Automating wranglin HW:

Write a function to convert the data file "018_HW_2x2_data.csv" from wide to "tidy" long format. Your function can assume that the input data consist of 4 columns (like our rat data), but should handle any number of rows (i.e. number of observations per cell).

In [1]:
```
import pandas as pd
import numpy as np

data = pd.read_csv("datasets/018_HW_2x2_data2.csv")
data
```

Out[1]:

	Condition_A_Level_1	Condition_A_Level_2	Condition_B_Level_1	Condition_B_Level_2
0	10	20	30	40
1	15	25	35	45
2	20	30	40	50
3	25	35	45	55
4	30	40	50	60
5	35	45	55	65

In [2]:
```
def turntotidy(filename, x = "Data", y = "Condition", z = "Level"):
    '''
    -turntotidy() takes a csv file and can turn its data into tidy
    -default column names are Data, Condition, and Level
    -enter own column names after filename argument to change default names
    '''

    import numpy as np
    import pandas as pd

    dat = pd.read_csv(filename)          # reads in filename inputted
    dat = dat.to_numpy()                 # convert to a numpy array

    observations, groups = dat.shape   #gives us shape of data (obs = obs per g
    length = observations * groups          #gives us how long the array needs

    #creating the array:
    values= np.reshape(dat, (length, 1), order = "F")  #arranges data in proper
    values = np.squeeze(values)

    # create variable columns:
        #outer grouping variable:
    condition = pd.Series(["A", "B"])    #create a series containing variable c
    condition = condition.repeat((length/2))
    condition = condition.reset_index(drop = True) #make sure indexes are desce

        #inner grouping variable:
    level = pd.Series([1, 2])
    level = level.repeat((length/4))
    level = pd.concat([level]*2, ignore_index=True)
```

```
    #create the dataframe using a dictionary with the two series i just created
    datadict= pd.DataFrame(
        {x : values,
         y : condition,
         z : level }
    )
    return(datadict)
```

In [3]:
```
help(turntotidy)
```

```
Help on function turntotidy in module __main__:

turntotidy(filename, x='Data', y='Condition', z='Level')
    -turntotidy() takes a csv file and can turn its data into tidy
    -default column names are Data, Condition, and Level
    -input own column names after filename argument to change
```

In [24]:
```
turntotidy("datasets/018_HW_2x2_data2.csv", "Observation", "Letter", "Number")
```

Out[24]:

	Observation	Letter	Number
0	10	A	1
1	15	A	1
2	20	A	1
3	25	A	1
4	30	A	1
5	35	A	1
6	20	A	2
7	25	A	2
8	30	A	2
9	35	A	2
10	40	A	2
11	45	A	2
12	30	B	1
13	35	B	1
14	40	B	1
15	45	B	1
16	50	B	1
17	55	B	1
18	40	B	2
19	45	B	2
20	50	B	2
21	55	B	2
22	60	B	2
23	65	B	2

In [25]:
```
turntotidy("datasets/018_HW_2x2_data.csv")
```

Out[25]:

	Data	Condition	Level
0	10	A	1
1	15	A	1
2	20	A	2
3	25	A	2
4	30	B	1
5	35	B	1
6	40	B	2
7	45	B	2