Business Analytics Programming Spring 2019, Midterm # 2 (Newark)- Solution

	a	b	text
0	10	5	Thanks so much for driving me home.
1	12	1	Thanks so much for cooking dinner. I really appreciate it.
2	20	100	Excuse me sir, you dropped your wallet.
3	15	25	I'm sorry for the mess. I wasn't expecting anyone today.
4	30	30	My name is Sophie and I'm learning English.

```
import numpy as np
import pandas as pd

```

1. **A,C- Create table df2, with just the columns 'text' and 'b'?**
   Solution:df2=df[['text','b']]
   B,D - Create table df2, with just the columns 'text' and 'a'?
   Solution:df2=df[['text','a']]

2. **What values will the code give you:**
   A: ['appreciate', 'it.','batman']

```
df2['text2']=df2.text.str.split()
df2.loc[1,'text2'].append('batman')
df2['text3']=df2.text2.apply(lambda x: x[8:])
df2.loc[1,'text3']

```

   B: ['I'm', 'sorry', 'superman']

```
df2['text2']=df2.text.str.split()
df2.loc[3,'text2'][2]='superman'
df2['text3']=df2.text2.apply(lambda x: x[0:3])
df2.loc[3,'text3']

```

   C:['greenlantern']

```
df2['text2']=df2.text.str.split()
df2.loc[4,'text2'].append('greenlantern')
df2['text3']=df2.text2.apply(lambda x: x[8:])
df2.loc[4,'text3']

```

   D: ['Thanks', 'flash', 'much']

```
df2['text2']=df2.text.str.split()
df2.loc[0,'text2'][1]='flash'
df2['text3']=df2.text2.apply(lambda x: x[0:3])
df2.loc[0,'text3']

```

3. **What is the value of x?**
   A: 34

```
df2['c']=df2.b.apply(lambda x: x**2+x*2-1)
x=df2.iloc[0]['c']

```

B: 167

```
df2['c']=df2.a.apply(lambda x: x**2+x*2-1)
x=df2.iloc[1]['c']

```

C: 2

```
df2['c']=df2.b.apply(lambda x: x**2+x*2-1)
x=df2.iloc[1]['c']

```

D: 400

```
df2['c']=df2.a.apply(lambda x: x**2)
x=df2.iloc[2]['c']

```

4. **What is the value of x?**
   A: ['me', 'dropped']

```
df2['text2']=df2.text.str.split()
df2['text4']=df2.text2.apply(np.array)
df2['text5']=df2.text4.apply(lambda x: x[np.isin(x,['dropped','me'])])
x=df2.loc[2,'text5']

```

B: ['sorry']

```
df2['text2']=df2.text.str.split()
df2['text4']=df2.text2.apply(np.array)
df2['text5']=df2.text4.apply(lambda x: x[np.isin(x,['mess','sorry'])])
x=df2.loc[3,'text5']

```

C: ['name', 'Sophie']

```
df2['text2']=df2.text.str.split()
df2['text4']=df2.text2.apply(np.array)
df2['text5']=df2.text4.apply(lambda x: x[np.isin(x,['Sophie','name'])])
x=df2.loc[4,'text5']

```

D: ['sorry', 'expecting']

```
df2['text2']=df2.text.str.split()
df2['text4']=df2.text2.apply(np.array)
df2['text5']=df2.text4.apply(lambda x: x[np.isin(x,['expecting','sorry'])])
x=df2.loc[3,'text5']

```

5. **What is the value of x?**
   A: ['so', 'for']

```
df2['text2']=df2.text.str.split()
df2['text4']=df2.text2.apply(np.array)
df2['text6']=df2.text4.apply(lambda x: x[[1,3]])
l=[]
df2.text6.apply(lambda x: l.extend(x.tolist()))
x=l[2:4]

```

B: ['much', 'cooking']

```
df2['text2']=df2.text.str.split()
df2['text4']=df2.text2.apply(np.array)
df2['text6']=df2.text4.apply(lambda x: x[[2,4]])
l=[]
df2.text6.apply(lambda x: l.extend(x.tolist()))
x=l[2:4]
```

C: ['dinner.', 'you']

```
df2['text2']=df2.text.str.split()
df2['text4']=df2.text2.apply(np.array)
df2['text6']=df2.text4.apply(lambda x: x[[3,5]])
l=[]
df2.text6.apply(lambda x: l.extend(x.tolist()))
x=l[3:5]
```

D: ['me', 'dropped']

```
df2['text2']=df2.text.str.split()
df2['text4']=df2.text2.apply(np.array)
df2['text6']=df2.text4.apply(lambda x: x[[1,4]])
l=[]
df2.text6.apply(lambda x: l.extend(x.tolist()))
x=l[4:6]
```

6. What is the value of x?

A: 40

```
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','apple','cherry'])
df3=pd.DataFrame({'a':a,'b':b})
x=df3.iloc[3]['b']
```

B: 1

```
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[4]['a']
```

C: 2

```
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[3]['a']
```

D: 4

```
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','apple','cherry'])
df3=pd.DataFrame({'a':a,'b':b})
x=df3.iloc[3]['a']
```

7. What is the value of x?

A: 20

```
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[3]['b']

```

B: 5

```
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','apple','cherry'])
df3=pd.DataFrame({'a':a,'b':b})
x=df3.iloc[4]['a']

```

C: 4

```
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','apple','cherry'])
df3=pd.DataFrame({'a':a,'b':b})
x=df3.iloc[3]['a']

```

D: 1

```
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[4]['a']

```

8. **The API function returns a JSON object within a JSON object, that is stored in the variable A. The JSON object within a JSON object is named B. Normalize the inner JSON object into a pandas dataframe named df.**
   A: df=json_normalize(A,'B')

   The API function returns a JSON object within a JSON object, that is stored in the variable D. The JSON object within a JSON object is named C. Normalize the inner JSON object into a pandas dataframe named df.
   B: df=json_normalize(D,'C')

   The API function returns a JSON object within a JSON object, that is stored in the variable B. The JSON object within a JSON object is named A. Normalize the inner JSON object into a pandas dataframe named df.
   C: df=json_normalize(B,'A')

   The API function returns a JSON object within a JSON object, that is stored in the variable C. The JSON object within a JSON object is named D. Normalize the inner JSON object into a pandas dataframe named df.
   D: df=json_normalize(C,'D')

9. **What is the value of z?**
   A: [0. , 1. , 2. , 4.2, 4.]

```
x=np.array([0,2,4,6,8])
y=x*.5
z=y
y[3]=y[4]+0.2

```

B: [0. , 1. , 5.5, 3. , 4.]

```
x=np.array([0,2,4,6,8])
y=x*.5
z=y
y[2]=y[3]+2.5
```

C: [0. , 2.5, 2. , 3. , 4.]

```
x=np.array([0,2,4,6,8])
y=x*.5
z=y
y[1]=y[2]+.5
```

D: [2.5, 1. , 2. , 3. , 4.]

```
x=np.array([0,2,4,6,8])
y=x*.5
z=y
y[0]=y[1]+1.5
```

10. **Using table df, what is the sum of all the prices of each type of candy (the total sum of snickers, the total sum of mounds,etc.) Write the code to do it.**
    A: df.groupby('Candy')['Price'].sum()
    B: df.groupby('Candy')['Price'].mean()
    C: df.groupby('Candy')['Price'].max()
    D: df.groupby('Candy')['Sales'].sum()

	Candy	Price	Sales	Date
a	snickers	2	100	Jan-1-2017
b	mounds	3	50	May-10-2017
c	twix	4	60	Feb-5-2017
d	twix	2	200	Mar-7-2017
e	snickers	5	400	Apr-12-2017
f	mounds	4	240	Nov-8-2017
g	dove	7	200	Jul-13-2017

11. **Using the above table, change all the prices of 'dove' to 15, where Sales is less than 160 and greater than 110.**
    A: df.loc[(df.Candy=='dove') & (df.Sales>110) & (df.Sales <160), 'Price']=15
    B: df.loc[(df.Candy=='twix') & (df.Sales>90) & (df.Sales <140), 'Price']=8
    C: df.loc[(df.Candy=='snickers') & (df.Sales>70) & (df.Sales <120), 'Price']=10
    D: df.loc[(df.Candy=='mounds') & (df.Sales>50) & (df.Sales <100), 'Price']=5

12. **What is sum(s[[5,2]])? A: 3**

```
s=pd.Series([1,2,3,4,5,6],index=[2,5,3,7,1,9])
```

What is sum(s[[1,4]])? B: 11

```
s=pd.Series([1,2,3,4,5,6],index=[2,5,3,6,1,4])
```

What is sum(s[[3,2]])? C: 4

```
s=pd.Series([1,2,3,4,5,6],index=[2,5,3,6,1,4])

```

What is sum(s[[3,5]])? D: 5

```
s=pd.Series([1,2,3,4,5,6],index=[2,5,3,6,1,4])

```

13. **What is sum(s['a':'c'])? A: 6**

```
s=pd.Series([1,2,3,4,5,6],index=['a','b','c','d','d','e'])

```

What is sum(s['b':'d'])? B: 14

```
s=pd.Series([1,2,3,4,5,6],index=['a','b','c','d','d','e'])

```

What is sum(s['c':'e'])? C: 18

```
s=pd.Series([1,2,3,4,5,6],index=['a','b','c','d','d','e'])

```

What is sum(s['a':'d'])? D: 15

```
s=pd.Series([1,2,3,4,5,6],index=['a','b','c','d','d','e'])

```

14. **After executing the below code, how many rows will be in df?**
A: 60

```
df2 = pd.DataFrame()
df = pd.DataFrame()
mid=0
for i in range(34):
  if i==0:
    tjson=api.statuses.user_timeline(screen_name="realDonaldTrump",tweet_mode='extended',
  count = 60)
  else:
    tjson=api.statuses.user_timeline(screen_name="realDonaldTrump",tweet_mode='extended',
  count = 60,max_id = mid)
  if len(tjson)>0:
    df=json_normalize(tjson)
    mid=df['id'].min()
    mid=mid-1
    df2 = pd.concat([df2, df], ignore_index=True)

```

B: 80

```
df2 = pd.DataFrame()
df = pd.DataFrame()
mid=0
for i in range(34):
  if i==0:
    tjson=api.statuses.user_timeline(screen_name="realDonaldTrump",tweet_mode='extended',
  count = 80)
  else:
    tjson=api.statuses.user_timeline(screen_name="realDonaldTrump",tweet_mode='extended',
  count = 80,max_id = mid)
  if len(tjson)>0:
    df=json_normalize(tjson)
    mid=df['id'].min()
    mid=mid-1
    df2 = pd.concat([df2, df], ignore_index=True)

```

C: 120

```
df2 = pd.DataFrame()
df = pd.DataFrame()
mid=0
for i in range(34):
  if i==0:
    tjson=api.statuses.user_timeline(screen_name="realDonaldTrump",tweet_mode='extended',
  count = 120)
  else:
    tjson=api.statuses.user_timeline(screen_name="realDonaldTrump",tweet_mode='extended',
  count = 120,max_id = mid)
  if len(tjson)>0:
    df=json_normalize(tjson)
    mid=df['id'].min()
    mid=mid-1
    df2 = pd.concat([df2, df], ignore_index=True)

```

D: 140

```
df2 = pd.DataFrame()
df = pd.DataFrame()
mid=0
for i in range(34):
  if i==0:
    tjson=api.statuses.user_timeline(screen_name="realDonaldTrump",tweet_mode='extended',
  count = 140)
  else:
    tjson=api.statuses.user_timeline(screen_name="realDonaldTrump",tweet_mode='extended',
  count = 140,max_id = mid)
  if len(tjson)>0:
    df=json_normalize(tjson)
    mid=df['id'].min()
    mid=mid-1
    df2 = pd.concat([df2, df], ignore_index=True)

```