Pandas Lab Exercise (Kaggle Automobile Dataset)

We shall now test your skills in using Pandas package. We will be using the [automobiles Dataset (https://www.kaggle.com/nisargpatel/automobiles/data)](https://www.kaggle.com/nisargpatel/automobiles/data) from Kaggle.

Answer each question asked below wrt the automobiles dataset. Load pandas as pd and upload the Automobile.csv file as auto

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

Load the Automobile dataset into variable "auto"

In [1]:
```
import pandas as pd
auto=pd.read_csv('Automobile.csv')
auto
```

Out[1]:

	symboling	normalized_losses	make	fuel_type	aspiration	number_of_doors	body_sty
0	3	168	alfa-romero	gas	std	two	convertib
1	3	168	alfa-romero	gas	std	two	convertib
2	1	168	alfa-romero	gas	std	two	hatchbac
3	2	164	audi	gas	std	four	seda
4	2	164	audi	gas	std	four	seda
...	...	...	...	...	...	...	
196	-1	95	volvo	gas	std	four	seda
197	-1	95	volvo	gas	turbo	four	seda
198	-1	95	volvo	gas	std	four	seda
199	-1	95	volvo	diesel	turbo	four	seda
200	-1	95	volvo	gas	turbo	four	seda

201 rows × 26 columns

Check the head of the DataFrame.

In [2]:
```
auto.head()
```

Out[2]:

	symboling	normalized_losses	make	fuel_type	aspiration	number_of_doors	body_style
0	3	168	alfa-romero	gas	std	two	convertible
1	3	168	alfa-romero	gas	std	two	convertible
2	1	168	alfa-romero	gas	std	two	hatchback
3	2	164	audi	gas	std	four	sedan
4	2	164	audi	gas	std	four	sedan

5 rows × 26 columns

How many rows and columns are there?

In [3]:
```
import pandas as pd

df = pd.read_csv('Automobile.csv')


rows = df.shape


print(f"Number of rows: {rows}")
```

```
Number of rows: (201, 26)
```

In [5]:
```
import pandas as pd


df = pd.read_csv('Automobile.csv')

columns = df.shape



print(f"Number of columns: {columns}")
```

```
Number of columns: (201, 26)
```

What is the average Price of all cars in the dataset?

In [6]:
```
import pandas as pd
df = pd.read_csv('Automobile.csv')

average_price = df['price'].mean()


print(f"The average price of all cars is: {average_price}")
```

```
The average price of all cars is: 13207.129353233831
```

Which is the cheapest make and costliest make of car in the lot?

In [9]:
```
import pandas as pd


df = pd.read_csv('Automobile.csv')


cheapest_car = df.loc[df['price'].idxmin()]




print(f"The cheapest car make is: {cheapest_car['make']} with a price of {ch


```

```
The cheapest car make is: subaru with a price of 5118
```

In [7]:
```
import pandas as pd


df = pd.read_csv('Automobile.csv')


costliest_car = df.loc[df['price'].idxmax()]


print(f"The costliest car make is: {costliest_car['make']} with a price of 


```

```
The costliest car make is: mercedes-benz with a price of 45400
```

How many cars have horsepower greater than 100?

In [10]:
```
import pandas as pd

df = pd.read_csv('Automobile.csv')

cars_with_horsepower_above_100 = df[df['horsepower'] > 100].shape[0]

print(f"The number of cars with horsepower greater than 100 is: {cars_with_h
```

```
The number of cars with horsepower greater than 100 is: 90
```

How many hatchback cars are in the dataset ?

In [11]:
```
import pandas as pd

df = pd.read_csv('Automobile.csv')

hatchback_cars = df[df['body_style'] == 'hatchback'].shape[0]

print(f"The number of hatchback cars in the dataset is: {hatchback_cars}")
```

```
The number of hatchback cars in the dataset is: 68
```

What are the 3 most commonly found cars in the dataset?

In [12]:
```
import pandas as pd

df = pd.read_csv('Automobile.csv')

car_make_counts = df['make'].value_counts()

top_3_most_common_cars = car_make_counts.head(3)

print(f"The 3 most commonly found car makes in the dataset are:")
print(top_3_most_common_cars)
```

```
The 3 most commonly found car makes in the dataset are:
toyota    32
nissan    18
mazda     17
Name: make, dtype: int64
```

Someone purchased a car for 7099, what is the make of the car?

In [13]:
```
import pandas as pd


df = pd.read_csv('Automobile.csv')


car_with_price_7099 = df[df['price'] == 7099]


if not car_with_price_7099.empty:
    make_of_car = car_with_price_7099['make'].iloc[0]
    print(f"The make of the car purchased for 7099 is: {make_of_car}")
else:
    print("No car was found with the price of 7099.")
```

```
The make of the car purchased for 7099 is: nissan
```

Which cars are priced greater than 40000?

In [15]:
```
import pandas as pd


df = pd.read_csv('Automobile.csv')


cars_above_40000 = df[df['price'] > 40000]


print("Cars priced greater than 40,000:")
print(cars_above_40000[['make',  'price']])
```

```
Cars priced greater than 40,000:
             make  price
15            bmw  41315
70  mercedes-benz  40960
71  mercedes-benz  45400
```

Which are the cars that are both a sedan and priced less than 7000?

In [17]:
```
import pandas as pd

df = pd.read_csv('Automobile.csv')

sedans_under_7000 = df[(df['body_style'] == 'sedan') & (df['price'] < 7000)]


print("Sedan cars priced less than 7000:")
print(sedans_under_7000[['make',  'price']])
```

```
Sedan cars priced less than 7000:
           make  price
19    chevrolet   6575
24        dodge   6692
42        isuzu   6785
50        mazda   6695
82   mitsubishi   6989
86       nissan   5499
88       nissan   6649
89       nissan   6849
118    plymouth   6692
152      toyota   6938
```

Count the number of unique values in the `fuel_type` column.

In [18]:
```
import pandas as pd

df = pd.read_csv('Automobile.csv')


unique_fuel_types = df['fuel_type'].nunique()


print(f"The number of unique values in the 'fuel_type' column is: {unique_fu
```

```
The number of unique values in the 'fuel_type' column is: 2
```

List all the cars that have a horsepower between 100 and 200, and display their `make`, `horsepower`, and `price`.

In [19]:
```
import pandas as pd


df = pd.read_csv('Automobile.csv')


cars_with_horsepower_100_200 = df[(df['horsepower'] >= 100) & (df['horsepowe


print("Cars with horsepower between 100 and 200:")
print(cars_with_horsepower_100_200[['make', 'horsepower', 'price']])
```

```
Cars with horsepower between 100 and 200:
            make  horsepower  price
0    alfa-romero         111  13495
1    alfa-romero         111  16500
2    alfa-romero         154  16500
3           audi         102  13950
4           audi         115  17450
..           ...         ...    ...
196        volvo         114  16845
197        volvo         160  19045
198        volvo         134  21485
199        volvo         106  22470
200        volvo         114  22625

[88 rows x 3 columns]
```

Find the average `city_mpg` and `highway_mpg` for each `body_style` .

In [20]:
```
import pandas as pd


df = pd.read_csv('Automobile.csv')


average_mpg = df.groupby('body_style')[['city_mpg', 'highway_mpg']].mean()


print("Average city_mpg and highway_mpg for each body_style:")
print(average_mpg)
```

```
Average city_mpg and highway_mpg for each body_style:
              city_mpg  highway_mpg
body_style
convertible  20.500000    26.000000
hardtop      21.625000    27.250000
hatchback    26.602941    32.382353
sedan        25.053191    30.574468
wagon        24.040000    28.720000
```

What is the median `price` for each `make` ?

In [21]:
```
import pandas as pd


df = pd.read_csv('Automobile.csv')


median_price_by_make = df.groupby('make')['price'].median()


print("Median price for each make:")
print(median_price_by_make)
```

```
Median price for each make:
make
alfa-romero      16500.0
audi             17580.0
bmw              22835.0
chevrolet         6295.0
dodge             7609.0
honda             7295.0
isuzu             8916.5
jaguar           35550.0
mazda            10595.0
mercedes-benz    32892.0
mercury          16503.0
mitsubishi        8499.0
nissan            8124.0
peugot           16630.0
plymouth          7609.0
porsche          33278.0
renault           9595.0
saab             15275.0
subaru            7894.0
toyota            9103.0
volkswagen        9737.5
volvo            18420.0
Name: price, dtype: float64
```

List all cars that have a `wheel_base` greater than 100 and a `curb_weight` less than 2500.

In [23]:
```
import pandas as pd


df = pd.read_csv('Automobile.csv')

cars_filtered = df[(df['wheel_base'] > 100) & (df['curb_weight'] < 2500)]


print("Cars with wheel_base greater than 100 and curb_weight less than 2500
print(cars_filtered[['make', 'wheel_base', 'curb_weight', 'price']])
```

```
Cars with wheel_base greater than 100 and curb_weight less than 2500:
       make  wheel_base  curb_weight  price
9       bmw       101.2         2395  16430
10      bmw       101.2         2395  16925
169  toyota       102.4         2326   8948
170  toyota       102.4         2480  10698
171  toyota       102.4         2414   9988
172  toyota       102.4         2414  10898
173  toyota       102.4         2458  11248
```

Create a new column `price_per_hp` that calculates the price of the car per horsepower.

In [24]:
```
import pandas as pd

df = pd.read_csv('Automobile.csv')


df['price_per_hp'] = df['price'] / df['horsepower']


print("Dataset with the new 'price_per_hp' column:")
print(df[['make', 'horsepower', 'price', 'price_per_hp']])
```

```
Dataset with the new 'price_per_hp' column:
            make  horsepower  price  price_per_hp
0    alfa-romero         111  13495    121.576577
1    alfa-romero         111  16500    148.648649
2    alfa-romero         154  16500    107.142857
3           audi         102  13950    136.764706
4           audi         115  17450    151.739130
..           ...         ...    ...           ...
196        volvo         114  16845    147.763158
197        volvo         160  19045    119.031250
198        volvo         134  21485    160.335821
199        volvo         106  22470    211.981132
200        volvo         114  22625    198.464912

[201 rows x 4 columns]
```

Count how many cars have a `number_of_doors` as `four`.

In [25]:
```
import pandas as pd

df = pd.read_csv('Automobile.csv')


cars_with_four_doors = df[df['number_of_doors'] == 'four']


count_four_doors = cars_with_four_doors.shape[0]

print(f"Number of cars with four doors: {count_four_doors}")
```

```
Number of cars with four doors: 114
```

Find the top 5 cars based on their `highway_mpg` and `price`.

In [27]:
```
import pandas as pd


df = pd.read_csv('Automobile.csv')


top_cars = df.sort_values(by=['highway_mpg', 'price'], ascending=[False, Fa

top_5_cars = top_cars.head(5)


print("Top 5 cars based on highway_mpg and price:")
print(top_5_cars[['make', 'highway_mpg', 'price']])
```

```
Top 5 cars based on highway_mpg and price:
          make  highway_mpg  price
29       honda           54   6479
17   chevrolet           53   5151
87      nissan           50   7099
155     toyota           47   7788
156     toyota           47   7738
```

How many cars have missing values in the `normalized_losses` column?

In [28]:
```
import pandas as pd


df = pd.read_csv('Automobile.csv')


missing_normalized_losses = df['normalized_losses'].isnull().sum()

print(f"Number of cars with missing values in 'normalized_losses': {missing_
```

```
Number of cars with missing values in 'normalized_losses': 0
```

Create a new column `car_age` that calculates the age of the car based on the `year_of_manufacture` (assume the current year is 2025).

In [31]:
```
import pandas as pd


df = pd.read_csv('Automobile.csv')

current_year = 2025


df['car_age'] = current_year - df['year_of_manufacture']


print("Dataset with the new 'car_age' column:")
print(df[['make','year_of_manfacture',  'car_age']])
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last
t)
~\anaconda3\lib\site-packages\pandas\core\indexes\base.py in get_loc(self,
key, method, tolerance)
   3628             try:
-> 3629                 return self._engine.get_loc(casted_key)
   3630             except KeyError as err:

~\anaconda3\lib\site-packages\pandas\_libs\index.pyx in pandas._libs.inde
x.IndexEngine.get_loc()

~\anaconda3\lib\site-packages\pandas\_libs\index.pyx in pandas._libs.inde
x.IndexEngine.get_loc()

pandas\_libs\hashtable_class_helper.pxi in pandas._libs.hashtable.PyObject
HashTable.get_item()

pandas\_libs\hashtable_class_helper.pxi in pandas._libs.hashtable.PyObject
HashTable.get_item()

KeyError: 'year_of_manufacture'

The above exception was the direct cause of the following exception:

KeyError                                  Traceback (most recent call last
t)
~\AppData\Local\Temp\ipykernel_6312\3558602481.py in <module>
      7
      8
----> 9 df['car_age'] = current_year - df['year_of_manufacture']
     10
     11

~\anaconda3\lib\site-packages\pandas\core\frame.py in __getitem__(self, ke
y)
   3503             if self.columns.nlevels > 1:
   3504                 return self._getitem_multilevel(key)
-> 3505             indexer = self.columns.get_loc(key)
   3506             if is_integer(indexer):
   3507                 indexer = [indexer]

~\anaconda3\lib\site-packages\pandas\core\indexes\base.py in get_loc(self,
key, method, tolerance)
   3629                 return self._engine.get_loc(casted_key)
   3630             except KeyError as err:
-> 3631                 raise KeyError(key) from err
   3632             except TypeError:
   3633                 # If we have a listlike key, _check_indexing_error
will raise

KeyError: 'year_of_manufacture'
```

The END