PANDAS - Assignment

By Prakash Ghosh

Problem 1. How-to-count-distance-to-the-previous-zero

For each value, count the difference of the distance from the previous zero (or the start of the Series, whichever is closer) and if there are no previous zeros,print the position

Consider a DataFrame df where there is an integer column
{'X':[7, 2, 0, 3, 4, 2, 5, 0, 3, 4]}

The values should therefore be [1, 2, 0, 1, 2, 3, 4, 0, 1, 2]. Make this a new column 'Y'.

In [6]:
```
import pandas as pd
import numpy as np

# DataFrame Creation
df = pd.DataFrame({'X':[7, 2, 0, 3, 4, 2, 5, 0, 3, 4]})

last_zero=0
last_zero_ind=[]

# Count the difference of the distance from the previous zero
for i in df['X']:
    if i==0:
        last_zero=0
    else:
        last_zero=last_zero+1
    last_zero_ind.append(last_zero)

print('\nThe values for the previous zeros:\t' , last_zero_ind)

# Make this list as new column of the DataFrame
df['Y'] = last_zero_ind

df
```

```
The values for the previous zeros:       [1, 2, 0, 1, 2, 3, 4, 0, 1, 2]
```

Out[6]:

	X	Y
0	7	1
1	2	2
2	0	0
3	3	1
4	4	2
5	2	3
6	5	4
7	0	0
8	3	1
9	4	2

Problem 2. Create a DatetimeIndex that contains each business day of 2015 and use it to index a Series of random numbers.

In [35]:
```
import pandas as pd
import numpy as np

#Create a DatetimeIndex that contains each business day of 2015
dt_ind = pd.date_range('2015-01-01', '2015-12-31', freq='B')
# Series of Random Numbers
s = np.round((np.random.rand(len(dt_ind))*1000),0)

# Make the DatetimeIndex to index of the Series of Random Numbers
df =pd.DataFrame({'Date_Index':dt_ind})
df['Rand_Num'] = pd.Series(s)
df.set_index(['Date_Index'],inplace=True)

#  Show First 20 Values from the DataFrame
df.head(30)
```

Out[35]:

	Rand_Num
Date_Index	
2015-01-01	963.0
2015-01-02	402.0
2015-01-05	115.0
2015-01-06	665.0
2015-01-07	604.0
2015-01-08	731.0
2015-01-09	770.0
2015-01-12	562.0
2015-01-13	593.0
2015-01-14	672.0
2015-01-15	653.0
2015-01-16	948.0
2015-01-19	427.0
2015-01-20	961.0
2015-01-21	538.0
2015-01-22	847.0
2015-01-23	124.0
2015-01-26	270.0
2015-01-27	920.0
2015-01-28	575.0
2015-01-29	459.0
2015-01-30	656.0

	Rand_Num
Date_Index	
2015-02-02	22.0
2015-02-03	116.0
2015-02-04	496.0
2015-02-05	7.0
2015-02-06	152.0
2015-02-09	444.0
2015-02-10	972.0
2015-02-11	708.0

Problem 3. Find the sum of the values in s for every Wednesday

In [36]:
```
# Add a Column in DataFrame with Weekday Name
df['Week_Day'] = dt_ind.day_name()

# Create a list where
#       if the weekday name is Wednesday then assign its value as sum of the Random numbers upto previous Wednesday
wed_sum=0
lst_wed_sum =[]
for i in df.index:
    if df.loc[i].Week_Day=='Wednesday':
        wed_sum=wed_sum+df.loc[i].Rand_Num
        lst_wed_sum.append(wed_sum)
        wed_sum=0
    else:
        wed_sum=wed_sum+df.loc[i].Rand_Num
        lst_wed_sum.append('')

# Add the list to DataFrame it will show the sum of the values in s for every Wednesday in the DataFrame
df['Wed_Sum'] = lst_wed_sum

#  Show First 20 Values from the DataFrame
df.head(30)
```

Out[36]:

	Rand_Num	Week_Day	Wed_Sum
Date_Index			
2015-01-01	963.0	Thursday	
2015-01-02	402.0	Friday	
2015-01-05	115.0	Monday	
2015-01-06	665.0	Tuesday	
2015-01-07	604.0	Wednesday	2749
2015-01-08	731.0	Thursday	
2015-01-09	770.0	Friday	
2015-01-12	562.0	Monday	
2015-01-13	593.0	Tuesday	
2015-01-14	672.0	Wednesday	3328
2015-01-15	653.0	Thursday	
2015-01-16	948.0	Friday	
2015-01-19	427.0	Monday	
2015-01-20	961.0	Tuesday	
2015-01-21	538.0	Wednesday	3527
2015-01-22	847.0	Thursday	
2015-01-23	124.0	Friday	
2015-01-26	270.0	Monday	
2015-01-27	920.0	Tuesday	
2015-01-28	575.0	Wednesday	2736
2015-01-29	459.0	Thursday	
2015-01-30	656.0	Friday	

	Rand_Num	Week_Day	Wed_Sum
Date_Index			
2015-02-02	22.0	Monday	
2015-02-03	116.0	Tuesday	
2015-02-04	496.0	Wednesday	1749
2015-02-05	7.0	Thursday	
2015-02-06	152.0	Friday	
2015-02-09	444.0	Monday	
2015-02-10	972.0	Tuesday	
2015-02-11	708.0	Wednesday	2283

Problem 4. Average For each calendar month

In [31]:
```
# Add a Column in DataFrame with Monthe Name
df['Month_Name'] = dt_ind.month_name()

# Group the mean of the Rand_Num by Month_Name
df.groupby(['Month_Name'])['Rand_Num'].mean()
```

Out[31]:
```
Month_Name
April        478.318182
August       425.095238
December     551.391304
February     533.450000
January      455.727273
July         423.347826
June         557.272727
March        440.954545
May          601.285714
November     586.142857
October      491.181818
September    598.181818
Name: Rand_Num, dtype: float64
```

Problem 5. For each group of four consecutive calendar months in s, find the date on which the highest value occurred.

In [32]:
```
# Add a Column in DataFrame with Four_Consecutive_Month Indicator by the following logic
# Devide the month by 4 and take ceil of it,
#     for the first 4 months result is 1,
#     for the next 4 monthes it is 2 and
#     for the rest 4 Months it is 3
df['Four_Consecutive_Month'] = np.ceil(dt_ind.month/4)

# Create anoter DataFrame df2 to find the Max number for Four_Consecutive_Month
df2= pd.DataFrame(df.groupby(['Four_Consecutive_Month'])['Rand_Num'].max())

# Loop through the new DataFrame to find the Max value the find the date in the main DataFrame
df3=pd.DataFrame()
for i in df2.index:
    df3=df3.append( df[(df['Four_Consecutive_Month']==i) & (df['Rand_Num']==df2.loc[i].Rand_Num)])

df3.drop(['Four_Consecutive_Month', 'Wed_Sum'], axis=1,inplace=True)
df3
```

Out[32]:

	Rand_Num	Week_Day	Month_Name
Date_Index			
2015-04-28	999.0	Tuesday	April
2015-07-21	996.0	Tuesday	July
2015-10-23	1000.0	Friday	October