Business Analytics Programming - Spring 2019, Practice Set # 4

1. What is x?

```
a=pd.Series([5,4,3,2,1],index=['peach','orange','melon','cherry','apple'])
b=pd.Series([50,40,30,20,10],index=['apple','cherry','melon','orange','peach'])
df=pd.DataFrame({'a':a,'b':b})
x=sum(df.iloc[2,0:2])

```

2. What is x?

```
a=pd.Series([5,4,3,2,1],index=['apple','cherry','melon','orange','peach'])
b=pd.Series([50,40,30,20,10],index=['apple','cherry','melon','orange','peach'])
df2=pd.DataFrame({'a':a,'b':b})
x=sum(df2.iloc[3,[0,1]])

```

3. What is x?

```
a=pd.Series([5,4,3,2,1],index=['peach','orange','melon','cherry','apple'])
b=pd.Series([50,40,30,20,10],index=['peach','orange','melon','cherry','apple'])
df3=pd.DataFrame({'a':a,'b':b})
x=df3.iloc[2]['b']

```

4. What is x?

```
a=pd.Series([5,4,3,2,1],index=['peach','orange','melon','cherry','apple'])
b=pd.Series([50,30,20,10],index=['peach','melon','cherry','apple'])
df4=pd.DataFrame({'a':a,'b':b})
x=df4.iloc[3].sum()

```

5. What is y?

```
x=np.arange(5.)
x[2]=3.5
y=sum(x)

```

6. What is y?

```
x=np.arange(5)
x[2]=x[3]*3.5
y=sum(x)

```

7. What is z?

```
x=np.array([0,2,4,6,8])
y=x*1.5
z=y
z[1:3]=y[[0,4]]

```

8. **What is z?**

```
x=np.array([0,2,4,6,8])
y=x*1.5
z=y
y[2]=z[3]+2.5

```

9. **What is z?**

```
x=[4,3,2,1,0]
y=[10,11,13,14,15]
z=[]
for i in x:
  if i>2:
    z.append(y[i])
  else:
    z.append(i)

```

10. **Fix the code.**

```
t_loc = api.trends.available()
print(t_loc)

from pandas.io.json import json_normalize

df_loc=json_normalize(t_loc)
df_loc.country.value_counts()

dfNew=df_loc[df_loc[’name’].str.contains(’New’)]
ny=dfNew.loc[dfNew.name==’New York’,’woeid’]

ny_trend = trends.place(_id=ny)

```

11. **Fix the code.**

```
t_loc = api.trends.available()
print(t_loc)

from pandas.io.json import json_normalize

df_loc=json_normalize(t_loc)
df_loc.country.value_counts()

dfNew=df_loc[df_loc[’name’].str.contains(’New’)]
ny=dfNew.loc[dfNew.name==’New York’,’woeid’]

ny_trend = trends.place(_id=ny)

```

12. **Which pandas function normalizes JSON objects?**

13. **The API function returns a JSON object within a JSON object, that is stored in the variable x. The JSON object within a JSON object is named $screen\_name$. Normalize the inner JSON object into a pandas dataframe named df.**