In [1]:

```
# Fundamental Data Structure
# Array / list
# start and end with []
# can be numeric, object, boolean and mixed type
# mutable, unordered, duplicates are allowed
```

In [2]:

```
numlist = [100,56,12,16,19,45,78,90,4,25,100,100]
print(numlist)

objlist = ["AAA",'A',"john","create",'A','A']
print(objlist)

boollist = [True,False,False,True]
print(boollist)

mixedlist = [24,56,78,'AAA','CCC','DDD','Ant','Ant','Ant']
print(mixedlist)
```

```
[100, 56, 12, 16, 19, 45, 78, 90, 4, 25, 100, 100]
['AAA', 'A', 'john', 'create', 'A', 'A']
[True, False, False, True]
[24, 56, 78, 'AAA', 'CCC', 'DDD', 'Ant', 'Ant', 'Ant']
```

In [3]:

```
# Tuples:
# start and end with ()
# immutable(cannot append and edit)
# unordered and allowed duplicates
```

In [4]:

```
numtuple = (100,56,12,16,19,45,78,90,4,25,100,100)
print(numtuple)

objtuple = ("AAA",'A',"john","create",'A','A')
print(objtuple)

booltuple = (True,False,False,True)
print(booltuple)

mixedtuple = (24,56,78,'AAA','CCC','DDD','Ant','Ant','Ant')
print(mixedtuple)
```

```
(100, 56, 12, 16, 19, 45, 78, 90, 4, 25, 100, 100)
('AAA', 'A', 'john', 'create', 'A', 'A')
(True, False, False, True)
(24, 56, 78, 'AAA', 'CCC', 'DDD', 'Ant', 'Ant', 'Ant')
```

In [5]:

```
# Sets:
# starts and end with {} curly brackets
# do not allow duplicates
```

In [6]:

```
numset = {100, 56, 12, 16, 19, 45, 78, 90, 4, 25, 100, 100}
print(numset)

objsets = {"AAA",'A',"john","create",'A','A'}
print(objsets)

boolsets = {True,False,False,True}
print(boolsets)

mixedsets = {24,56,78,'AAA','CCC','DDD','Ant','Ant','Ant'}
print(mixedsets)
```

```
{100, 4, 12, 45, 78, 16, 19, 56, 25, 90}
{'AAA', 'john', 'A', 'create'}
{False, True}
{'DDD', 'CCC', 78, 'AAA', 'Ant', 24, 56}
```

In [7]:

```
# Dictionary
# start and end with {} but elements are key value pairs
# key must be object/string in quotes followed by colon
# value can be both numeric or object
```

In [8]:

```
dict = {'Male':1,'Femla':0}
print(dict)
```

```
{'Male': 1, 'Femla': 0}
```

In [9]:

```
# for accessing specific elements [] are used.
```

In [10]:

```
print(numlist[0])
print(numlist[5])

print(objtuple[3])

print(objtuple[2])
```

```
100
45
create
john
```

In [11]:

```
# for acccessing set Iterator must be used

for i in objsets:
    print(i)
```

```
AAA
john
A
create
```

In [12]:

```
# Predefined function syntax - print()
# predefined function are lowercase followed by ()  either paranthesis is null or argume
# Multiple arguments within paranthesis must be given as a list
```

In [13]:

```
# Python libraries or Modules for Data Science
# 1) Pandas: Library for dataframe based operations like read files, write files, concat
# Date function
# Dataframe is by defualt rows are observations and columns are variables

# 2) Matplotlib.pyplot: Data vistulization library
# we can alter the plot window size, create subplots, edit axes, etc.

# 3) NumPy: Array or list based operations. For running algorithms numpy is essential. S
# a) SciPy.stats - Statistical Analysis & Modelling
# b) statsmodels - IOT Data or Time series Data
# c) sklearn or scikit learn - Core library for Machine Learning


# Others libraries that need to be installed

# nltk, spacy, wordcloud, textblob - Natural Language Processing

# Tensorflow, keras - Deep Learning and Image Processing

# OpenCV - Computer Vision or Facial Recogniation, etc.
```

In [14]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

In [15]:

```
df = pd.read_csv('D:\Datasets\listings.csv')
```

In [16]:

```
df
```

Out[16]:

	id	name	host_id	host_name	neighbourhood_group	neig
0	2384	Hyde Park - Walk to UChicago	2613	Rebecca	NaN	
1	94450	Comfy Garden Suite in Andersonville	504470	Mark	NaN	
2	145659	Trendy Roscoe Village 3BR/2BR walk to shops	683529	Joe	NaN	N
3	7126	Tiny Studio Apartment 94 Walk Score	17928	Sarah	NaN	
4	189821	Best in Chicago, private, amazing garden space	899757	Meighan	NaN	Lo
...	...	...	...	...	...	
7742	807864161683006482	Chicago Summer Backyard Fun	112243295	Mike	NaN	
7743	807880884094680264	Loop 1br w/ pool, gym & lounge, nr Riverwalk	107434423	Blueground	NaN	
7744	808048225308192110	Stunning 2 Floor Penthouse Downtown - Sleeps 12	170785489	Dmd	NaN	
7745	808067261823221295	Fully furnished 2BR w/ Cozy Living Room & Wi-Fi!	490752114	Sevyn	NaN	F
7746	808108595377919122	Cloud9| Up to 14 | Full amenities | The Wrigley	248760412	Cloud9	NaN	

7747 rows × 18 columns

In [17]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7747 entries, 0 to 7746
Data columns (total 18 columns):
 #   Column                          Non-Null Count  Dtype  
---  ------                          --------------  -----  
 0   id                              7747 non-null   int64  
 1   name                            7747 non-null   object 
 2   host_id                         7747 non-null   int64  
 3   host_name                       7747 non-null   object 
 4   neighbourhood_group             0 non-null      float64
 5   neighbourhood                   7747 non-null   object 
 6   latitude                        7747 non-null   float64
 7   longitude                       7747 non-null   float64
 8   room_type                       7747 non-null   object 
 9   price                           7747 non-null   int64  
 10  minimum_nights                  7747 non-null   int64  
 11  number_of_reviews               7747 non-null   int64  
 12  last_review                     6254 non-null   object 
 13  reviews_per_month               6254 non-null   float64
 14  calculated_host_listings_count  7747 non-null   int64  
 15  availability_365                7747 non-null   int64  
 16  number_of_reviews_ltm           7747 non-null   int64  
 17  license                         6573 non-null   object 
dtypes: float64(4), int64(8), object(6)
memory usage: 1.1+ MB
```

In [18]:

```
df.head()
```

Out[18]:

	id	name	host_id	host_name	neighbourhood_group	neighbourhood	latitud
0	2384	Hyde Park - Walk to UChicago	2613	Rebecca	NaN	Hyde Park	41.7879
1	94450	Comfy Garden Suite in Andersonville	504470	Mark	NaN	Edgewater	41.9796
2	145659	Trendy Roscoe Village 3BR/2BR walk to shops	683529	Joe	NaN	North Center	41.9434
3	7126	Tiny Studio Apartment 94 Walk Score	17928	Sarah	NaN	West Town	41.9016
4	189821	Best in Chicago, private, amazing garden space	899757	Meighan	NaN	Logan Square	41.9291

In [19]:

```
df.tail()
```

Out[19]:

	id	name	host_id	host_name	neighbourhood_group	neighb
7742	807864161683006482	Chicago Summer Backyard Fun	112243295	Mike	NaN	Ir
7743	807880884094680264	Loop 1br w/ pool, gym & lounge, nr Riverwalk	107434423	Blueground	NaN	
7744	808048225308192110	Stunning 2 Floor Penthouse Downtown - Sleeps 12	170785489	Dmd	NaN	W
7745	808067261823221295	Fully furnished 2BR w/ Cozy Living Room & Wi-Fi!	490752114	Sevyn	NaN	Ro
7746	808108595377919122	Cloud9| Up to 14 | Full amenities | The Wrigley	248760412	Cloud9	NaN	W

In [20]:

```
df.shape
```

Out[20]:

```
(7747, 18)
```

In [21]:

```
df.columns
```

Out[21]:

```
Index(['id', 'name', 'host_id', 'host_name', 'neighbourhood_group',
       'neighbourhood', 'latitude', 'longitude', 'room_type', 'price',
       'minimum_nights', 'number_of_reviews', 'last_review',
       'reviews_per_month', 'calculated_host_listings_count',
       'availability_365', 'number_of_reviews_ltm', 'license'],
      dtype='object')
```

In [22]:

```
df.isnull().sum()
```

Out[22]:

```
id                                    0
name                                  0
host_id                               0
host_name                             0
neighbourhood_group                7747
neighbourhood                         0
latitude                              0
longitude                             0
room_type                             0
price                                 0
minimum_nights                        0
number_of_reviews                     0
last_review                        1493
reviews_per_month                  1493
calculated_host_listings_count        0
availability_365                      0
number_of_reviews_ltm                 0
license                            1174
dtype: int64
```

In [23]:

```
df.duplicated().sum()
```

Out[23]:

```
0
```

In [24]:

```
df.describe()
```

Out[24]:

	id	host_id	neighbourhood_group	latitude	longitude	
count	7.747000e+03	7.747000e+03	0.0	7747.000000	7747.000000	7747.(
mean	2.650872e+17	1.617357e+08	NaN	41.895250	-87.662637	184.2
std	3.448603e+17	1.526951e+08	NaN	0.061759	0.043208	1160.(
min	2.384000e+03	2.153000e+03	NaN	41.650640	-87.847243	0.(
25%	3.094478e+07	3.288698e+07	NaN	41.867725	-87.686305	77.(
50%	4.973334e+07	1.074344e+08	NaN	41.898470	-87.657760	124.(
75%	6.629074e+17	2.574644e+08	NaN	41.938337	-87.631890	189.(
max	8.495391e+17	5.056757e+08	NaN	42.022200	-87.529541	99998.(

In [25]:

```
df.nunique()
```

Out[25]:

```
id                                7747
name                              7239
host_id                           3590
host_name                         2020
neighbourhood_group                  0
neighbourhood                       76
latitude                          6107
longitude                         5857
room_type                            4
price                              623
minimum_nights                      56
number_of_reviews                  409
last_review                        874
reviews_per_month                  656
calculated_host_listings_count      38
availability_365                   366
number_of_reviews_ltm              114
license                           3983
dtype: int64
```

In [ ]:

```

```