Colab Link:
https://colab.research.google.com/drive/1_MAJ4RMmeow8ldscbCY47MxOODlnskES?usp=sharing

- **Installation of pandas**
  - Importing pandas
  - Importing the dataset
  - Dataframe/Series
- **Basic ops on a DataFrame**
  - df.info()
  - df.head()
  - df.tail()
  - df.shape()
  - df.describe()
- **Basic ops on columns**
  - Different ways of accessing cols
  - Check for Unique values
  - Rename column
  - Deleting col
  - Creating new cols
  - Quiz1 added
- **Basic ops on rows**
  - Implicit/explicit index
  - df.index[]
  - Indexing in series
  - Slicing in series
  - loc/iloc
  - Indexing/Slicing in dataframe
  - Adding a row
  - Check for duplicates
  - Deleting a row
- **Working with both rows and cols**
  - Quiz2 added
- **More in-built ops in pandas**
  - sum()
  - count()
  - mean()
- **Sorting**
  - Quiz3 added
- **Creating series and Dataframes from scratch**

```
!pip install pandas
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-w
```

```
Requirement already satisfied: pandas in /usr/local/lib/python3.7/dist-package
Requirement already satisfied: python-dateutil>=2.7.3 in /usr/local/lib/python
Requirement already satisfied: numpy>=1.17.3 in /usr/local/lib/python3.7/dist-
Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.7/dist-p
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-packa
```

```
import pandas as pd
import numpy as np
```

```
!wget "https://drive.google.com/uc?export=download&id=1E3bwvYGf1ig32RmcYiWc0I
```

```
--2022-06-28 15:41:33--  https://drive.google.com/uc?export=download&id=1E3bwv
Resolving drive.google.com (drive.google.com)... 142.251.107.102, 142.251.107.
Connecting to drive.google.com (drive.google.com)|142.251.107.102|:443... conn
HTTP request sent, awaiting response... 303 See Other
Location: https://doc-0s-68-docs.googleusercontent.com/docs/securesc/ha0ro937c
Warning: wildcards not supported in HTTP.
--2022-06-28 15:41:34--  https://doc-0s-68-docs.googleusercontent.com/docs/sec
Resolving doc-0s-68-docs.googleusercontent.com (doc-0s-68-docs.googleuserconte
Connecting to doc-0s-68-docs.googleusercontent.com (doc-0s-68-docs.googleuserc
HTTP request sent, awaiting response... 200 OK
Length: 83785 (82K) [text/csv]
Saving to: 'mckinsey.csv'

mckinsey.csv        100%[===================>]  81.82K  --.-KB/s    in 0.001s

2022-06-28 15:41:34 (88.4 MB/s) - 'mckinsey.csv' saved [83785/83785]
```

```
df = pd.read_csv("mckinsey.csv")
df
```

	country	year	population	continent	life_exp	gdp_cap
0	Afghanistan	1952	8425333	Asia	28.801	779.445314
1	Afghanistan	1957	9240934	Asia	30.332	820.853030
2	Afghanistan	1962	10267083	Asia	31.997	853.100710
3	Afghanistan	1967	11537966	Asia	34.020	836.197138
4	Afghanistan	1972	13079460	Asia	36.088	739.981106
...	...	...	...	...	...	...
1699	Zimbabwe	1987	9216418	Africa	62.351	706.157306
1700	Zimbabwe	1992	10704340	Africa	60.377	693.420786
1701	Zimbabwe	1997	11404948	Africa	46.809	792.449960
1702	Zimbabwe	2002	11926563	Africa	39.989	672.038623
1703	Zimbabwe	2007	12311143	Africa	43.487	469.709298

1704 rows × 6 columns

```
type(df)
```

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

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1704 entries, 0 to 1703
Data columns (total 6 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   country     1704 non-null   object 
 1   year        1704 non-null   int64  
 2   population  1704 non-null   int64  
 3   continent   1704 non-null   object 
 4   life_exp    1704 non-null   float64
 5   gdp_cap     1704 non-null   float64
dtypes: float64(2), int64(2), object(2)
memory usage: 80.0+ KB
```

```
df.shape
```

```
(1704, 6)
```

```
df.head(15)
```

	country	year	population	continent	life_exp	gdp_cap
0	Afghanistan	1952	8425333	Asia	28.801	779.445314
1	Afghanistan	1957	9240934	Asia	30.332	820.853030
2	Afghanistan	1962	10267083	Asia	31.997	853.100710
3	Afghanistan	1967	11537966	Asia	34.020	836.197138
4	Afghanistan	1972	13079460	Asia	36.088	739.981106
5	Afghanistan	1977	14880372	Asia	38.438	786.113360
6	Afghanistan	1982	12881816	Asia	39.854	978.011439
7	Afghanistan	1987	13867957	Asia	40.822	852.395945
8	Afghanistan	1992	16317921	Asia	41.674	649.341395
9	Afghanistan	1997	22227415	Asia	41.763	635.341351
10	Afghanistan	2002	25268405	Asia	42.129	726.734055
11	Afghanistan	2007	31889923	Asia	43.828	974.580338
12	Albania	1952	1282697	Europe	55.230	1601.056136
13	Albania	1957	1476505	Europe	59.280	1942.284244
14	Albania	1962	1728137	Europe	64.820	2312.888958

```
df.tail()
```

	country	year	population	continent	life_exp	gdp_cap
1699	Zimbabwe	1987	9216418	Africa	62.351	706.157306
1700	Zimbabwe	1992	10704340	Africa	60.377	693.420786
1701	Zimbabwe	1997	11404948	Africa	46.809	792.449960
1702	Zimbabwe	2002	11926563	Africa	39.989	672.038623
1703	Zimbabwe	2007	12311143	Africa	43.487	469.709298

```
df.describe()
```

	year	population	life_exp	gdp_cap
count	1704.00000	1.704000e+03	1704.000000	1704.000000
mean	1979.50000	2.960121e+07	59.474439	7215.327081
std	17.26533	1.061579e+08	12.917107	9857.454543
min	1952.00000	6.001100e+04	23.599000	241.165876
25%	1965.75000	2.793664e+06	48.198000	1202.060309
50%	1979.50000	7.023596e+06	60.712500	3531.846988
75%	1993.25000	1.958522e+07	70.845500	9325.462346
max	2007.00000	1.318683e+09	82.603000	113523.132900

```
df.describe(include="object")
```

	country	continent
count	1704	1704
unique	142	5
top	Afghanistan	Africa
freq	12	624

```
# Basic operations on columns
```

```
df.columns
```

```
Index(['country', 'year', 'population', 'continent', 'life_exp', 'gdp_cap'],
dtype='object')
```

```
df.keys()
```

```
Index(['country', 'year', 'population', 'continent', 'life_exp', 'gdp_cap'],
dtype='object')
```

```
df["country"]
```

```
0       Afghanistan
1       Afghanistan
2       Afghanistan
3       Afghanistan
4       Afghanistan
           ...
1699       Zimbabwe
1700       Zimbabwe
1701       Zimbabwe
1702       Zimbabwe
1703       Zimbabwe
Name: country, Length: 1704, dtype: object
```

```
type(df["country"])
```

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

```
df[["country", "continent"]].head()
```

	country	continent
0	Afghanistan	Asia
1	Afghanistan	Asia
2	Afghanistan	Asia
3	Afghanistan	Asia
4	Afghanistan	Asia

```
df["country"].unique()
```

```
array(['Afghanistan', 'Albania', 'Algeria', 'Angola', 'Argentina',
       'Australia', 'Austria', 'Bahrain', 'Bangladesh', 'Belgium',
       'Benin', 'Bolivia', 'Bosnia and Herzegovina', 'Botswana', 'Brazil',
       'Bulgaria', 'Burkina Faso', 'Burundi', 'Cambodia', 'Cameroon',
       'Canada', 'Central African Republic', 'Chad', 'Chile', 'China',
       'Colombia', 'Comoros', 'Congo, Dem. Rep.', 'Congo, Rep.',
       'Costa Rica', "Cote d'Ivoire", 'Croatia', 'Cuba', 'Czech Republic',
       'Denmark', 'Djibouti', 'Dominican Republic', 'Ecuador', 'Egypt',
       'El Salvador', 'Equatorial Guinea', 'Eritrea', 'Ethiopia',
       'Finland', 'France', 'Gabon', 'Gambia', 'Germany', 'Ghana',
       'Greece', 'Guatemala', 'Guinea', 'Guinea-Bissau', 'Haiti',
       'Honduras', 'Hong Kong, China', 'Hungary', 'Iceland', 'India',
       'Indonesia', 'Iran', 'Iraq', 'Ireland', 'Israel', 'Italy',
       'Jamaica', 'Japan', 'Jordan', 'Kenya', 'Korea, Dem. Rep.',
       'Korea, Rep.', 'Kuwait', 'Lebanon', 'Lesotho', 'Liberia', 'Libya',
       'Madagascar', 'Malawi', 'Malaysia', 'Mali', 'Mauritania',
       'Mauritius', 'Mexico', 'Mongolia', 'Montenegro', 'Morocco',
       'Mozambique', 'Myanmar', 'Namibia', 'Nepal', 'Netherlands',
```

```
       'New Zealand', 'Nicaragua', 'Niger', 'Nigeria', 'Norway', 'Oman',
       'Pakistan', 'Panama', 'Paraguay', 'Peru', 'Philippines', 'Poland',
       'Portugal', 'Puerto Rico', 'Reunion', 'Romania', 'Rwanda',
       'Sao Tome and Principe', 'Saudi Arabia', 'Senegal', 'Serbia',
       'Sierra Leone', 'Singapore', 'Slovak Republic', 'Slovenia',
       'Somalia', 'South Africa', 'Spain', 'Sri Lanka', 'Sudan',
       'Swaziland', 'Sweden', 'Switzerland', 'Syria', 'Taiwan',
       'Tanzania', 'Thailand', 'Togo', 'Trinidad and Tobago', 'Tunisia',
       'Turkey', 'Uganda', 'United Kingdom', 'United States', 'Uruguay',
       'Venezuela', 'Vietnam', 'West Bank and Gaza', 'Yemen, Rep.',
       'Zambia', 'Zimbabwe'], dtype=object)
```

```
len(df["country"].unique())
```

```
142
```

```
df['country'].value_counts()
```

```
Afghanistan          12
Pakistan             12
New Zealand          12
Nicaragua            12
Niger                12
                     ..
Eritrea              12
Equatorial Guinea    12
El Salvador          12
Egypt                12
Zimbabwe             12
Name: country, Length: 142, dtype: int64
```

```
df["country"].nunique()
```

```
142
```

```
df.rename({"country":"Country"}, axis=1, inplace=True)
```

```
df
```

	Country	year	population	continent	life_exp	gdp_cap
0	Afghanistan	1952	8425333	Asia	28.801	779.445314
1	Afghanistan	1957	9240934	Asia	30.332	820.853030
2	Afghanistan	1962	10267083	Asia	31.997	853.100710
3	Afghanistan	1967	11537966	Asia	34.020	836.197138

```
df.rename({"Country":"country"}, axis=1, inplace=True)
```

```
df.country # dont use it
```

```
0       Afghanistan
1       Afghanistan
2       Afghanistan
3       Afghanistan
4       Afghanistan
           ...
1699       Zimbabwe
1700       Zimbabwe
1701       Zimbabwe
1702       Zimbabwe
1703       Zimbabwe
Name: country, Length: 1704, dtype: object
```

```
df.drop("continent", axis=1, inplace=True)
```

```
df
```

	country	year	population	life_exp	gdp_cap
0	Afghanistan	1952	8425333	28.801	779.445314
1	Afghanistan	1957	9240934	30.332	820.853030
2	Afghanistan	1962	10267083	31.997	853.100710
3	Afghanistan	1967	11537966	34.020	836.197138
4	Afghanistan	1972	13079460	36.088	739.981106
...	...	...	...	...	...
1699	Zimbabwe	1987	9216418	62.351	706.157306
1700	Zimbabwe	1992	10704340	60.377	693.420786
1701	Zimbabwe	1997	11404948	46.809	792.449960
1702	Zimbabwe	2002	11926563	39.989	672.038623
1703	Zimbabwe	2007	12311143	43.487	469.709298

1704 rows × 5 columns

```
df["New"] = df["life_exp"] + df["year"]
```

```
df
```

	country	year	population	life_exp	gdp_cap	New
0	Afghanistan	1952	8425333	28.801	779.445314	1980.801
1	Afghanistan	1957	9240934	30.332	820.853030	1987.332
2	Afghanistan	1962	10267083	31.997	853.100710	1993.997
3	Afghanistan	1967	11537966	34.020	836.197138	2001.020
4	Afghanistan	1972	13079460	36.088	739.981106	2008.088
...	...	...	...	...	...	...
1699	Zimbabwe	1987	9216418	62.351	706.157306	2049.351
1700	Zimbabwe	1992	10704340	60.377	693.420786	2052.377
1701	Zimbabwe	1997	11404948	46.809	792.449960	2043.809
1702	Zimbabwe	2002	11926563	39.989	672.038623	2041.989
1703	Zimbabwe	2007	12311143	43.487	469.709298	2050.487

1704 rows × 6 columns

```
df['Own'] = [i for i in range(df.shape[0])]
```

```
df
```

	country	year	population	life_exp	gdp_cap	New	Own
0	Afghanistan	1952	8425333	28.801	779.445314	1980.801	0
1	Afghanistan	1957	9240934	30.332	820.853030	1987.332	1
2	Afghanistan	1962	10267083	31.997	853.100710	1993.997	2
3	Afghanistan	1967	11537966	34.020	836.197138	2001.020	3
4	Afghanistan	1972	13079460	36.088	739.981106	2008.088	4
...	...	...	...	...	...	...	...
1699	Zimbabwe	1987	9216418	62.351	706.157306	2049.351	1699
1700	Zimbabwe	1992	10704340	60.377	693.420786	2052.377	1700
1701	Zimbabwe	1997	11404948	46.809	792.449960	2043.809	1701
1702	Zimbabwe	2002	11926563	39.989	672.038623	2041.989	1702
1703	Zimbabwe	2007	12311143	43.487	469.709298	2050.487	1703

1704 rows × 7 columns

```
df.drop(["New", "Own"], axis=1, inplace=True)
```

```
df
```

	country	year	population	life_exp	gdp_cap
0	Afghanistan	1952	8425333	28.801	779.445314
1	Afghanistan	1957	9240934	30.332	820.853030
2	Afghanistan	1962	10267083	31.997	853.100710
3	Afghanistan	1967	11537966	34.020	836.197138
4	Afghanistan	1972	13079460	36.088	739.981106
...	...	...	...	...	...
1699	Zimbabwe	1987	9216418	62.351	706.157306
1700	Zimbabwe	1992	10704340	60.377	693.420786
1701	Zimbabwe	1997	11404948	46.809	792.449960
1702	Zimbabwe	2002	11926563	39.989	672.038623
1703	Zimbabwe	2007	12311143	43.487	469.709298

1704 rows × 5 columns

```
# Working with Rows
```

```
ser = df["country"]
```

```
ser[4]
```

```
'Afghanistan'
```

```
ser[6:15]
```

```
6     Afghanistan
7     Afghanistan
8     Afghanistan
9     Afghanistan
10    Afghanistan
11    Afghanistan
12        Albania
13        Albania
14        Albania
Name: country, dtype: object
```

```
ser
```

```
0       Afghanistan
1       Afghanistan
2       Afghanistan
3       Afghanistan
4       Afghanistan
```

```
            ...
1699       Zimbabwe
1700       Zimbabwe
1701       Zimbabwe
1702       Zimbabwe
1703       Zimbabwe
Name: country, Length: 1704, dtype: object
```

```
ser.index = np.arange(1, df.shape[0]+1, step=1)
```

```
ser
```

```
1       Afghanistan
2       Afghanistan
3       Afghanistan
4       Afghanistan
5       Afghanistan
           ...
1700       Zimbabwe
1701       Zimbabwe
1702       Zimbabwe
1703       Zimbabwe
1704       Zimbabwe
Name: country, Length: 1704, dtype: object
```

```
# Indexes/Labels
# Explicit Indexes - visble outside, can be anything - numbers, 0-N-1, 1-N, st
# Implicit Indexes - 0-N-1
```

```
ser
```

```
1       Afghanistan
2       Afghanistan
3       Afghanistan
4       Afghanistan
5       Afghanistan
           ...
1700       Zimbabwe
1701       Zimbabwe
1702       Zimbabwe
1703       Zimbabwe
1704       Zimbabwe
Name: country, Length: 1704, dtype: object
```

```
ser.index[2]
```

```
3
```

```
data = pd.Series(["a", "b", "c"], index=[1, 5, 3])
data
```

```
1    a
5    b
```

```
3    c
dtype: object
```

```
data[1] # indexing uses explicit indices
```

```
'a'
```

```
data[1:3] # slicing used implicit indices
```

```
5    b
3    c
dtype: object
```

```
data = pd.Series(["a", "b", "c"], index=["x", "y", "z"])
data
```

```
x    a
y    b
z    c
dtype: object
```

```
data = pd.Series(["a", "b", "c"], index=[1, 2, 2])
data
```

```
1    a
2    b
2    c
dtype: object
```

```
data[2]
```

```
2    b
2    c
dtype: object
```

```
# indexing uses explicit indices
# slicing uses implicit indices
```

```
# Indexers - loc and iloc
```

```
# loc - Allows indexing and slicing that always references the explicit index
```

```
data = pd.Series(["a", "b", "c"], index=[1, 2, 2])
data
```

```
1    a
2    b
2    c
dtype: object
```

```
data.loc[2]
```

```
2    b
2    c
dtype: object
```

```
data.loc[1]
```

```
'a'
```

```
data.loc[1:3]
```

```
1    a
2    b
2    c
dtype: object
```

```
data.loc[1:2]
```

```
1    a
2    b
2    c
dtype: object
```

```
# iloc - Allows indexing and slicing that always references the implicit index
```

```
data.iloc[1]
```

```
'b'
```

```
data.iloc[0]
```

```
'a'
```

```
data.loc[0] # indexing it gives an error
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
/usr/local/lib/python3.7/dist-packages/pandas/core/indexes/base.py in
get_loc(self, key, method, tolerance)
   3360             try:
-> 3361                 return self._engine.get_loc(casted_key)
   3362             except KeyError as err:
```

8 frames

```
KeyError: 0
```

```
data.iloc[0:2] # end is not inclusive just like in Python normal indexing
```

```
1    a
2    b
dtype: object
```

```
   3361                 return self._engine.get_loc(casted_key)
```

```
data.loc[100:105] # slicing it doesnt it give error
```

```
Series([], dtype: object)
```

```
df
```

	country	year	population	life_exp	gdp_cap
0	Afghanistan	1952	8425333	28.801	779.445314
1	Afghanistan	1957	9240934	30.332	820.853030
2	Afghanistan	1962	10267083	31.997	853.100710
3	Afghanistan	1967	11537966	34.020	836.197138
4	Afghanistan	1972	13079460	36.088	739.981106
...	...	...	...	...	...
1699	Zimbabwe	1987	9216418	62.351	706.157306
1700	Zimbabwe	1992	10704340	60.377	693.420786
1701	Zimbabwe	1997	11404948	46.809	792.449960
1702	Zimbabwe	2002	11926563	39.989	672.038623
1703	Zimbabwe	2007	12311143	43.487	469.709298

1704 rows × 5 columns

```
df.loc[3]
```

```
country       Afghanistan
year                 1967
population       11537966
life_exp            34.02
gdp_cap         836.197138
Name: 3, dtype: object
```

```
df.iloc[3]
```

```
country       Afghanistan
year                 1967
population       11537966
life_exp            34.02
gdp_cap         836.197138
Name: 3, dtype: object
```

```
df.index = df.index + 1
```

```
df
```

	country	year	population	life_exp	gdp_cap
1	Afghanistan	1952	8425333	28.801	779.445314
2	Afghanistan	1957	9240934	30.332	820.853030
3	Afghanistan	1962	10267083	31.997	853.100710
4	Afghanistan	1967	11537966	34.020	836.197138
5	Afghanistan	1972	13079460	36.088	739.981106
...	...	...	...	...	...
1700	Zimbabwe	1987	9216418	62.351	706.157306
1701	Zimbabwe	1992	10704340	60.377	693.420786
1702	Zimbabwe	1997	11404948	46.809	792.449960
1703	Zimbabwe	2002	11926563	39.989	672.038623
1704	Zimbabwe	2007	12311143	43.487	469.709298

1704 rows × 5 columns

```
df.iloc[[1, 10, 100]]
```

	country	year	population	life_exp	gdp_cap
2	Afghanistan	1957	9240934	30.332	820.853030
11	Afghanistan	2002	25268405	42.129	726.734055
101	Bangladesh	1972	70759295	45.252	630.233627

```
df.loc[[1, 10, 100]]
```

	country	year	population	life_exp	gdp_cap

```
df.iloc[-1]
```

```
country         Zimbabwe
year                2007
population      12311143
life_exp           43.487
gdp_cap        469.709298
Name: 1704, dtype: object
```

```
df.loc[-1]
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
/usr/local/lib/python3.7/dist-packages/pandas/core/indexes/range.py in
get_loc(self, key, method, tolerance)
    384                 try:
--> 385                     return self._range.index(new_key)
    386                 except ValueError as err:

ValueError: -1 is not in range

The above exception was the direct cause of the following exception:

KeyError                                  Traceback (most recent call last)
```

5 frames

```
/usr/local/lib/python3.7/dist-packages/pandas/core/indexes/range.py in
get_loc(self, key, method, tolerance)
    385                     return self._range.index(new_key)
    386                 except ValueError as err:
--> 387                     raise KeyError(key) from err
    388             raise KeyError(key)
    389         return super().get_loc(key, method=method,
tolerance=tolerance)

KeyError: -1
```

```
df.index = df["country"]
```

```
df
```

	country	year	population	life_exp	gdp_cap
country					
Afghanistan	Afghanistan	1952	8425333	28.801	779.445314
Afghanistan	Afghanistan	1957	9240934	30.332	820.853030
Afghanistan	Afghanistan	1962	10267083	31.997	853.100710
Afghanistan	Afghanistan	1967	11537966	34.020	836.197138
Afghanistan	Afghanistan	1972	13079460	36.088	739.981106

```
df = pd.read_csv("mckinsey.csv")
df
```

	country	year	population	continent	life_exp	gdp_cap
0	Afghanistan	1952	8425333	Asia	28.801	779.445314
1	Afghanistan	1957	9240934	Asia	30.332	820.853030
2	Afghanistan	1962	10267083	Asia	31.997	853.100710
3	Afghanistan	1967	11537966	Asia	34.020	836.197138
4	Afghanistan	1972	13079460	Asia	36.088	739.981106
...	...	...	...	...	...	...
1699	Zimbabwe	1987	9216418	Africa	62.351	706.157306
1700	Zimbabwe	1992	10704340	Africa	60.377	693.420786
1701	Zimbabwe	1997	11404948	Africa	46.809	792.449960
1702	Zimbabwe	2002	11926563	Africa	39.989	672.038623
1703	Zimbabwe	2007	12311143	Africa	43.487	469.709298

1704 rows × 6 columns

```
df_temp = df.set_index("country")
df_temp
```

country	year	population	continent	life_exp	gdp_cap
Afghanistan	1952	8425333	Asia	28.801	779.445314
Afghanistan	1957	9240934	Asia	30.332	820.853030
Afghanistan	1962	10267083	Asia	31.997	853.100710
Afghanistan	1967	11537966	Asia	34.020	836.197138

```
df_temp.loc["Afghanistan"]
```

country	year	population	continent	life_exp	gdp_cap
Afghanistan	1952	8425333	Asia	28.801	779.445314
Afghanistan	1957	9240934	Asia	30.332	820.853030
Afghanistan	1962	10267083	Asia	31.997	853.100710
Afghanistan	1967	11537966	Asia	34.020	836.197138
Afghanistan	1972	13079460	Asia	36.088	739.981106
Afghanistan	1977	14880372	Asia	38.438	786.113360
Afghanistan	1982	12881816	Asia	39.854	978.011439
Afghanistan	1987	13867957	Asia	40.822	852.395945
Afghanistan	1992	16317921	Asia	41.674	649.341395
Afghanistan	1997	22227415	Asia	41.763	635.341351
Afghanistan	2002	25268405	Asia	42.129	726.734055
Afghanistan	2007	31889923	Asia	43.828	974.580338

```
df
```

	country	year	population	continent	life_exp	gdp_cap
0	Afghanistan	1952	8425333	Asia	28.801	779.445314
1	Afghanistan	1957	9240934	Asia	30.332	820.853030
2	Afghanistan	1962	10267083	Asia	31.997	853.100710

```
data_dict = {'country': 'India', 'year': 2000,'life_exp':37.08,'population':1
```

```
df.append(data_dict, ignore_index=True)
```

	country	year	population	continent	life_exp	gdp_cap
0	Afghanistan	1952	8425333	Asia	28.801	779.445314
1	Afghanistan	1957	9240934	Asia	30.332	820.853030
2	Afghanistan	1962	10267083	Asia	31.997	853.100710
3	Afghanistan	1967	11537966	Asia	34.020	836.197138
4	Afghanistan	1972	13079460	Asia	36.088	739.981106
...	...	...	...	...	...	...
1700	Zimbabwe	1992	10704340	Africa	60.377	693.420786
1701	Zimbabwe	1997	11404948	Africa	46.809	792.449960
1702	Zimbabwe	2002	11926563	Africa	39.989	672.038623
1703	Zimbabwe	2007	12311143	Africa	43.487	469.709298
1704	India	2000	13500000	NaN	37.080	900.230000

1705 rows × 6 columns

```
df.loc[len(df.index)] = ["India", 2000, 1350000, "Asia", 37.080, 900.230000]
```

```
df
```

	country	year	population	continent	life_exp	gdp_cap
0	Afghanistan	1952	8425333	Asia	28.801	779.445314
1	Afghanistan	1957	9240934	Asia	30.332	820.853030
2	Afghanistan	1962	10267083	Asia	31.997	853.100710

```
df.loc[len(df.index)] = {'country': 'India', 'year': 2000,'life_exp':37.08,'pc
```

```
df
```

	country	year	population	continent	life_exp	gdp_cap
0	Afghanistan	1952	8425333	Asia	28.801	779.445314
1	Afghanistan	1957	9240934	Asia	30.332	820.853030
2	Afghanistan	1962	10267083	Asia	31.997	853.100710
3	Afghanistan	1967	11537966	Asia	34.020	836.197138
4	Afghanistan	1972	13079460	Asia	36.088	739.981106
...	...	...	...	...	...	...
1701	Zimbabwe	1997	11404948	Africa	46.809	792.449960
1702	Zimbabwe	2002	11926563	Africa	39.989	672.038623
1703	Zimbabwe	2007	12311143	Africa	43.487	469.709298
1704	India	2000	1350000	Asia	37.080	900.230000
1706	India	2000	13500000	NaN	37.080	900.230000

1706 rows × 6 columns

```
df.iloc[len(df.index)] = {'country': 'India', 'year': 2000,'life_exp':37.08,'
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
<ipython-input-115-6ba8a45c0f22> in <module>()
----> 1 df.iloc[len(df.index)] = {'country': 'India', 'year':
2000,'life_exp':37.08,'population':13500000,'gdp_cap':900.23}

                                 1 frames
/usr/local/lib/python3.7/dist-packages/pandas/core/indexing.py in
_has_valid_setitem_indexer(self, indexer)
   1459             elif is_integer(i):
   1460                 if i >= len(ax):
-> 1461                     raise IndexError("iloc cannot enlarge its target
object")
   1462             elif isinstance(i, dict):
   1463                 raise IndexError("iloc cannot enlarge its target
object")

IndexError: iloc cannot enlarge its target object
```

```
df.loc[1705] = {'country': 'India', 'year': 2000,'life_exp':37.08,'population
```

```
df
```

	country	year	population	continent	life_exp	gdp_cap
0	Afghanistan	1952	8425333	Asia	28.801	779.445314
1	Afghanistan	1957	9240934	Asia	30.332	820.853030
2	Afghanistan	1962	10267083	Asia	31.997	853.100710
3	Afghanistan	1967	11537966	Asia	34.020	836.197138
4	Afghanistan	1972	13079460	Asia	36.088	739.981106
...	...	...	...	...	...	...
1702	Zimbabwe	2002	11926563	Africa	39.989	672.038623
1703	Zimbabwe	2007	12311143	Africa	43.487	469.709298
1704	India	2000	1350000	Asia	37.080	900.230000
1706	India	2000	13500000	NaN	37.080	900.230000
1705	India	2000	13500000	NaN	37.080	900.230000

1707 rows × 6 columns

```
df.drop(1704, axis=0)
```

	country	year	population	continent	life_exp	gdp_cap
0	Afghanistan	1952	8425333	Asia	28.801	779.445314
1	Afghanistan	1957	9240934	Asia	30.332	820.853030
2	Afghanistan	1962	10267083	Asia	31.997	853.100710
3	Afghanistan	1967	11537966	Asia	34.020	836.197138
4	Afghanistan	1972	13079460	Asia	36.088	739.981106
...	...	...	...	...	...	...
1701	Zimbabwe	1997	11404948	Africa	46.809	792.449960
1702	Zimbabwe	2002	11926563	Africa	39.989	672.038623
1703	Zimbabwe	2007	12311143	Africa	43.487	469.709298
1706	India	2000	13500000	NaN	37.080	900.230000
1705	India	2000	13500000	NaN	37.080	900.230000

1706 rows × 6 columns

```
df.loc[df.duplicated()]
```

```
Int64Index([1705], dtype='int64')
```

```
df.drop_duplicates(keep=False, inplace=True)
```

```
df.drop(1704, axis=0, inplace=True)
```

```
df
```

	country	year	population	continent	life_exp	gdp_cap
0	Afghanistan	1952	8425333	Asia	28.801	779.445314
1	Afghanistan	1957	9240934	Asia	30.332	820.853030
2	Afghanistan	1962	10267083	Asia	31.997	853.100710
3	Afghanistan	1967	11537966	Asia	34.020	836.197138
4	Afghanistan	1972	13079460	Asia	36.088	739.981106
...	...	...	...	...	...	...
1699	Zimbabwe	1987	9216418	Africa	62.351	706.157306
1700	Zimbabwe	1992	10704340	Africa	60.377	693.420786
1701	Zimbabwe	1997	11404948	Africa	46.809	792.449960
1702	Zimbabwe	2002	11926563	Africa	39.989	672.038623
1703	Zimbabwe	2007	12311143	Africa	43.487	469.709298

1704 rows × 6 columns

```
# Working with rows and columns
```

```
df.iloc[1:5, 1:4]
```

	year	population	continent
1	1957	9240934	Asia
2	1962	10267083	Asia
3	1967	11537966	Asia
4	1972	13079460	Asia

```
df.loc[1:5, "country":"continent"]
```

	country	year	population	continent
1	Afghanistan	1957	9240934	Asia
2	Afghanistan	1962	10267083	Asia

```
df.iloc[1:10:2]
```

	country	year	population	continent	life_exp	gdp_cap
1	Afghanistan	1957	9240934	Asia	30.332	820.853030
3	Afghanistan	1967	11537966	Asia	34.020	836.197138
5	Afghanistan	1977	14880372	Asia	38.438	786.113360
7	Afghanistan	1987	13867957	Asia	40.822	852.395945
9	Afghanistan	1997	22227415	Asia	41.763	635.341351

```
# Built-ijn operations in pandas
```

```
df["life_exp"].mean()
```

```
59.474439366197174
```

```
df["life_exp"].sum()
```

```
101344.44467999999
```

```
df["life_exp"].sum() / df["life_exp"].count()
```

```
59.474439366197174
```

```
df["life_exp"].max()
```

```
82.603
```

```
# Sorting
df
```

	country	year	population	continent	life_exp	gdp_cap
0	Afghanistan	1952	8425333	Asia	28.801	779.445314
1	Afghanistan	1957	9240934	Asia	30.332	820.853030
2	Afghanistan	1962	10267083	Asia	31.997	853.100710
3	Afghanistan	1967	11537966	Asia	34.020	836.197138
4	Afghanistan	1972	13079460	Asia	36.088	739.981106
...	...	...	...	...	...	...

```
df.sort_values("year", ascending=False).sort_values("life_exp") # wrong way
```

	country	year	population	continent	life_exp	gdp_cap
1292	Rwanda	1992	7290203	Africa	23.599	737.068595
0	Afghanistan	1952	8425333	Asia	28.801	779.445314
552	Gambia	1952	284320	Africa	30.000	485.230659
36	Angola	1952	4232095	Africa	30.015	3520.610273
1344	Sierra Leone	1952	2143249	Africa	30.331	879.787736
...	...	...	...	...	...	...
1487	Switzerland	2007	7554661	Europe	81.701	37506.419070
695	Iceland	2007	301931	Europe	81.757	36180.789190
802	Japan	2002	127065841	Asia	82.000	28604.591900
671	Hong Kong, China	2007	6980412	Asia	82.208	39724.978670
803	Japan	2007	127467972	Asia	82.603	31656.068060

1704 rows × 6 columns

```
df.sort_values(["year", "life_exp"], ascending=[False, True])
```

	country	year	population	continent	life_exp	gdp_cap
1463	Swaziland	2007	1133066	Africa	39.613	4513.480643
1043	Mozambique	2007	19951656	Africa	42.082	823.685621
1691	Zambia	2007	11746035	Africa	42.384	1271.211593

```
# Creating the dataframe from scratch
```

```
pd.Series([10, 20, 30], index=[2, 3, 4])
```

```
2    10
3    20
4    30
dtype: int64
```

```
# Row-oriented way
```

```
pd.DataFrame([[10, 20], [30, 40]], columns=["A", "B"])
```

	A	B
0	10	20
1	30	40

```
pd.DataFrame([[10, 20]], columns=["A", "B"])
```

	A	B
0	10	20

```
# Column-oriented way
```

```
pd.DataFrame({"A":[10, 30], "B": [20, 40]})
```

	A	B
0	10	20
1	30	40

Optional Discussion

```
X = np.arange(12).reshape((3, 4))
row = np.array([0, 1, 2])
mask = np.array([1, 0, 1, 0], dtype=bool)
# print(X[row[:, np.newaxis], mask])
# didnt understand, how to interpret the slicing in print statement.
```

didn't understand, how to interpret the slicing in print statement.

```
row
```

```
array([0, 1, 2])
```

```
X
```

```
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])
```

```
row[:, np.newaxis]
```

```
array([[0],
       [1],
       [2]])
```

```
X[row[:, np.newaxis], mask]
```

```
array([[ 0,  2],
       [ 4,  6],
       [ 8, 10]])
```

```
row
```

```
array([0, 1, 2])
```

```
mask
```

```
array([ True, False,  True, False])
```

```
print(X[row[:, np.newaxis], mask])
```

```
[[ 0  2]
 [ 4  6]
 [ 8 10]]
```

```
print(X[row, mask])
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
<ipython-input-208-75a392dca967> in <module>()
----> 1 print(X[row, mask])

IndexError: shape mismatch: indexing arrays could not be broadcast together
with shapes (3,) (2,)
```

SEARCH STACK OVERFLOW

https://stackoverflow.com/questions/46124469/shape-mismatch-indexing-arrays-could-not-be-

```
row[:, np.newaxis]
```

```
array([[0],
       [1],
       [2]])
```

```
a = np.array((1,2,3, np.array([10,11])),dtype=object)
a
```

```
array([1, 2, 3, array([10, 11])], dtype=object)
```

```
b = a.copy()
```

```
np.shares_memory(a[3], b[3])
```

```
True
```

```
np.shares_memory(a[1], b[1])
```

```
False
```

```
np.shares_memory(a, b)
```

```
False
```

```
a[3][0] = 100
```

```
a
```

```
array([1, 2, 3, array([100,  11])], dtype=object)
```

```
b
```

```
array([1, 2, 3, array([100,  11])], dtype=object)
```

0s completed at 00:07