Importing Necessary Libraries

```
import pandas as pd
import numpy as np
```

Read CSV File

```
auto = pd.read_csv("assignment1.csv")
```

.head() is used to display first 5 rows

```
auto.head()
```

```
   3    ?  alfa-romero  gas  std   two  convertible  rwd  front  88.6
...  \
0  3    ?  alfa-romero  gas  std   two  convertible  rwd  front  88.6
...
1  1    ?  alfa-romero  gas  std   two    hatchback  rwd  front  94.5
...
2  2  164         audi  gas  std  four        sedan  fwd  front  99.8
...
3  2  164         audi  gas  std  four        sedan  4wd  front  99.4
...
4  2    ?         audi  gas  std   two        sedan  fwd  front  99.8
...

   130  mpfi  3.47  2.68     9  111  5000  21  27  13495
0  130  mpfi  3.47  2.68   9.0  111  5000  21  27  16500
1  152  mpfi  2.68  3.47   9.0  154  5000  19  26  16500
2  109  mpfi  3.19   3.4  10.0  102  5500  24  30  13950
3  136  mpfi  3.19   3.4   8.0  115  5500  18  22  17450
4  136  mpfi  3.19   3.4   8.5  110  5500  19  25  15250

[5 rows x 26 columns]
```

.tail() is used to display last 5 rows

```
auto.tail()
```

```
     3   ? alfa-romero     gas    std   two convertible  rwd  front
88.6  \
199 -1  95       volvo     gas    std  four       sedan  rwd  front
109.1
200 -1  95       volvo     gas  turbo  four       sedan  rwd  front
109.1
201 -1  95       volvo     gas    std  four       sedan  rwd  front
109.1
202 -1  95       volvo  diesel  turbo  four       sedan  rwd  front
109.1
203 -1  95       volvo     gas  turbo  four       sedan  rwd  front
```

```
109.1

     ...  130  mpfi  3.47  2.68     9  111  5000  21  27  13495
199  ...  141  mpfi  3.78  3.15   9.5  114  5400  23  28  16845
200  ...  141  mpfi  3.78  3.15   8.7  160  5300  19  25  19045
201  ...  173  mpfi  3.58  2.87   8.8  134  5500  18  23  21485
202  ...  145   idi  3.01   3.4  23.0  106  4800  26  27  22470
203  ...  141  mpfi  3.78  3.15   9.5  114  5400  19  25  22625

[5 rows x 26 columns]
```

.shape is used to display the number of Rows and Columns

```
auto.shape

(204, 26)
```

.size is used to display number of cells

```
auto.size

5304
```

.describe() is used to return the Description of the Data

```
auto.describe()

                3        88.6       168.8        64.1        48.8  \
count  204.000000  204.000000  204.000000  204.000000  204.000000
mean     0.823529   98.806373  174.075000   65.916667   53.749020
std      1.239035    5.994144   12.362123    2.146716    2.424901
min     -2.000000   86.600000  141.100000   60.300000   47.800000
25%      0.000000   94.500000  166.300000   64.075000   52.000000
50%      1.000000   97.000000  173.200000   65.500000   54.100000
75%      2.000000  102.400000  183.200000   66.900000   55.500000
max      3.000000  120.900000  208.100000   72.300000   59.800000

             2548         130           9          21          27
count   204.000000  204.000000  204.000000  204.000000  204.000000
mean   2555.602941  126.892157   10.148137   25.240196   30.769608
std     521.960820   41.744569    3.981000    6.551513    6.898337
min    1488.000000   61.000000    7.000000   13.000000   16.000000
25%    2145.000000   97.000000    8.575000   19.000000   25.000000
50%    2414.000000  119.500000    9.000000   24.000000   30.000000
75%    2939.250000  142.000000    9.400000   30.000000   34.500000
max    4066.000000  326.000000   23.000000   49.000000   54.000000
```

.iloc[:] is used to display selected rows and columns

```
auto.iloc[:]
```

```
     3    ?  alfa-romero     gas    std   two  convertible  rwd  front
88.6  \
0    3    ?  alfa-romero     gas    std   two  convertible  rwd  front
88.6
1    1    ?  alfa-romero     gas    std   two    hatchback  rwd  front
94.5
2    2  164         audi     gas    std  four        sedan  fwd  front
99.8
3    2  164         audi     gas    std  four        sedan  4wd  front
99.4
4    2    ?         audi     gas    std   two        sedan  fwd  front
99.8
..  ..  ...          ...     ...    ...   ...          ...  ...    ...
...
199 -1   95        volvo     gas    std  four        sedan  rwd  front
109.1
200 -1   95        volvo     gas  turbo  four        sedan  rwd  front
109.1
201 -1   95        volvo     gas    std  four        sedan  rwd  front
109.1
202 -1   95        volvo  diesel  turbo  four        sedan  rwd  front
109.1
203 -1   95        volvo     gas  turbo  four        sedan  rwd  front
109.1

     ...  130  mpfi  3.47  2.68     9  111  5000  21  27  13495
0    ...  130  mpfi  3.47  2.68   9.0  111  5000  21  27  16500
1    ...  152  mpfi  2.68  3.47   9.0  154  5000  19  26  16500
2    ...  109  mpfi  3.19   3.4  10.0  102  5500  24  30  13950
3    ...  136  mpfi  3.19   3.4   8.0  115  5500  18  22  17450
4    ...  136  mpfi  3.19   3.4   8.5  110  5500  19  25  15250
..   ...  ...   ...   ...   ...   ...  ...   ...  ..  ..    ...
199  ...  141  mpfi  3.78  3.15   9.5  114  5400  23  28  16845
200  ...  141  mpfi  3.78  3.15   8.7  160  5300  19  25  19045
201  ...  173  mpfi  3.58  2.87   8.8  134  5500  18  23  21485
202  ...  145   idi  3.01   3.4  23.0  106  4800  26  27  22470
203  ...  141  mpfi  3.78  3.15   9.5  114  5400  19  25  22625

[204 rows x 26 columns]
```

Output of a Single Column

```
auto.iloc[:,2]

0      alfa-romero
1      alfa-romero
2             audi
3             audi
```

```
4              audi
          ...
199           volvo
200           volvo
201           volvo
202           volvo
203           volvo
Name: alfa-romero, Length: 204, dtype: object
```

Output of a Single Row

```
auto.iloc[3,:]
```

```
3                   2
?                 164
alfa-romero      audi
gas               gas
std               std
two              four
convertible     sedan
rwd               4wd
front           front
88.6             99.4
168.8           176.6
64.1             66.4
48.8             54.3
2548             2824
dohc              ohc
four             five
130               136
mpfi             mpfi
3.47             3.19
2.68              3.4
9                 8.0
111               115
5000             5500
21                 18
27                 22
13495           17450
Name: 3, dtype: object
```

Adding Header to the File

```
header = [
    "symboling", "normalized-losses", "make", "fuel-type",
"aspiration", "num-of-doors", "body-style",
    "drive-wheels", "engine-location", "wheel-base", "length",
"width", "height", "curb-weight", "engine-type",
    "num-of-cylinders", "engine-size", "fuel-system",
"bore","stroke","compression-ratio","horsepower",
```

```
    "peak-rpm", "city-mpg","highway-mpg","price"
]

auto = pd.read_csv("assignment1.csv", names=header)

auto.head()
```

```
   symboling normalized-losses         make fuel-type aspiration num-
of-doors  \
0          3                  ?  alfa-romero       gas        std
two
1          3                  ?  alfa-romero       gas        std
two
2          1                  ?  alfa-romero       gas        std
two
3          2                164         audi       gas        std
four
4          2                164         audi       gas        std
four

    body-style drive-wheels engine-location  wheel-base  ...  engine-
size  \
0  convertible          rwd           front        88.6  ...
130
1  convertible          rwd           front        88.6  ...
130
2    hatchback          rwd           front        94.5  ...
152
3        sedan          fwd           front        99.8  ...
109
4        sedan          4wd           front        99.4  ...
136

   fuel-system  bore  stroke compression-ratio horsepower  peak-rpm
city-mpg  \
0         mpfi  3.47    2.68               9.0        111      5000
21
1         mpfi  3.47    2.68               9.0        111      5000
21
2         mpfi  2.68    3.47               9.0        154      5000
19
3         mpfi  3.19     3.4              10.0        102      5500
24
4         mpfi  3.19     3.4               8.0        115      5500
18

  highway-mpg  price
0          27  13495
1          27  16500
2          26  16500
```

```
3           30  13950
4           22  17450

[5 rows x 26 columns]
```

Replace ? to NaN

```
auto.replace('?', np.nan, inplace=True)
```

.isna() is used to display if there exists Missing Values

```
auto.isna().sum()
```

```
symboling             0
normalized-losses    41
make                  0
fuel-type             0
aspiration            0
num-of-doors          2
body-style            0
drive-wheels          0
engine-location       0
wheel-base            0
length                0
width                 0
height                0
curb-weight           0
engine-type           0
num-of-cylinders      0
engine-size           0
fuel-system           0
bore                  4
stroke                4
compression-ratio     0
horsepower            2
peak-rpm              2
city-mpg              0
highway-mpg           0
price                 4
dtype: int64
```

Check Data Types

```
auto.dtypes
```

```
symboling              int64
normalized-losses     object
make                  object
fuel-type             object
aspiration            object
num-of-doors          object
```

```
body-style             object
drive-wheels           object
engine-location        object
wheel-base            float64
length                float64
width                 float64
height                float64
curb-weight             int64
engine-type            object
num-of-cylinders       object
engine-size             int64
fuel-system            object
bore                   object
stroke                 object
compression-ratio     float64
horsepower             object
peak-rpm               object
city-mpg                int64
highway-mpg             int64
price                  object
dtype: object
```

Default data type is "object". We need to change it into wither int or float

```
# For Replacing NaN to mean
auto["normalized-losses"] = auto["normalized-losses"].astype(float)
auto["stroke"] = auto["stroke"].astype(float)
auto["bore"] = auto["bore"].astype(float)
auto["horsepower"] = auto["horsepower"].astype(float)
auto["peak-rpm"] = auto["peak-rpm"].astype(float)
```

Find Mean of the Column

```
mean_normalize_loss = auto["normalized-losses"].mean()
mean_stroke = auto["stroke"].mean()
mean_bore = auto["bore"].mean()
mean_horsepower = auto["horsepower"].mean()
mean_peak_rpm = auto["peak-rpm"].mean()
```

Replcaed Missing Value to Mean

```
auto["normalized-losses"].fillna(value=mean_normalize_loss,
inplace=True)
auto["stroke"].fillna(value=mean_stroke, inplace=True)
auto["bore"].fillna(value=mean_bore, inplace=True)
auto["horsepower"].fillna(value=mean_horsepower,inplace=True)
auto["peak-rpm"].fillna(value=mean_peak_rpm,inplace=True)

C:\Users\aakas\AppData\Local\Temp\ipykernel_12152\2477188144.py:1:
FutureWarning: A value is trying to be set on a copy of a DataFrame or
Series through chained assignment using an inplace method.
```

```
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
original object.


  auto["normalized-losses"].fillna(value=mean_normalize_loss,
inplace=True)
C:\Users\aakas\AppData\Local\Temp\ipykernel_12152\2477188144.py:2:
FutureWarning: A value is trying to be set on a copy of a DataFrame or
Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
original object.


  auto["stroke"].fillna(value=mean_stroke, inplace=True)
C:\Users\aakas\AppData\Local\Temp\ipykernel_12152\2477188144.py:3:
FutureWarning: A value is trying to be set on a copy of a DataFrame or
Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
original object.


  auto["bore"].fillna(value=mean_bore, inplace=True)
C:\Users\aakas\AppData\Local\Temp\ipykernel_12152\2477188144.py:4:
FutureWarning: A value is trying to be set on a copy of a DataFrame or
Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
```

```
original object.


  auto["horsepower"].fillna(value=mean_horsepower,inplace=True)
C:\Users\aakas\AppData\Local\Temp\ipykernel_12152\2477188144.py:5:
FutureWarning: A value is trying to be set on a copy of a DataFrame or
Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
original object.


  auto["peak-rpm"].fillna(value=mean_peak_rpm,inplace=True)
```

Finding Frequency

```
count_num_of_door = auto["num-of-doors"].value_counts()

print(count_num_of_door)

num-of-doors
four    114
two      89
Name: count, dtype: int64
```

Replace With Max Frequency

```
auto["num-of-doors"].replace(np.nan, inplace=True)

C:\Users\aakas\AppData\Local\Temp\ipykernel_12152\3632943159.py:1:
FutureWarning: Series.replace without 'value' and with non-dict-like
'to_replace' is deprecated and will raise in a future version.
Explicitly specify the new values instead.
  auto["num-of-doors"].replace(np.nan, inplace=True)
C:\Users\aakas\AppData\Local\Temp\ipykernel_12152\3632943159.py:1:
FutureWarning: A value is trying to be set on a copy of a DataFrame or
Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
original object.
```

```
  auto["num-of-doors"].replace(np.nan, inplace=True)
```

Removing Any Row with Missing Price

```
auto.dropna(subset=["price"], inplace=True)
```

Checking Missing Values

```
auto.isna().sum()
```

```
symboling            0
normalized-losses    0
make                 0
fuel-type            0
aspiration           0
num-of-doors         0
body-style           0
drive-wheels         0
engine-location      0
wheel-base           0
length               0
width                0
height               0
curb-weight          0
engine-type          0
num-of-cylinders     0
engine-size          0
fuel-system          0
bore                 0
stroke               0
compression-ratio    0
horsepower           0
peak-rpm             0
city-mpg             0
highway-mpg          0
price                0
dtype: int64
```

Find Maximum Length, Width and Height

```
max_length = auto["length"].max()
max_width = auto["width"].max()
max_height = auto["height"].max()

print(max_length, max_width, max_height)
```

```
208.1 72.0 59.8
```

Replacing with Length/Max_Length, Width/Max_Width, Height/Max_Height

```
auto["length"] = auto["length"]/max_length
auto["width"] = auto["width"]/max_width
auto["height"] = auto["height"]/max_height
```

Adding a Column for Kilometers per Litre

```
auto["city-kmpl"] = 235/auto["city-mpg"]
auto["highway-kmpl"] = 235/auto["city-mpg"]

auto.iloc[:]
```

```
     symboling  normalized-losses         make fuel-type aspiration  \
0            3              122.0  alfa-romero       gas        std
1            3              122.0  alfa-romero       gas        std
2            1              122.0  alfa-romero       gas        std
3            2              164.0         audi       gas        std
4            2              164.0         audi       gas        std
..         ...                ...          ...       ...        ...
200         -1               95.0        volvo       gas        std
201         -1               95.0        volvo       gas      turbo
202         -1               95.0        volvo       gas        std
203         -1               95.0        volvo    diesel      turbo
204         -1               95.0        volvo       gas      turbo

    num-of-doors   body-style drive-wheels engine-location  wheel-base
...  \
0            two  convertible          rwd           front        88.6
...
1            two  convertible          rwd           front        88.6
...
2            two    hatchback          rwd           front        94.5
...
3           four        sedan          fwd           front        99.8
...
4           four        sedan          4wd           front        99.4
...
..           ...          ...          ...             ...         ...
...
200         four        sedan          rwd           front       109.1
...
201         four        sedan          rwd           front       109.1
...
202         four        sedan          rwd           front       109.1
...
203         four        sedan          rwd           front       109.1
...
204         four        sedan          rwd           front       109.1
...
```

```
     bore  stroke  compression-ratio  horsepower peak-rpm city-mpg  \
0    3.47    2.68                9.0       111.0   5000.0       21
1    3.47    2.68                9.0       111.0   5000.0       21
2    2.68    3.47                9.0       154.0   5000.0       19
3    3.19    3.40               10.0       102.0   5500.0       24
4    3.19    3.40                8.0       115.0   5500.0       18
..    ...     ...                ...         ...      ...      ...
200  3.78    3.15                9.5       114.0   5400.0       23
201  3.78    3.15                8.7       160.0   5300.0       19
202  3.58    2.87                8.8       134.0   5500.0       18
203  3.01    3.40               23.0       106.0   4800.0       26
204  3.78    3.15                9.5       114.0   5400.0       19

     highway-mpg  price  city-kmpl  highway-kmpl
0             27  13495   8.928024      8.928024
1             27  16500   8.928024      8.928024
2             26  16500   8.077736      8.077736
3             30  13950  10.203456     10.203456
4             22  17450   7.652592      7.652592
..           ...    ...        ...           ...
200           28  16845   9.778312      9.778312
201           25  19045   8.077736      8.077736
202           23  21485   7.652592      7.652592
203           27  22470  11.053744     11.053744
204           25  22625   8.077736      8.077736

[201 rows x 28 columns]
```