Pandas

```
import pandas as pd
```

```
# DatFrame from lists.

dl = ["father", "mother", "son", "daughter", "uncle", "aunt"]
dfl = pd.DataFrame(dl)
dfl.columns = ["Relation"]
print(dfl)
```

```
   Relation
0    father
1    mother
2       son
3  daughter
4     uncle
5      aunt
```

```
# DataFrame from Tuples.

dt = ("Om", 77, 9.3,  "Shree")
dft = pd.DataFrame(dt)
dft.columns = ["Details"]
print(dft)
```

```
  Details
0      Om
1      77
2     9.3
3   Shree
```

```
# DataFrame from  range-guided sequnece

ds = [x for x in range(10, 20)]
dfs = pd.DataFrame(ds)
dfs.columns = ["Integers"]
print(dfs)
```

```
   Integers
0        10
1        11
2        12
3        13
4        14
5        15
6        16
7        17
8        18
9        19
```

```
# DataFrame from Dictionaries.

dd = {"name": ["Om", "Shree", "Sakshi", "Anand", "Raghav"],
      "roll" : [77, 72, 70, 54, 32],
      "cgpa" : [9.3, 8.6, 7.2, 6.4, 10],
      "department" : ["IT", "CSE", "CSSE", "CSCE", "Mechatronics"]
}
dfd = pd.DataFrame(dd)
print(dfd)
```

```
     name  roll  cgpa    department
0      Om    77   9.3            IT
1   Shree    72   8.6           CSE
2  Sakshi    70   7.2          CSSE
3   Anand    54   6.4          CSCE
4  Raghav    32  10.0  Mechatronics
```

```
# Print specific columns of a dataframe
print(dfd[["name", "cgpa"]])

# Iloc in pandas
```

```
row = dfd.iloc[1,2]
print(row)
```

```
     name  cgpa
0      Om   9.3
1   Shree   8.6
2  Sakshi   7.2
3   Anand   6.4
4  Raghav  10.0
8.6
```

```
print(type(dfl))
print(type(dft))
print(type(dfd))
print(type(dfs))
```

```
<class 'pandas.core.frame.DataFrame'>
<class 'pandas.core.frame.DataFrame'>
<class 'pandas.core.frame.DataFrame'>
<class 'pandas.core.frame.DataFrame'>
```

```
# DataFrame from CSV.
pokedf = pd.read_csv("pokemon.csv")
# DataFrame .head()
pokedf.head()
```

	abilities	against_bug	against_dark	against_dragon	against_electric	against_fairy	against_fight	against_fire	against_flying
0	['Overgrow', 'Chlorophyll']	1.0	1.0	1.0	0.5	0.5	0.5	2.0	2.0
1	['Overgrow', 'Chlorophyll']	1.0	1.0	1.0	0.5	0.5	0.5	2.0	2.0
2	['Overgrow', 'Chlorophyll']	1.0	1.0	1.0	0.5	0.5	0.5	2.0	2.0
3	['Blaze', 'Solar Power']	0.5	1.0	1.0	1.0	0.5	1.0	0.5	1.0
4	['Blaze', 'Solar Power']	0.5	1.0	1.0	1.0	0.5	1.0	0.5	1.0

5 rows × 41 columns

```
# DataFrame .tail()
pokedf.tail()
```

	abilities	against_bug	against_dark	against_dragon	against_electric	against_fairy	against_fight	against_fire	against_flying
796	['Beast Boost']	0.25	1.0	0.5	2.0	0.5	1.0	2.0	0.5
797	['Beast Boost']	1.00	1.0	0.5	0.5	0.5	2.0	4.0	1.0
798	['Beast Boost']	2.00	0.5	2.0	0.5	4.0	2.0	0.5	1.0
799	['Prism Armor']	2.00	2.0	1.0	1.0	1.0	0.5	1.0	1.0
800	['Soul-Heart']	0.25	0.5	0.0	1.0	0.5	1.0	2.0	0.5

5 rows × 41 columns

```
# Describe function
pokedf.describe()
```

	against_bug	against_dark	against_dragon	against_electric	against_fairy	against_fight	against_fire	against_flying	against_
count	801.000000	801.000000	801.000000	801.000000	801.000000	801.000000	801.000000	801.000000	801.0
mean	0.996255	1.057116	0.968789	1.073970	1.068976	1.065543	1.135456	1.192884	0.9
std	0.597248	0.438142	0.353058	0.654962	0.522167	0.717251	0.691853	0.604488	0.5
min	0.250000	0.250000	0.000000	0.000000	0.250000	0.000000	0.250000	0.250000	0.0
25%	0.500000	1.000000	1.000000	0.500000	1.000000	0.500000	0.500000	1.000000	1.0
50%	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.0
75%	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	2.000000	1.000000	1.0
max	4.000000	4.000000	2.000000	4.000000	4.000000	4.000000	4.000000	4.000000	4.0

8 rows × 34 columns

```
# Column management in dataframes.
print(pokedf.columns)
p2 = pokedf["is_legendary"]
print(type(p2))
```

```
Index(['abilities', 'against_bug', 'against_dark', 'against_dragon',
       'against_electric', 'against_fairy', 'against_fight', 'against_fire',
       'against_flying', 'against_ghost', 'against_grass', 'against_ground',
       'against_ice', 'against_normal', 'against_poison', 'against_psychic',
       'against_rock', 'against_steel', 'against_water', 'attack',
       'base_egg_steps', 'base_happiness', 'base_total', 'capture_rate',
       'classfication', 'defense', 'experience_growth', 'height_m', 'hp',
       'japanese_name', 'name', 'percentage_male', 'pokedex_number',
       'sp_attack', 'sp_defense', 'speed', 'type1', 'type2', 'weight_kg',
       'generation', 'is_legendary'],
      dtype='object')
<class 'pandas.core.series.Series'>
```

```
# Creating a new dataframe with specific columns only.
p2 = pokedf[['name', 'is_legendary']]
print(type(p2))
pokedf.iloc[0:4]
```

```
<class 'pandas.core.frame.DataFrame'>
```

	abilities	against_bug	against_dark	against_dragon	against_electric	against_fairy	against_fight	against_fire	against_flying
0	['Overgrow', 'Chlorophyll']	1.0	1.0	1.0	0.5	0.5	0.5	2.0	2.0
1	['Overgrow', 'Chlorophyll']	1.0	1.0	1.0	0.5	0.5	0.5	2.0	2.0
2	['Overgrow', 'Chlorophyll']	1.0	1.0	1.0	0.5	0.5	0.5	2.0	2.0
3	['Blaze', 'Solar Power']	0.5	1.0	1.0	1.0	0.5	1.0	0.5	1.0

4 rows × 41 columns

```
print(p2.index, p2.is_legendary)
```

```
RangeIndex(start=0, stop=801, step=1) 0      0
1      0
2      0
3      0
4      0
      ..
796    1
797    1
798    1
799    1
800    1
Name: is_legendary, Length: 801, dtype: int64
```

```
pokedf.iloc[2:10, 3:6]
```

	against_dragon	against_electric	against_fairy
2	1.0	0.5	0.5
3	1.0	1.0	0.5
4	1.0	1.0	0.5
5	1.0	2.0	0.5
6	1.0	2.0	1.0
7	1.0	2.0	1.0
8	1.0	2.0	1.0
9	1.0	1.0	1.0

```
# Print the specific rows where the value is either fire or water.
p5 = pokedf[pokedf.type1.isin(["fire", "water"])]
p5
```

	abilities	against_bug	against_dark	against_dragon	against_electric	against_fairy	against_fight	against_fire	against_flying
3	['Blaze', 'Solar Power']	0.50	1.0	1.0	1.0	0.5	1.0	0.50	1.0
4	['Blaze', 'Solar Power']	0.50	1.0	1.0	1.0	0.5	1.0	0.50	1.0
5	['Blaze', 'Solar Power']	0.25	1.0	1.0	2.0	0.5	0.5	0.50	1.0
6	['Torrent', 'Rain Dish']	1.00	1.0	1.0	2.0	1.0	1.0	0.50	1.0
7	['Torrent', 'Rain Dish']	1.00	1.0	1.0	2.0	1.0	1.0	0.50	1.0
...	...	...	...	...	...	...	...	...	...
751	['Water Bubble', 'Water Absorb']	1.00	1.0	1.0	2.0	1.0	0.5	1.00	2.0
770	['Innards Out', 'Unaware']	1.00	1.0	1.0	2.0	1.0	1.0	0.50	1.0
775	['Shell Armor']	0.50	1.0	2.0	0.5	1.0	1.0	0.25	1.0
778	['Dazzling', 'Strong Jaw', 'Wonder Skin ']	2.00	2.0	1.0	2.0	1.0	0.5	0.50	1.0
787	['Misty Surge', 'Telepathy']	0.50	0.5	0.0	2.0	1.0	0.5	0.50	1.0

166 rows × 41 columns

```
# Iterable method.
p6 = pd.DataFrame(columns = pokedf.columns)
print(p6)
```

```
Empty DataFrame
Columns: [abilities, against_bug, against_dark, against_dragon, against_electric, against_fairy, against_fight, against_fire, against_f
Index: []
```

```
[0 rows x 41 columns]
```

```
# Count the no. of entries in all categories.
pokedf.groupby(pokedf.type1).count()
```

type1	abilities	against_bug	against_dark	against_dragon	against_electric	a
bug	72	72	72	72	72	
dark	29	29	29	29	29	
dragon	27	27	27	27	27	
electric	39	39	39	39	39	
fairy	18	18	18	18	18	
fighting	28	28	28	28	28	
fire	52	52	52	52	52	
flying	3	3	3	3	3	
ghost	27	27	27	27	27	
grass	78	78	78	78	78	
ground	32	32	32	32	32	
ice	23	23	23	23	23	
normal	105	105	105	105	105	
poison	32	32	32	32	32	
psychic	53	53	53	53	53	
rock	45	45	45	45	45	
steel	24	24	24	24	24	
water	114	114	114	114	114	

18 rows × 40 columns

✓ 0s completed at 12:33 PM