In class Practice 2/28/2020 numpy and pandas

Part I

1. A view is also known as _____.

   a. deep copy
   b. auto procedure
   c. temporary copy
   d. shallow copy.

Given the following snippet, answer questions (2) and (3)

```
import numpy as np
arr = np.arange(24)
arr = arr.reshape (4,6)
```

2. What is arr.shape?

   a. (3, 5)
   b. (4, 6)
   c. 24
   d. 23

3. What is the type of arr?

   a. numpy.array
   b. numpy.ndarray
   c. numpy.frame
   d. numpy.DataFrame

Given the following array brr, answer the questions (4) to (6)

```
array([[ 0,  1,  2,  3,  4,  5],
       [ 6,  7,  8,  9, 10, 11],
       [12, 13, 14, 15, 16, 17],
       [18, 19, 20, 21, 22, 23]])
```

4. Which statement will generate the following result?

```
array([12, 13, 14, 15, 16, 17])
```

   a. brr[1]
   b. brr[2]
   c. brr[3]
   d. none of the above

5. Which statement will generate the following result?

```
array([[ 0,  1,  2,  3,  4,  5],
       [18, 19, 20, 21, 22, 23]])
```

   a. brr[(1,4)]
   b. brr [[0, 3]]
   c. brr[(0, 3)]

d. none of the above

6. Which statement will generate the following result?

```
array([[ 1,  2,  3,  4],
       [ 7,  8,  9, 10],
       [13, 14, 15, 16],
       [19, 20, 21, 22]])
```

a. `brr[1:4, 1:4]`
b. `brr[:, 1:5]`
c. `brr[0:3, :]`
d. none of the above

Consider the following pandas snippet, answer questions (7) to (9)

```
import pandas as pd
```

```
gr = pd.Series ([90, 80, 70])
```

7. What is the result of gr.count()?

a. 1
b. 2
c. 3
d. 80

8. What is type(gr.mean())?

a. `float`
b. `int`
c. `str`
d. `numpy.int64`

9. What is gr.max()?

a. `90`
b. `80.0`
c. `70`
d. `none of the above`

10. Which of the following cannot generate the result below? Assume we import pandas as pd.

```
James     80
Paul      90
Peter    100
dtype: int64
```

a. `gr = pd.Series([80, 100, 90], axis0 =['James', 'Peter', 'Paul'])`
b. `gr = pd.Series ({'James':80, 'Peter':100, 'Paul':90})`
c. `gr = pd.Series([80, 100, 90], index=['James', 'Peter', 'Paul'])`
d. `none of the above`

11. Which of the following will generate the following result when date_range of pandas is used?

```
DatetimeIndex(['2020-02-23', '2020-03-01', '2020-03-08', '2020-03-15'], dtype
='datetime64[ns]', freq='W-SUN')
```

a. `dates = pd.date_range('2020-02-21', '2020-03-15', freq='W')`
b. `dates = pd.date_range('2020-02-21', '2020-03-15', freq='M')`
c. `dates = pd.date_range('2020-02-21', '2020-03-15', freq='S')`
d. `dates = pd.date_range('2020-02-21', '2020-03-15', freq='H')`

12. For the following pandas snippet, what is the output?

```
house_hold_info = pd.read_csv('housing.csv')

house_hold_info.head()
```

a. **The file record heading and first five records in housing.csv file will be returned**
b. **The first row which is the heading of file will be output**
c. **The first record in housing.csv will be returned**
d. **none of the above**

Part II Short Answers

1. Use numpy to fill 2 x 3 array with zeros `a = np.zeros((2,3))`
2. Use numpy to fill 3 x 3 array with ones assume nmupy was imported as np. `a = np.ones((3,3))`
3. Use numpy to fill 3 x 5 array with 4s assume nmupy was imported as np. `a = np.full((3,5),4)`
4. Use arrange to create 2 x 2 array containing numbers 0-3 as

```
array([[0, 1],
       [2, 3]])
```

`a = np.arange(4).reshape(2,2)`

After creating this array, use broadcasting to perform the following (5-7)

5. Cube every element of the array `a**3`
6. Subtract every element by 12 `a-12`
7. Multiply every element by 21 `a*21`

```
temps = OrderedDict()
temps = {'Mon':[48,69], 'Tue':[51,73], 'Wed':[66,72], 'Thu':[55,67], 'Fri':[65,70]}
weekday_temperatures= pd.Dataframe(temps,index=['Low','High']
weekday_temperatures
weekday_temperatures.loc[:,'Mon':'Wed']
weekday_temperatures.loc['High',:]
pd.set_option('precision',2)
weekday_temperatures.mean()
weekday_temperatures.mean(axis=1)
```

Part III

1. The *temps* is a OrderedDict which represents low and high of temperature of weekdays.
   temps = {'Mon': [48, 69], 'Tue': [51, 73], 'Wed': [66, 72], 'Thu': [55, 67], 'Fri': [65, 70]}
   (a) Covert temps into pandas DataFrame with 'Low' and 'High' as indices, then display the DataFrame.
   (b) Use the column names to select 'Mon' through 'Wed'
   (c) Use index 'High' to select high temperatures for each day
   (d) Set float-point precision to 2 and compute average temperature for each day
   (e) Compute average temperature for Low and High

2. Create numpy array array1

```
array([[0, 1],
       [2, 3]])
```

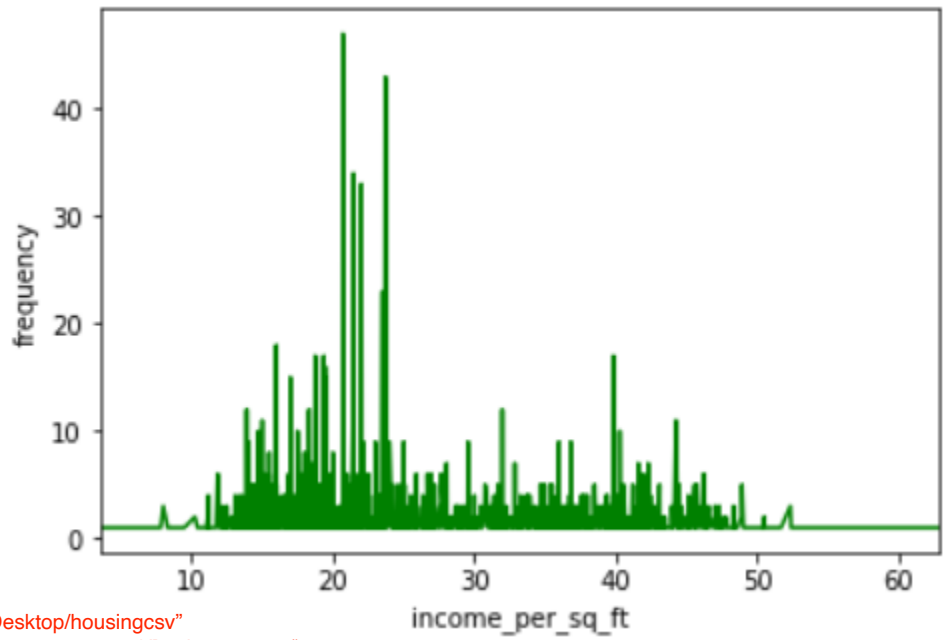
   and array2

```
array([[4, 5],
       [6, 7]])
```

```
array1=np.array([[0,1],[2,3]])
array2=np.array([[4,5],[6,7]])
array3=np.vstack((array1,array2))
array3
array4=np.append(array1,array2,axis=1)
array4
array5=np.vstack((array4,array4))
array5
array6=np.hstack((array3,array3))
array6
```

   (a) Create a 4 x 2 array3 with array1 stacked on top of array2
   (b) Create a 2 x 4 array4 with array2 to the right of array1
   (c) Use vertical stacking to create a 4 x 4 array5 using 2 copies of array4
   (d) Use horizontal stacking to create a 4 x 4 array6 using 2 copies of array3

3. Read in housing.csv and use matplotlib.pyplot to plot a diagram using income_per_sq_ft as x axis.




```
url="/Users/yifuhe/Desktop/housingcsv"
house_hold_info['income_per_sq_ft'].value_counts()
h=f.sort_index()
plt.xlabel('income_per_sq_ft')
plt.ylabel('frequency')
h.plot(c='q')
plt.show()
```