Python Library for data Analysis

- Day Objectives
- Practice test on frequency distribution and combinatorics
- File / Data Encryption and Decryption
- Python Libraries for Data Science
- Reading data from CSV files into DataFrames
- Processing/Accessing DataFrames
- Columns
- Rows
- Updating DataFrames
- Writing DataFrame back to CSV files

In [3]:
```
######## Function to read CSV data into a DataFrame
# return the DataFrame object   csv(comma superated value)

filepath = 'DataFiles/Income.csv'
import pandas as pd
def readCSVdata(filepath):
    return pd.read_csv(filepath)

readCSVdata(filepath)


```

Out[3]:

	GEOID	State	2005	2006	2007	2008	2009	2010	2011	2012	2013
0	04000US01	Alabama	37150	37952	42212	44476	39980	40933	42590	43464	41381
1	04000US02	Alaska	55891	56418	62993	63989	61604	57848	57431	63648	61137
2	04000US04	Arizona	45245	46657	47215	46914	45739	46896	48621	47044	50602
3	04000US05	Arkansas	36658	37057	40795	39586	36538	38587	41302	39018	39919
4	04000US06	California	51755	55319	55734	57014	56134	54283	53367	57020	57528

In [8]:
```
incomedf = readCSVdata(filepath)

## Function to print all column names in a single line
# GEOID State 2005 2006 2007 2008 2009 2010 2011 2012 2013

def printDataFrameColumns(df):
    columns = df.columns
    for column in columns:
        print(column, end=" ")
    return
printDataFrameColumns(incomedf)


```

```
GEOID State 2005 2006 2007 2008 2009 2010 2011 2012 2013
```

In [57]:
```
##### Function to access a row based on a unique column value

def accessDataFramrow(df, key):

    for row in df.values:
        if key in row:
            for item in row:
                print(item, end = ' ')
            print('\n')
    return
accessDataFramrow(incomedf,'04000US06')

```

```
04000US06 California 51755 55319 55734 57014 56134 54283 53367 57020 57528
```

In [48]:
```
### Accessing a unique value based on row, column information
## Income of a state in a given year
def getRowIndex(df, rowkey):
    for i in range(len(df.values)):
        if ((df.values[i][0] == rowkey) or (df.values[i][1] == rowkey)):
            rowindex = i
    return rowindex
# getRowIndex(incomedf, 'California')

def getColumnIndex(df, columnkey):
    for i in range(len(df.values)):
            if df.columns[i] == columnkey:
                columnindex = i
    return columnindex

def valueFromRowColumn(df, rowkey, columnkey):
    for i in range(len(df.values)):
        if ((df.values[i][0] == rowkey) or (df.values[i][1] == rowkey)):
            rowindex = i
    for i in range(len(df.columns)):
        if df.columns[i] == columnkey:
            columnindex = i
    return df.values[rowindex][columnindex]
valueFromRowColumn(incomedf, 'California', '2009')

```

Out[48]:
```
56134
```

In [50]:
```
##### Function to update data based on rowkey and columnkey

def UpdateDataFromRowColumn(df, rowkey, columnkey, newdata):
    rowindex = getRowIndex(df, rowkey)
    columnindex = getColumnIndex(df, columnkey)
    row = df.values[rowindex]
    row[columnindex] = newdata
    df.loc[rowindex] = row
    return
UpdateDataFromRowColumn(incomedf, 'Arizona', '2007', 62993)
incomedf
```

Out[50]:

	GEOID	State	2005	2006	2007	2008	2009	2010	2011	2012	2013
0	04000US01	Alabama	37150	37952	42212	44476	39980	40933	42590	43464	41381
1	04000US02	Alaska	55891	56418	62993	63989	61604	57848	57431	63648	61137
2	04000US04	Arizona	45245	46657	62993	46914	45739	46896	48621	47044	50602
3	04000US05	Arkansas	36658	37057	40795	39586	36538	38587	41302	39018	39919
4	04000US06	California	51755	55319	55734	57014	56134	54283	53367	57020	57528

In [58]:
```
accessDataFramrow(incomedf, 62993)
```

```
04000US02 Alaska 55891 56418 62993 63989 61604 57848 57431 63648 61137

04000US04 Arizona 45245 46657 62993 46914 45739 46896 48621 47044 50602
```

In [ ]:
```
#########  Function to write DataFrame to CSV

incomedf.to_csv(filepath, index=False)

```

In [63]:
```
##### Function to add a new row of data to DataFrame

def addRowDataDataFrame(df, rowdata):
    lastrowindex = len(df.values)-1
    df.loc[lastrowindex+1] = rowdata
    return
rowdata = [1,2,3,4,5,56,13,14,313,12,0]
addRowDataDataFrame(incomedf, rowdata)
incomedf
```

Out[63]:

	GEOID	State	2005	2006	2007	2008	2009	2010	2011	2012	2013
0	04000US01	Alabama	37150	37952	42212	44476	39980	40933	42590	43464	41381
1	04000US02	Alaska	55891	56418	62993	63989	61604	57848	57431	63648	61137
2	04000US04	Arizona	45245	46657	62993	46914	45739	46896	48621	47044	50602
3	04000US05	Arkansas	36658	37057	40795	39586	36538	38587	41302	39018	39919
4	04000US06	California	51755	55319	55734	57014	56134	54283	53367	57020	57528
5	1	2	3	4	5	56	13	14	313	12	0

In [72]:
```
### FUcntion to delete a row in a DataFrame

def deleteRowDataFrame(df, rowkey):
    rowindex = getRowIndex(df, rowkey)
    return df.drop([rowindex])

incomedf = deleteRowDataFrame(incomedf, 1)
# incomedf.drop([5])
incomedf
```

Out[72]:

	GEOID	State	2005	2006	2007	2008	2009	2010	2011	2012	2013
0	04000US01	Alabama	37150	37952	42212	44476	39980	40933	42590	43464	41381
1	04000US02	Alaska	55891	56418	62993	63989	61604	57848	57431	63648	61137
2	04000US04	Arizona	45245	46657	62993	46914	45739	46896	48621	47044	50602
3	04000US05	Arkansas	36658	37057	40795	39586	36538	38587	41302	39018	39919
4	04000US06	California	51755	55319	55734	57014	56134	54283	53367	57020	57528

In [ ]:
```

```