PRIME INTUIT
Finishing School

Pandas

PRIME INTUIT
Finishing School

Pandas

Pandas is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool, built on top of the Python programming language.

https://pandas.pydata.org/

What we will learn:

- Pandas objects: Series and Dataframes
- Pandas for Data Wrangling
- Visualizing Data

PRIME INTUIT
Finishing School

Pandas – Series Objects

Pandas Series is a one-dimensional labeled array capable of holding data of any type (integer, string, float, python objects, etc.). The axis labels are collectively called index.
The object supports both integer and label-based indexing and provides a host of methods for performing operations involving the index.

```
Import numpy as np
Import Pandas as pd

s = pd.Series([0, 1, 1, 2, 3, 5, 8, 13])
print(s)
s.values
s.index
```

Covered

PRIME INTUIT
Finishing School

Pandas – Series Objects

```
# series is iteratable

for v in s.values:
        print(v)

for i in s.index:
        print(i)

# Zip them into tuples

for item in zip(s.index, s.values):
        print(item)
```

Looks like a dict where we had a key and a value
Here, we have a index and a value, there are similarities but they are different as well

Pandas – Series Objects

# access any number

```
S[0]
S[1]
S[5]

Mercury = pd.Series([0.33, 57.9, 4222.6], index = (['mass', 'diameter', 'daylength'])

Mercury['diameter']

Mercury['mass']
```

✓ right way

```
Mercury.mass
```

wrong way

not recommended as it has short comes

Pandas – Series Objects

```
# Create a series using array

arr = np.random.randint(0, 10, 10)

arr

rand_series = pd.Series(arr)

print(rand_series)

ind = np.arange(10, 20)
rand_series = pd.Series(arr, index = ind)

print(rand_series)
```

Covered

Pandas – Series Objects

```
# Create a series using dictionary

d = {}
d ['mass'] = 0.33
d ['diameter'] = 57.9
d ['daylength'] =4222.6

print(d)

Mercury = pd.Series(d)
print(Mercury)

Mercury = pd.Series(d, index['mass', 'diameter'])
Print(Mercury)
```

Correct

PRIME INTUIT
Finishing School

Pandas – Series Objects (iLoc & Loc)

# iLoc & Loc

0 1 2 3 4 5 6 7

```
s = pd.Series([0, 1, 1, 2, 3, 5, 8, 13], index = [1, 2, 3, 4, 5, 6, 7])

print(s)

s.loc[4]          # location based on custom index

s.iloc[4]         # implicit or std way starting with 0 index

s.iloc[0]

s.loc[0]
```

?

PRIME INTUIT
Finishing School

Pandas – Series Objects (iLoc & Loc)

```
# iLoc & Loc

Mercury = pd.Series(d, index= [‘mass’, ‘diameter’, ‘daylength’]

Mercury.loc[‘mass’]
Mercury.iloc[0]
Mercury.iloc[-1]

Mercury.iloc[0:2]

Mercury.loc[‘mass’ : ’diameter’]                # note change, end value included
```

PRIME INTUIT
Finishing School

Pandas – Series Objects (Simple Operations)

```
# Operations

Mass = pd.Series(0.33, 4.07, 5.97, 0.642, 1090, 568, 86.0, 102, 0.0146), index=
['mercury', 'venus', 'earth', 'mars', 'jupiter', 'saturn', 'uranus', 'neptune', 'pluto']

Print(mass)

Mass > 100

Mass[mass>100]
Mass[(mass>100) & (mass<600)]

Mass * 2
Mass /10
Np.mean(mass)
```

[illegible]

PRIME INTUIT
Finishing School

Pandas – Series Objects (Simple Operations)

```
# Operations

Np.amin(mass)

Np.amax(mass)

Np.median(5.97)

Mass * Mass                              # need to have same indices

Mass + mass
Bigmass = mass[mass > 100]
Bigmass
New_mass = mass + bigmass
Print(new_mass)                          # NaN when one of the series does not have the value
```

Covered

PRIME INTUIT
Finishing School

Pandas – Series Objects (Simple Operations)

# Operations

Pd.isnull(new_mass)

Newmass[~pd.isnull(new_mass)]

Mass[‘Eris’] = 0.000292
Mass[‘Moon’] = 0.7346

Mass

Mass.drop([‘Moon’])

diameter = pd.Series(4079, 12104, 12756, 3475, 6792, 142904, 120536, 51110, 49528, 2370), index= [‘mercury’, ‘venus’, ‘earth’, ‘mars’, ‘jupiter’, ‘saturn’, ‘uranus’, ‘neptune’, ‘pluto’]

PRIME INTUIT
Finishing School

Pandas – Series Objects (Exercise)

```
# Exercise, using mass and dia find density of each planet using series

diameter = pd.Series(4079, 12104, 12756, 3475, 6792, 142904, 120536, 51110,
49528, 2370), index= [‘mercury’, ‘venus’, ‘earth’, ‘mars’, ‘jupiter’, ‘saturn’, ‘uranus’,
‘neptune’, ‘pluto’]

Density                    = mass / volume       = mass / Pi D**3) /6

Density = pd.Series([])
Print(density)
For planet in mass.index:
        density[planet] = mass[planet] / (np.pi*diameter[planet] * diameter[planet]
*diameter[planet] / 6)

Print(density)
```

covered

Pandas – Series Objects (Exercise)

# Exercise, using mass and dia find density of each planet using series
Better way of resolving the same using series and numpy:

```
Density = mass / (np.pi * np.power(diameter, 3) / 6)

Density

Mass['Undiscovered'] = 6
Density = mass / (np.pi * np.power(diameter, 3) / 6)
density
```

Covered

PRIME INTUIT
Finishing School

Pandas – Series Objects (Exercise)

```
# Exercise, replace all values with NaN with mean density:

Density_mean = np.mean(density)

For key in density.index:
        if pd.isnull(density[key]):
                density[key] = density_mean
```

Better way of resolving the same using series and numpy:

```
Mass['Undiscovered'] = 6

Pd.isnull(density)
Density[pd.isnull(density)] = np.mean
density
```

Covered