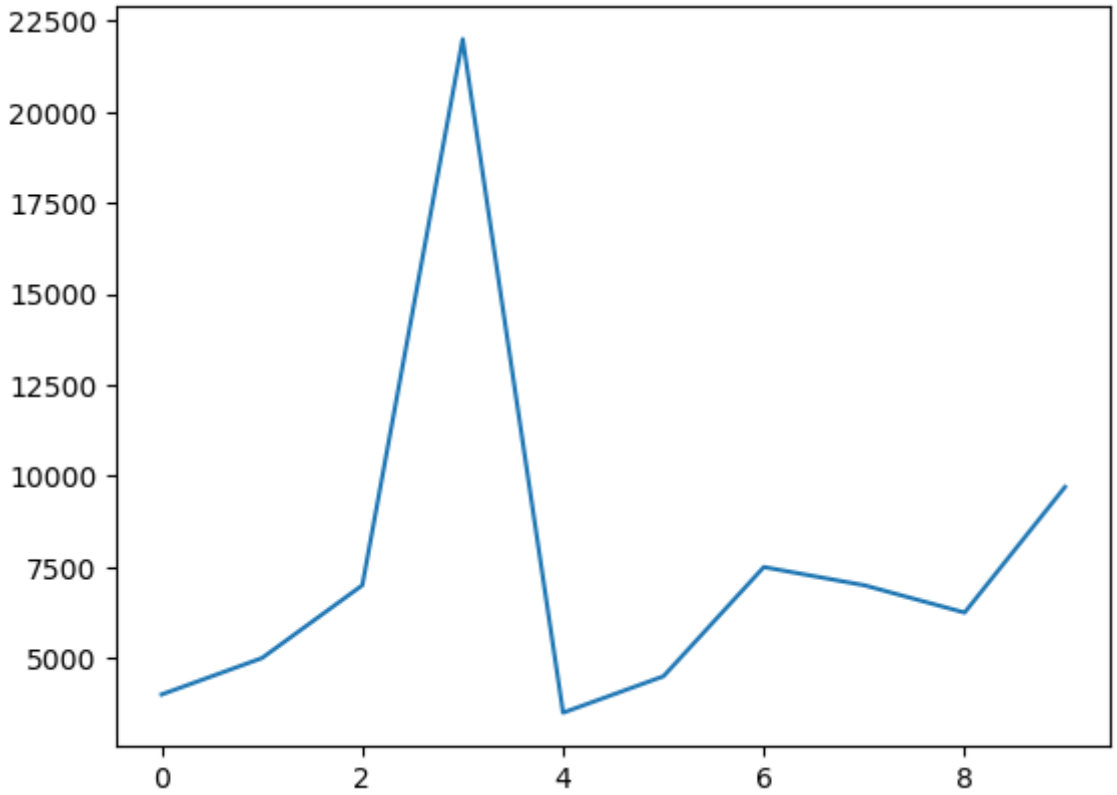
In [55]:
```
df1['Price'] = df1['Price']//100
```

In [56]:
```
df1['Price']
```

Out[56]:
```
0     4000
1     5000
2     7000
3    22000
4     3500
5     4500
6     7500
7     7000
8     6250
9     9700
Name: Price, dtype: int32
```

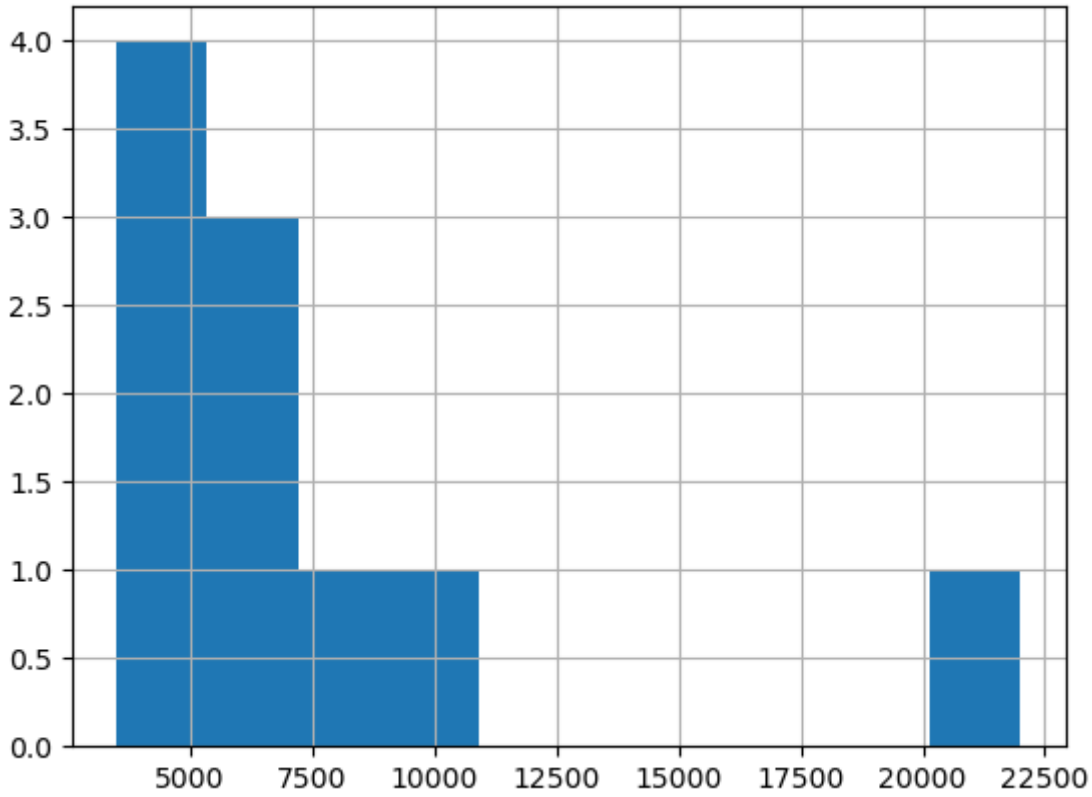
In [57]:
```
df1['Price'].plot()
```

Out[57]:
```
<AxesSubplot: >
```



In [58]:
```
df1['Price'].hist()
```

Out[58]: <AxesSubplot: >



Manipulating data

In [59]:
```
# temporary lowered the string
df1['Make'].str.lower()
```

Out[59]:
```
0    toyota
1     honda
2    toyota
3       bmw
4    nissan
5    toyota
6     honda
7     honda
8    toyota
9    nissan
Name: Make, dtype: object
```

In [60]:
```
# in pandas for saving the cahnges you have to reassign it
df1['Make'] = df1['Make'].str.lower()
```

In [61]:
```
df1['Make']
```

Out[61]:
```
0    toyota
1     honda
2    toyota
3       bmw
4    nissan
5    toyota
6     honda
7     honda
8    toyota
9    nissan
Name: Make, dtype: object
```

In [71]:
```
car_missing_data = pd.read_csv('car-sales-missing-data.csv')
```

In [72]:
```
car_missing_data.head(7)
```

Out[72]:

	Make	Colour	Odometer	Doors	Price
0	Toyota	White	150043.0	4.0	$4,000
1	Honda	Red	87899.0	4.0	$5,000
2	Toyota	Blue	NaN	3.0	$7,000
3	BMW	Black	11179.0	5.0	$22,000
4	Nissan	White	213095.0	4.0	$3,500
5	Toyota	Green	NaN	4.0	$4,500
6	Honda	NaN	NaN	4.0	$7,500

In [73]:
```
car_missing_data['Odometer'].mean()
```

Out[73]:
```
92302.66666666667
```

In [74]:
```
car_missing_data['Odometer'].fillna(car_missing_data['Odometer'].mean())
```

Out[74]:
```
0    150043.000000
1     87899.000000
2     92302.666667
3     11179.000000
4    213095.000000
5     92302.666667
6     92302.666667
7     92302.666667
8     60000.000000
9     31600.000000
Name: Odometer, dtype: float64
```

In [81]:
```
#we can see that the changes is not saved
car_missing_data
```

Out[81]:

	Make	Colour	Odometer	Doors	Price
0	Toyota	White	150043.000000	4.0	$4,000
1	Honda	Red	87899.000000	4.0	$5,000
2	Toyota	Blue	92302.666667	3.0	$7,000
3	BMW	Black	11179.000000	5.0	$22,000
4	Nissan	White	213095.000000	4.0	$3,500
5	Toyota	Green	92302.666667	4.0	$4,500
6	Honda	NaN	92302.666667	4.0	$7,500
7	Honda	Blue	92302.666667	4.0	NaN
8	Toyota	White	60000.000000	NaN	NaN
9	NaN	White	31600.000000	4.0	$9,700

In [82]:
```
# here we have used inplace for save the changes we have made
car_missing_data['Odometer'].fillna(car_missing_data['Odometer'].mean() , inplace=True)
```

In [83]:
```
car_missing_data
```

Out[83]:

	Make	Colour	Odometer	Doors	Price
0	Toyota	White	150043.000000	4.0	$4,000
1	Honda	Red	87899.000000	4.0	$5,000
2	Toyota	Blue	92302.666667	3.0	$7,000
3	BMW	Black	11179.000000	5.0	$22,000
4	Nissan	White	213095.000000	4.0	$3,500
5	Toyota	Green	92302.666667	4.0	$4,500
6	Honda	NaN	92302.666667	4.0	$7,500
7	Honda	Blue	92302.666667	4.0	NaN
8	Toyota	White	60000.000000	NaN	NaN
9	NaN	White	31600.000000	4.0	$9,700

In [87]:
```
#drop rows which consist na values
#we can also use inplace to make our change permanent i.e. car_missing_data.dropna(inplace=True)
car_missing_data.dropna()
```

In [94]:
```

```

Out[94]:

	Make	Colour	Odometer	Doors	Price
0	Toyota	White	150043.000000	4.0	$4,000
1	Honda	Red	87899.000000	4.0	$5,000
2	Toyota	Blue	92302.666667	3.0	$7,000
3	BMW	Black	11179.000000	5.0	$22,000
4	Nissan	White	213095.000000	4.0	$3,500
5	Toyota	Green	92302.666667	4.0	$4,500

In [96]:
```
#we are assigning the dropped dataset(drop na)in to another dataframe so that we can have our original and changed(where we dropp
car_missing_data = pd.read_csv('car-sales-missing-data.csv')
```

```
car_missing_data_dropped = car_missing_data.dropna()
```

In [101…
```
car_missing_data.head(8)
```

Out[101]:

	Make	Colour	Odometer	Doors	Price
0	Toyota	White	150043.0	4.0	$4,000
1	Honda	Red	87899.0	4.0	$5,000
2	Toyota	Blue	NaN	3.0	$7,000
3	BMW	Black	11179.0	5.0	$22,000
4	Nissan	White	213095.0	4.0	$3,500
5	Toyota	Green	NaN	4.0	$4,500
6	Honda	NaN	NaN	4.0	$7,500
7	Honda	Blue	NaN	4.0	NaN

In [109…
```
# for adding column in datasets
seat_column = pd.Series([5,5,5,5,5])
df1['Seats'] = seat_column
```

In [110…
```
df1
```

Out[110]:

	Make	Colour	Odometer (KM)	Doors	Price	Seats
0	toyota	White	150043	4	4000	5.0
1	honda	Red	87899	4	5000	5.0
2	toyota	Blue	32549	3	7000	5.0
3	bmw	Black	11179	5	22000	5.0
4	nissan	White	213095	4	3500	5.0
5	toyota	Green	99213	4	4500	NaN
6	honda	Blue	45698	4	7500	NaN
7	honda	Blue	54738	4	7000	NaN
8	toyota	White	60000	4	6250	NaN
9	nissan	White	31600	4	9700	NaN

In [111…

```
#but, this will only fill top 5 cells of the'seat' column of our dataset, for filling the other remaining cells we use fillna
#replace the na with value 5 and use inplace to save change
df1.fillna(5 , inplace=True)
```

In [112…

```
df1
```

Out[112]:

	Make	Colour	Odometer (KM)	Doors	Price	Seats
0	toyota	White	150043	4	4000	5.0
1	honda	Red	87899	4	5000	5.0
2	toyota	Blue	32549	3	7000	5.0
3	bmw	Black	11179	5	22000	5.0
4	nissan	White	213095	4	3500	5.0
5	toyota	Green	99213	4	4500	5.0
6	honda	Blue	45698	4	7500	5.0
7	honda	Blue	54738	4	7000	5.0
8	toyota	White	60000	4	6250	5.0
9	nissan	White	31600	4	9700	5.0

In [113…
```
# column using python list
fuel = [7.5,9.4,6.5,8.4,4.9,7.2,9.3,8.6,5.1,8.1]
df1['fuel per 100KM'] = fuel
```

In [114…
```
df1
```

Out[114]:

	Make	Colour	Odometer (KM)	Doors	Price	Seats	fuel per 100KM
0	toyota	White	150043	4	4000	5.0	7.5
1	honda	Red	87899	4	5000	5.0	9.4
2	toyota	Blue	32549	3	7000	5.0	6.5
3	bmw	Black	11179	5	22000	5.0	8.4
4	nissan	White	213095	4	3500	5.0	4.9
5	toyota	Green	99213	4	4500	5.0	7.2
6	honda	Blue	45698	4	7500	5.0	9.3
7	honda	Blue	54738	4	7000	5.0	8.6
8	toyota	White	60000	4	6250	5.0	5.1
9	nissan	White	31600	4	9700	5.0	8.1

In [115…
```
# above, we uses list to make column but the length of list and dataset should be equal otherwise it will show error
```

In [116…
```
df1['total_fuel_used']= (df1['Odometer (KM)']/100)*df1['fuel per 100KM']
```

In [117…
```
df1
```

Out[117]:

	Make	Colour	Odometer (KM)	Doors	Price	Seats	fuel per 100KM	total_fuel_used
0	toyota	White	150043	4	4000	5.0	7.5	11253.225
1	honda	Red	87899	4	5000	5.0	9.4	8262.506
2	toyota	Blue	32549	3	7000	5.0	6.5	2115.685
3	bmw	Black	11179	5	22000	5.0	8.4	939.036
4	nissan	White	213095	4	3500	5.0	4.9	10441.655
5	toyota	Green	99213	4	4500	5.0	7.2	7143.336
6	honda	Blue	45698	4	7500	5.0	9.3	4249.914
7	honda	Blue	54738	4	7000	5.0	8.6	4707.468
8	toyota	White	60000	4	6250	5.0	5.1	3060.000
9	nissan	White	31600	4	9700	5.0	8.1	2559.600

In [118…
```
df1['wheels'] = 4
```

In [119…
```
df1
```

Out[119]:

	Make	Colour	Odometer (KM)	Doors	Price	Seats	fuel per 100KM	total_fuel_used	wheels
0	toyota	White	150043	4	4000	5.0	7.5	11253.225	4
1	honda	Red	87899	4	5000	5.0	9.4	8262.506	4
2	toyota	Blue	32549	3	7000	5.0	6.5	2115.685	4
3	bmw	Black	11179	5	22000	5.0	8.4	939.036	4
4	nissan	White	213095	4	3500	5.0	4.9	10441.655	4
5	toyota	Green	99213	4	4500	5.0	7.2	7143.336	4
6	honda	Blue	45698	4	7500	5.0	9.3	4249.914	4
7	honda	Blue	54738	4	7000	5.0	8.6	4707.468	4
8	toyota	White	60000	4	6250	5.0	5.1	3060.000	4
9	nissan	White	31600	4	9700	5.0	8.1	2559.600	4

In [121…

```
df1['Passes road safety'] = True
df1
```

Out[121]:

	Make	Colour	Odometer (KM)	Doors	Price	Seats	fuel per 100KM	total_fuel_used	wheels	Passes road safety
0	toyota	White	150043	4	4000	5.0	7.5	11253.225	4	True
1	honda	Red	87899	4	5000	5.0	9.4	8262.506	4	True
2	toyota	Blue	32549	3	7000	5.0	6.5	2115.685	4	True
3	bmw	Black	11179	5	22000	5.0	8.4	939.036	4	True
4	nissan	White	213095	4	3500	5.0	4.9	10441.655	4	True
5	toyota	Green	99213	4	4500	5.0	7.2	7143.336	4	True
6	honda	Blue	45698	4	7500	5.0	9.3	4249.914	4	True
7	honda	Blue	54738	4	7000	5.0	8.6	4707.468	4	True
8	toyota	White	60000	4	6250	5.0	5.1	3060.000	4	True
9	nissan	White	31600	4	9700	5.0	8.1	2559.600	4	True

In [124…
```
df1.dtypes
```

Out[124]:
```
Make                  object
Colour                object
Odometer (KM)          int64
Doors                  int64
Price                  int32
Seats                float64
fuel per 100KM       float64
total_fuel_used      float64
wheels                 int64
Passes road safety      bool
dtype: object
```

In [132…
```
df1.drop('wheels',axis=1,inplace=True)
```

In [133…
```
df1
```

Out[133]:

	Make	Colour	Odometer (KM)	Doors	Price	Seats	fuel per 100KM	total_fuel_used	Passes road safety
0	toyota	White	150043	4	4000	5.0	7.5	11253.225	True
1	honda	Red	87899	4	5000	5.0	9.4	8262.506	True
2	toyota	Blue	32549	3	7000	5.0	6.5	2115.685	True
3	bmw	Black	11179	5	22000	5.0	8.4	939.036	True
4	nissan	White	213095	4	3500	5.0	4.9	10441.655	True
5	toyota	Green	99213	4	4500	5.0	7.2	7143.336	True
6	honda	Blue	45698	4	7500	5.0	9.3	4249.914	True
7	honda	Blue	54738	4	7000	5.0	8.6	4707.468	True
8	toyota	White	60000	4	6250	5.0	5.1	3060.000	True
9	nissan	White	31600	4	9700	5.0	8.1	2559.600	True

In [145…
```
df1_shuffled = df1.sample(frac=1)
```

In [146…
```
df1_shuffled
```

Out[146]:

	Make	Colour	Odometer (KM)	Doors	Price	Seats	fuel per 100KM	total_fuel_used	Passes road safety
5	toyota	Green	99213	4	4500	5.0	7.2	7143.336	True
1	honda	Red	87899	4	5000	5.0	9.4	8262.506	True
0	toyota	White	150043	4	4000	5.0	7.5	11253.225	True
9	nissan	White	31600	4	9700	5.0	8.1	2559.600	True
4	nissan	White	213095	4	3500	5.0	4.9	10441.655	True
7	honda	Blue	54738	4	7000	5.0	8.6	4707.468	True
8	toyota	White	60000	4	6250	5.0	5.1	3060.000	True
6	honda	Blue	45698	4	7500	5.0	9.3	4249.914	True
2	toyota	Blue	32549	3	7000	5.0	6.5	2115.685	True
3	bmw	Black	11179	5	22000	5.0	8.4	939.036	True

In [149…

```
df1_shuffled.reset_index(drop=True)
```

Out[149]:

	Make	Colour	Odometer (KM)	Doors	Price	Seats	fuel per 100KM	total_fuel_used	Passes road safety
0	toyota	Green	99213	4	4500	5.0	7.2	7143.336	True
1	honda	Red	87899	4	5000	5.0	9.4	8262.506	True
2	toyota	White	150043	4	4000	5.0	7.5	11253.225	True
3	nissan	White	31600	4	9700	5.0	8.1	2559.600	True
4	nissan	White	213095	4	3500	5.0	4.9	10441.655	True
5	honda	Blue	54738	4	7000	5.0	8.6	4707.468	True
6	toyota	White	60000	4	6250	5.0	5.1	3060.000	True
7	honda	Blue	45698	4	7500	5.0	9.3	4249.914	True
8	toyota	Blue	32549	3	7000	5.0	6.5	2115.685	True
9	bmw	Black	11179	5	22000	5.0	8.4	939.036	True

In [150…
```
df1['Odometer (Miles)'] = df1['Odometer (KM)'].apply(lambda x:x/1.6)
```

In [151…
```
df1
```

Out[151]:

	Make	Colour	Odometer (KM)	Doors	Price	Seats	fuel per 100KM	total_fuel_used	Passes road safety	Odometer (Miles)
0	toyota	White	150043	4	4000	5.0	7.5	11253.225	True	93776.875
1	honda	Red	87899	4	5000	5.0	9.4	8262.506	True	54936.875
2	toyota	Blue	32549	3	7000	5.0	6.5	2115.685	True	20343.125
3	bmw	Black	11179	5	22000	5.0	8.4	939.036	True	6986.875
4	nissan	White	213095	4	3500	5.0	4.9	10441.655	True	133184.375
5	toyota	Green	99213	4	4500	5.0	7.2	7143.336	True	62008.125
6	honda	Blue	45698	4	7500	5.0	9.3	4249.914	True	28561.250
7	honda	Blue	54738	4	7000	5.0	8.6	4707.468	True	34211.250
8	toyota	White	60000	4	6250	5.0	5.1	3060.000	True	37500.000
9	nissan	White	31600	4	9700	5.0	8.1	2559.600	True	19750.000

In [2]:
```
pip install pandoc
```

```
Collecting pandoc
  Downloading pandoc-2.3.tar.gz (33 kB)
  Preparing metadata (setup.py): started
  Preparing metadata (setup.py): finished with status 'done'
Collecting plumbum
  Downloading plumbum-1.8.1-py3-none-any.whl (126 kB)
     ------------------------------------ 126.7/126.7 kB 1.9 MB/s eta 0:00:00
Requirement already satisfied: ply in c:\users\hanu\desktop\sample_folder\sample_folder_1\env\lib\site-packages (from pandoc)
(3.11)
Requirement already satisfied: pywin32 in c:\users\hanu\desktop\sample_folder\sample_folder_1\env\lib\site-packages (from plumbu
m->pandoc) (305.1)
Building wheels for collected packages: pandoc
  Building wheel for pandoc (setup.py): started
  Building wheel for pandoc (setup.py): finished with status 'done'
  Created wheel for pandoc: filename=pandoc-2.3-py3-none-any.whl size=33290 sha256=63dbd414eee84cfd0b5f290fec4ff0b1b9e87f8bc54b1
3fecd408fc593939a90
  Stored in directory: c:\users\hanu\appdata\local\pip\cache\wheels\df\68\e2\574df0737a398965be3a1977499bbda7a841a4605d8dda34d2
Successfully built pandoc
Installing collected packages: plumbum, pandoc
Successfully installed pandoc-2.3 plumbum-1.8.1
Note: you may need to restart the kernel to use updated packages.
```

In [ ]: