Import and Inspect Dataset | DataFrame | Pandas

```
import pandas as pd
```

Problem 1 - Import MPG dataset as car dataframe

car dataset url : https://github.com/YBI-Foundation/Dataset/raw/main/MPG.csv

Solution 1

```
car = pd.read_csv('https://github.com/YBI-Foundation/Dataset/raw/main/MPG.csv')
```

Problem 2 - Print the dataframe

Solution 2

```
car
```

Problem 3 - Inspect first 10 rows

Solution 3

```
car.head(10)
```

Problem 4 - Inspect last 5 rows

Solution 4

```
car.tail() #default 5 rows
```

Problem 5 - View all rows

Solution 5

```
pd.options.display.max_rows = 400
car
```

Problem 6 - How many missing values

Solution 6

```
car.isna().sum()
```

```
mpg             0
cylinders       0
displacement    0
horsepower      6
weight          0
acceleration    0
model_year      0
origin          0
name            0
dtype: int64
```

Problem 7 - Drop all missing values

Solution 7

```
car = car.dropna()
```

```
car.isna().sum()
```

```
mpg             0
cylinders       0
displacement    0
horsepower      0
weight          0
acceleration    0
model_year      0
origin          0
name            0
dtype: int64
```

Problem 8 - Summary statistics

Solution 8

```
car.describe()
```

Problem 9 - Data type in each column

Solution 9

```
car.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 392 entries, 0 to 397
Data columns (total 9 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   mpg           392 non-null    float64
 1   cylinders     392 non-null    int64  
 2   displacement  392 non-null    float64
 3   horsepower    392 non-null    float64
 4   weight        392 non-null    int64  
 5   acceleration  392 non-null    float64
 6   model_year    392 non-null    int64  
 7   origin        392 non-null    object 
 8   name          392 non-null    object 
dtypes: float64(4), int64(3), object(2)
memory usage: 30.6+ KB
```

Problem 10 - Shape of dataframe (How many rows and columns in 2D array)

Solution 10

```
car.shape
```

```
(392, 9)
```

Analysing Columns | DataFrame | Pandas

Problem 1 - Import MPG dataset and store as the pandas datafra,e with name mpg

Solution 1

```
mpg = pd.read_csv('https://github.com/YBI-Foundation/Dataset/raw/main/MPG.csv')
```

```
mpg
```

Problem 2 - Copy mpg dataframe as car

Solution 2

```
car = mpg.copy()
```

```
car
```

Problem 3 - Drop column name cylinders from original dataframe(mpg) and inspect what

Solution 3

```
mpg = mpg.drop('cylinders', axis = 1)
```

```
mpg.columns
```

```
Index(['mpg', 'displacement', 'horsepower', 'weight', 'acceleration',
       'model_year', 'origin', 'name'],
      dtype='object')
```

```
car.columns
```

```
Index(['mpg', 'cylinders', 'displacement', 'horsepower', 'weight',
       'acceleration', 'model_year', 'origin', 'name'],
      dtype='object')
```

Problem 4 - Analyse dataframe car

Solution 4

```
car.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 398 entries, 0 to 397
Data columns (total 9 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   mpg           398 non-null    float64
 1   cylinders     398 non-null    int64  
 2   displacement  398 non-null    float64
 3   horsepower    392 non-null    float64
 4   weight        398 non-null    int64  
 5   acceleration  398 non-null    float64
 6   model_year    398 non-null    int64  
 7   origin        398 non-null    object 
 8   name          398 non-null    object 
dtypes: float64(4), int64(3), object(2)
memory usage: 28.1+ KB
```

```
car.describe()
```

Problem 5 - Provide unique values in each columns cylinders and origin

Solution 5

```
car[['cylinders', 'origin']].value_counts()
```

```
cylinders  origin
8          usa       103
6          usa        74
4          usa        72
           japan      69
           europe     63
6          japan       6
3          japan       4
6          europe      4
5          europe      3
dtype: int64
```

Problem 6 - Provide unique values of column origin

Solution 6

```
car[['origin']].value_counts()
```

```
origin
usa       249
japan      79
europe     70
dtype: int64
```

```
car['origin'].unique()
```

```
array(['usa', 'japan', 'europe'], dtype=object)
```

```
car['origin'].nunique()
```

```
3
```

Problem 7 - Sort value of car dataframe as per displacemnet column

Solution 7

```
car.sort_values('displacement')
```

Problem 8 - Sort value of car dataframe as per displacement column in descending order

Solution 8

```
car.sort_values('displacement', ascending = False)
```

Problem 9 - Sort value of car dataframe as per displacement and weight columns in descending order

Solution 9

```
car.sort_values(['displacement', 'weight'], ascending = False)
```

Problem 10 - Summary statistics of all columns

Solution 10

```
car.describe(include = 'all')
```

Problem 11 - Transpose dataframe

Solution 11

```
car.T
```

Indexing & Slicing | DataFrame | Pandas

Problem 1 - Import Titanic dataset and store as the pandas datframe with name titanic

Titanic dataset url : https://github.com/YBI-Foundation/Dataset/raw/main/Titanic.csv

Solution 1

```
titanic = pd.read_csv('https://github.com/YBI-Foundation/Dataset/raw/main/Titanic.csv')
```

Problem 2 - Print info of titanic

Solution 2

```
titanic.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1309 entries, 0 to 1308
Data columns (total 14 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   pclass     1309 non-null   int64  
 1   survived   1309 non-null   int64  
 2   name       1309 non-null   object 
 3   sex        1309 non-null   object 
 4   age        1046 non-null   float64
 5   sibsp      1309 non-null   int64  
 6   parch      1309 non-null   int64  
 7   ticket     1309 non-null   object 
 8   fare       1308 non-null   float64
 9   cabin      295 non-null    object 
 10  embarked   1307 non-null   object 
 11  boat       486 non-null    object 
 12  body       121 non-null    float64
 13  home.dest  745 non-null    object 
dtypes: float64(3), int64(4), object(7)
memory usage: 143.3+ KB
```

Problem 3 - Print column lables only

Solution 3

```
titanic.columns
```

```
Index(['pclass', 'survived', 'name', 'sex', 'age', 'sibsp', 'parch', 'ticket',
       'fare', 'cabin', 'embarked', 'boat', 'body', 'home.dest'],
      dtype='object')
```

Problem 4 - Select passengers name column

Solution 4

```
titanic.name
```

```
0                        Allen, Miss. Elisabeth Walton
1                       Allison, Master. Hudson Trevor
2                         Allison, Miss. Helen Loraine
3                 Allison, Mr. Hudson Joshua Creighton
4       Allison, Mrs. Hudson J C (Bessie Waldo Daniels)
                             ...
1304                           Zabour, Miss. Hileni
1305                          Zabour, Miss. Thamine
1306                       Zakarian, Mr. Mapriededer
1307                             Zakarian, Mr. Ortin
1308                              Zimmerman, Mr. Leo
Name: name, Length: 1309, dtype: object
```

Problem 5 - Select passengers name column as pandas series and save as name

Solution 5

```
name = titanic['name']
```

```
name
```

```
0                        Allen, Miss. Elisabeth Walton
1                       Allison, Master. Hudson Trevor
2                         Allison, Miss. Helen Loraine
3                 Allison, Mr. Hudson Joshua Creighton
4       Allison, Mrs. Hudson J C (Bessie Waldo Daniels)
                             ...
1304                           Zabour, Miss. Hileni
1305                          Zabour, Miss. Thamine
1306                       Zakarian, Mr. Mapriededer
1307                             Zakarian, Mr. Ortin
1308                              Zimmerman, Mr. Leo
Name: name, Length: 1309, dtype: object
```

```
type(name)
```

```
pandas.core.series.Series
```

```
name.shape
```

```
(1309,)
```

Problem 6 - Select passengers name column and save as pandas dataframe

Solution 6

```
name = titanic[['name']]
```

```
name
```

```
type(name)
```

```
pandas.core.frame.DataFrame
```

```
name.shape
```

```
(1309, 1)
```

Problem 7 - Select 100th row and all columns with iloc function

Solution 7

```
titanic.iloc[100, : ]
```

```
pclass                                                  1
survived                                                1
name         Duff Gordon, Sir. Cosmo Edmund ("Mr Morgan")
sex                                                  male
age                                                  49.0
sibsp                                                   1
parch                                                   0
ticket                                           PC 17485
fare                                              56.9292
cabin                                                 A20
embarked                                                C
boat                                                    1
body                                                  NaN
home.dest                                  London / Paris
Name: 100, dtype: object
```

Problem 8 - Select 100th row with loc function

Solution 8

```
titanic.loc[100, : ]
```

```
pclass                                                  1
survived                                                1
name         Duff Gordon, Sir. Cosmo Edmund ("Mr Morgan")
sex                                                  male
age                                                  49.0
sibsp                                                   1
parch                                                   0
ticket                                           PC 17485
fare                                              56.9292
cabin                                                 A20
embarked                                                C
boat                                                    1
body                                                  NaN
home.dest                                  London / Paris
Name: 100, dtype: object
```

Problem 9 - Select all rows with column label name and fare column with iloc function

Solution 9

```
titanic.iloc[:,[2,8]]
```

Problem 10 - Select all rows with loc function and column label name and fare

Solution 10

```
titanic.loc[:,['name', 'fare']]
```

Problem 11 - Select row number 50th, 25th, 15th and column label passenger class, fare, age with both both loc and iloc function

Solution 11

```
titanic.loc[[50,25,15], ['pclass', 'fare', 'age']]
```

```
titanic.iloc[[50,25,15], [0, 8, 4]]
```

Problem 12 - Select rows from 10th, 25th and column label passenger class, fare, age with both loc and iloc function

Solution 12

```
titanic.loc[10 : 25, ['pclass', 'fare', 'age']]
```

```
titanic.iloc[10:26, [0, 8, 4]]
```

Problem 13 - Select rows from 10th to 15th and columns from passenger class to age with both loc and iloc function

Solution 13

```
titanic.loc[10:15, 'pclass' : 'age']
```

```
titanic.iloc[10 : 16, 0 : 5]
```

Problem 14 - Select all passengers with age equal to and more than 35 years

Solution 14

```
titanic[titanic['age'] >= 35]
```

Problem 15 - Select all passengers with age equal to and more than 35 years and column with label passenger class to age

Solution 15

```
titanic.loc[(titanic['age'] >= 35), 'pclass' : 'age']
```

Problem 16 - Select all female passengers with age equal to and more than 35 years

Solution 16

```
titanic.loc[(titanic['age'] >= 35) & (titanic['sex'] == 'female')]
```

Calculated Columns | DataFrame | Pandas

Problem 1 - Import Tips datasets and store as the pandas datframe with name tips

Tips dataset url : https://github.com/YBI-Foundation/Dataset/raw/main/Tips%20Payment%20Data.csv

Solution 1

```
tips = pd.read_csv('https://github.com/YBI-Foundation/Dataset/raw/main/Tips%20Payment%20Da
```

Problem 2 - Display the first 5 rows of tips dataframe

Solution 2

```
tips.head()
```

Problem 3 - Calculate percentage of tip to total bill

Solution 3

```
tips['Tip'] / tips['Total Bill'] * 100
```

Problem 4 - Create a new column of percentage tip

Solution 4

```
tip_percentage = tips['Tip'] / tips['Total Bill'] * 100
```

```
tip_percentage
```

Problem 5 - Insert percentage tip in existing tips dataframe

Solution 5

```
tips['tips_percentage'] = tips['Tip'] / tips['Total Bill'] * 100
```

```
tips.head()
```

Problem 6 - Round upto one decimal place the tip_percentage column values

Solution 6

```
tips['tips_percentage'] = tips['tips_percentage'].round(1)
```

```
tips.head()
```

Problem 7 - Drop column Payer Number

Solution 7

```
tips = tips.drop(['Payer Name'], axis = 1)
```

```
tips.head()
```

Problem 8 - Index tips dataframe as per CC NUMBER

Solution 8

```
tips.set_index('CC Number')
```

Problem 9 - Change index tips dataframe as per CC Number

Solution 9

```
tips = tips.set_index('CC Number')
```

```
tips.head()
```

Problem 10 - Reset index of tips dataframe to row index

Solution 10

```
tips = tips.reset_index()
```

```
tips.head()
```

✓ 0s completed at 11:30 PM