*You are currently looking at **version 1.0** of this notebook. To download notebooks and datafiles, as well as get help on Jupyter notebooks in the Coursera platform, visit the Jupyter Notebook FAQ (https://www.coursera.org/learn/python-data-analysis/resources/0dhYG) course resource.*

The Series Data Structure

```
In [ ]:  import pandas as pd
         pd.Series?
```

```
In [ ]:  animals = ['Tiger', 'Bear', 'Moose']
         pd.Series(animals)
```

```
In [ ]:  numbers = [1, 2, 3]
         pd.Series(numbers)
```

```
In [ ]:  animals = ['Tiger', 'Bear', None]
         pd.Series(animals)
```

```
In [ ]:  numbers = [1, 2, None]
         pd.Series(numbers)
```

```
In [ ]:  import numpy as np
         np.nan == None
```

```
In [ ]:  np.nan == np.nan
```

```
In [ ]:  np.isnan(np.nan)
```

```
In [ ]:  sports = {'Archery': 'Bhutan',
                    'Golf': 'Scotland',
                    'Sumo': 'Japan',
                    'Taekwondo': 'South Korea'}
         s = pd.Series(sports)
         s
```

```
In [ ]:  s.index
```

```
In [ ]:  s = pd.Series(['Tiger', 'Bear', 'Moose'], index=['India', 'America', 'Canada'
         ])
         s
```

```
In [ ]: sports = {'Archery': 'Bhutan',
               'Golf': 'Scotland',
               'Sumo': 'Japan',
               'Taekwondo': 'South Korea'}
        s = pd.Series(sports, index=['Golf', 'Sumo', 'Hockey'])
        s
```

Querying a Series

```
In [ ]: sports = {'Archery': 'Bhutan',
               'Golf': 'Scotland',
               'Sumo': 'Japan',
               'Taekwondo': 'South Korea'}
        s = pd.Series(sports)
        s
```

```
In [ ]: s.iloc[3]
```

```
In [ ]: s.loc['Golf']
```

```
In [ ]: s[3]
```

```
In [ ]: s['Golf']
```

```
In [ ]: sports = {99: 'Bhutan',
               100: 'Scotland',
               101: 'Japan',
               102: 'South Korea'}
        s = pd.Series(sports)
```

```
In [ ]: s[0] #This won't call s.iloc[0] as one might expect, it generates an error ins
        tead
```

```
In [ ]: s = pd.Series([100.00, 120.00, 101.00, 3.00])
        s
```

```
In [ ]: total = 0
        for item in s:
            total+=item
        print(total)
```

```
In [ ]: import numpy as np

        total = np.sum(s)
        print(total)
```

```
In [ ]:  #this creates a big series of random numbers
         s = pd.Series(np.random.randint(0,1000,10000))
         s.head()
```

```
In [ ]:  len(s)
```

```
In [ ]:  %%timeit -n 100
         summary = 0
         for item in s:
             summary+=item
```

```
In [ ]:  %%timeit -n 100
         summary = np.sum(s)
```

```
In [ ]:  s+=2 #adds two to each item in s using broadcasting
         s.head()
```

```
In [ ]:  for label, value in s.iteritems():
             s.set_value(label, value+2)
         s.head()
```

```
In [ ]:  %%timeit -n 10
         s = pd.Series(np.random.randint(0,1000,10000))
         for label, value in s.iteritems():
             s.loc[label]= value+2
```

```
In [ ]:  %%timeit -n 10
         s = pd.Series(np.random.randint(0,1000,10000))
         s+=2
```

```
In [ ]:  s = pd.Series([1, 2, 3])
         s.loc['Animal'] = 'Bears'
         s
```

```
In [ ]:  original_sports = pd.Series({'Archery': 'Bhutan',
                                       'Golf': 'Scotland',
                                       'Sumo': 'Japan',
                                       'Taekwondo': 'South Korea'})
         cricket_loving_countries = pd.Series(['Australia',
                                               'Barbados',
                                               'Pakistan',
                                               'England'],
                                             index=['Cricket',
                                                    'Cricket',
                                                    'Cricket',
                                                    'Cricket'])
         all_countries = original_sports.append(cricket_loving_countries)
```

```
In [ ]:  original_sports
```

```
In [ ]: cricket_loving_countries
```

```
In [ ]: all_countries
```

```
In [ ]: all_countries.loc['Cricket']
```

The DataFrame Data Structure

```
In [ ]: import pandas as pd
        purchase_1 = pd.Series({'Name': 'Chris',
                                'Item Purchased': 'Dog Food',
                                'Cost': 22.50})
        purchase_2 = pd.Series({'Name': 'Kevyn',
                                'Item Purchased': 'Kitty Litter',
                                'Cost': 2.50})
        purchase_3 = pd.Series({'Name': 'Vinod',
                                'Item Purchased': 'Bird Seed',
                                'Cost': 5.00})
        df = pd.DataFrame([purchase_1, purchase_2, purchase_3], index=['Store 1', 'Sto
        re 1', 'Store 2'])
        df.head()
```

```
In [ ]: df.loc['Store 2']
```

```
In [ ]: type(df.loc['Store 2'])
```

```
In [ ]: df.loc['Store 1']
```

```
In [ ]: df.loc['Store 1', 'Cost']
```

```
In [ ]: df.T
```

```
In [ ]: df.T.loc['Cost']
```

```
In [ ]: df['Cost']
```

```
In [ ]: df.loc['Store 1']['Cost']
```

```
In [ ]: df.loc[:,['Name', 'Cost']]
```

```
In [ ]: df.drop('Store 1')
```

```
In [ ]: df
```

```
In [ ]:  copy_df = df.copy()
         copy_df = copy_df.drop('Store 1')
         copy_df
```

```
In [ ]:  copy_df.drop?
```

```
In [ ]:  del copy_df['Name']
         copy_df
```

```
In [ ]:  df['Location'] = None
         df
```

Dataframe Indexing and Loading

```
In [ ]:  costs = df['Cost']
         costs
```

```
In [ ]:  costs+=2
         costs
```

```
In [ ]:  df
```

```
In [ ]:  !cat olympics.csv
```

```
In [ ]:  df = pd.read_csv('olympics.csv')
         df.head()
```

```
In [ ]:  df = pd.read_csv('olympics.csv', index_col = 0, skiprows=1)
         df.head()
```

```
In [ ]:  df.columns
```

```
In [ ]:  for col in df.columns:
             if col[:2]=='01':
                 df.rename(columns={col:'Gold' + col[4:]}, inplace=True)
             if col[:2]=='02':
                 df.rename(columns={col:'Silver' + col[4:]}, inplace=True)
             if col[:2]=='03':
                 df.rename(columns={col:'Bronze' + col[4:]}, inplace=True)
             if col[:1]=='№':
                 df.rename(columns={col:'#' + col[1:]}, inplace=True)

         df.head()
```

Querying a DataFrame

```
In [ ]:  df['Gold'] > 0
```

```
In [ ]:  only_gold = df.where(df['Gold'] > 0)
         only_gold.head()
```

```
In [ ]:  only_gold['Gold'].count()
```

```
In [ ]:  df['Gold'].count()
```

```
In [ ]:  only_gold = only_gold.dropna()
         only_gold.head()
```

```
In [ ]:  only_gold = df[df['Gold'] > 0]
         only_gold.head()
```

```
In [ ]:  len(df[(df['Gold'] > 0) | (df['Gold.1'] > 0)])
```

```
In [ ]:  df[(df['Gold.1'] > 0) & (df['Gold'] == 0)]
```

Indexing Dataframes

```
In [ ]:  df.head()
```

```
In [ ]:  df['country'] = df.index
         df = df.set_index('Gold')
         df.head()
```

```
In [ ]:  df = df.reset_index()
         df.head()
```

```
In [ ]:  df = pd.read_csv('census.csv')
         df.head()
```

```
In [ ]:  df['SUMLEV'].unique()
```

```
In [ ]:  df=df[df['SUMLEV'] == 50]
         df.head()
```

```
In [ ]:  columns_to_keep = ['STNAME',
                            'CTYNAME',
                            'BIRTHS2010',
                            'BIRTHS2011',
                            'BIRTHS2012',
                            'BIRTHS2013',
                            'BIRTHS2014',
                            'BIRTHS2015',
                            'POPESTIMATE2010',
                            'POPESTIMATE2011',
                            'POPESTIMATE2012',
                            'POPESTIMATE2013',
                            'POPESTIMATE2014',
                            'POPESTIMATE2015']
         df = df[columns_to_keep]
         df.head()
```

```
In [ ]:  df = df.set_index(['STNAME', 'CTYNAME'])
         df.head()
```

```
In [ ]:  df.loc['Michigan', 'Washtenaw County']
```

```
In [ ]:  df.loc[ [('Michigan', 'Washtenaw County'),
                  ('Michigan', 'Wayne County')] ]
```

Missing values

```
In [ ]:  df = pd.read_csv('log.csv')
         df
```

```
In [ ]:  df.fillna?
```

```
In [ ]:  df = df.set_index('time')
         df = df.sort_index()
         df
```

```
In [ ]:  df = df.reset_index()
         df = df.set_index(['time', 'user'])
         df
```

```
In [ ]:  df = df.fillna(method='ffill')
         df.head()
```