Handout for Session 7 (with Solutions)

0. Review of Algorithmic Thinking

1) Describe the task succintly and precisely.
2) Decompose the task into components and describe how to do each in English.
3) Translate each component into code and test them independently.
4) Combine together and test.

1. Paper Coding Exercise for Case 9

Q1: Without using a computer, translate the following component of case 9 into code by hand-writing into the space below.

Component: Given a list named `curVal`, representing the valuation of the current customer for the two products, as well as list named `priceVector`, representing the price of the two products, print `"Purchase product 0"` if the customer purchases the first product; print `"Purchase product 1"` if the customer purchases the second product; print `"Purchase nothing"` if the customer purchases neither.

```
[1]: # Input
     curVal=[25,15]
     priceVector=[25,10]

     # Write your code below
```

After you are done, trace through your code several times with different values of `curVal` and `priceVector` and check for syntax or logical errors.

Q2: Exchange your code with a neighbor and help one another check for errors. If you find an error, explain it to your neighbor with concrete inputs.

(Optional exercise if you finish early): Modify your code to work when `curVal` and `priceVector` are lists of arbitrary length. (Still do this on a piece of paper without the help of a computer.)

2. Pandas Series

2.1 Creating a Series Object (in 3 Ways)

```
[2]: import pandas as pd
     s=pd.Series([5,6,4])
     s

0    5
1    6
2    4
dtype: int64

[3]: s=pd.Series([5,6,4],index=['apple','orange','grape'])
     s

apple     5
orange    6
grape     4
dtype: int64
```

```
[4]: s=pd.Series({'apple':5,'orange':6,'grape':4})
     s

apple     5
orange    6
grape     4
dtype: int64

[5]: s=pd.Series()
     s['apple']=5
     s['orange']=6
     s['grape']=4
     s

apple     5
orange    6
grape     4
dtype: int64
```

2.2 Indexing a Series Object (in 3 Ways)

```
[6]: s[1]

6

[7]: s.iloc[1]

6

[8]: s.loc['orange']

6

[9]: s[:2]

apple     5
orange    6
dtype: int64

[10]: s.iloc[:2]

apple     5
orange    6
dtype: int64

[11]: s.loc[:'orange']

apple     5
orange    6
dtype: int64
```

Q3-a: Create the following `Series` object using three ways.

```
Fritos     20
Cheetos    15
Lays       25
dtype: int64
```

```
[12]: t=pd.Series({'Fritos':20,'Cheetos':15,'Lays':25})
      t
```

```
Fritos     20
Cheetos    15
Lays       25
dtype: int64
```

```
[13]: t=pd.Series([20,15,25],index=['Fritos','Cheetos','Lays'])
      t
```

```
Fritos     20
Cheetos    15
Lays       25
dtype: int64
```

```
[14]: t=pd.Series([])
      t['Fritos']=20
      t['Cheetos']=15
      t['Lays']=25
      t
```

```
Fritos     20
Cheetos    15
Lays       25
dtype: int64
```

Q3-b: Obtain the element for "Lays" using five ways.

```
[15]: t[2]
```

```
25
```

```
[16]: t[-1]
```

```
25
```

```
[17]: t.iloc[2]
```

```
25
```

```
[18]: t.iloc[-1]
```

```
25
```

```
[19]: t.loc['Lays']
```

```
25
```

Q3-c: Obtain everything but the first element using three ways.

```
[20]: t[1:]

Cheetos    15
Lays       25
dtype: int64

[21]: t.iloc[1:]

Cheetos    15
Lays       25
dtype: int64

[22]: t.loc['Cheetos':]

Cheetos    15
Lays       25
dtype: int64
```

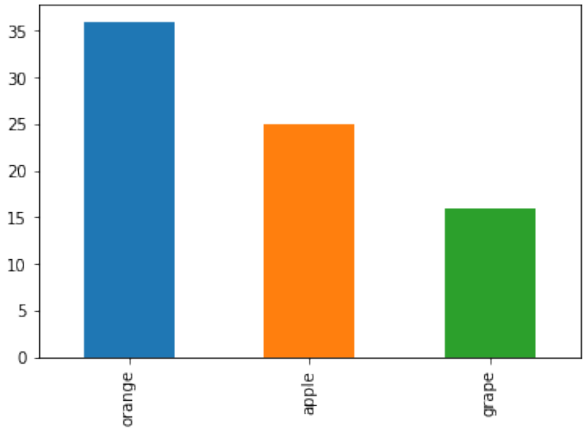
2.3 Manipulating a Series Object

```
[23]: s+1

apple     6
orange    7
grape     5
dtype: int64

[24]: s+s

apple     10
orange    12
grape      8
dtype: int64

[25]: import numpy as np
      np.exp(s)

apple     148.413159
orange    403.428793
grape      54.598150
dtype: float64

[26]: s.sort_index()

apple     5
grape     4
orange    6
dtype: int64

[27]: s.sort_index(ascending=False)

orange    6
grape     4
apple     5
dtype: int64
```

```
[28]: s.sort_values()

grape     4
apple     5
orange    6
dtype: int64

[29]: s.sort_values(ascending=False)

orange    6
apple     5
grape     4
dtype: int64

[30]: import matplotlib.pyplot as plt
      s.plot(kind='bar')
      plt.show()

<Figure size 640x480 with 1 Axes>

[31]: s.shape

(3,)

[32]: len(s)

3

[33]: for e in s:
          print (e)

5
6
4


[34]: for i in s.index:
          print(i,s[i])

apple 5
orange 6
grape 4
```

Q4: Create another version of the above bar chart so that the bars are sorted in descending order, and the values are squares of what they are now.

```
[35]: s2=(s**2).sort_values(ascending=False)
      s2.plot(kind='bar')
      plt.show()
```



3. Pandas DataFrame

3.1 Creating a DataFrame (3 Ways)

```
[36]: import pandas as pd
      df=pd.DataFrame([[5,3],[6,2],[4,1]])
      df
```

```
   0  1
0  5  3
1  6  2
2  4  1
```

```
[37]: df=pd.DataFrame([[5,2],[6,1],[4,3]],\
                      index=['apple','orange','grape'],\
                      columns=['Number','Rank'])
      df
```

```
        Number  Rank
apple        5     2
orange       6     1
grape        4     3
```

```
[38]: df=pd.DataFrame({'Number':[5,6,4],'Rank':[2,1,3]},index=['apple','orange','grape'])
      df
```

```
        Number  Rank
apple        5     2
orange       6     1
grape        4     3
```

```
[39]: df=pd.DataFrame()
      df.loc['apple','Number']=5
      df.loc['apple','Rank']=2
      df.loc['orange','Number']=6
      df.loc['orange','Rank']=1
      df.loc['grape','Number']=4
      df.loc['grape','Rank']=3
      df=df.astype(int)
      df
```

```
        Number  Rank
apple        5     2
orange       6     1
grape        4     3
```

3.2 Indexing a DataFrame (3 ways)

```
[40]: df['Number']

apple     5
orange    6
grape     4
Name: Number, dtype: int64

[41]: df['Number'][0]

5

[42]: df.iloc[:,0]

apple     5
orange    6
grape     4
Name: Number, dtype: int64

[43]: df.iloc[0,0]

5

[44]: df.loc[:,'Number']

apple     5
orange    6
grape     4
Name: Number, dtype: int64

[45]: df.loc['apple','Number']

5
```

Q5-a: Obtain the second column of the DataFrame `df` in at least three ways.

```
[46]: df['Rank']

apple     2
orange    1
grape     3
Name: Rank, dtype: int64

[47]: df.iloc[:,-1]

apple     2
orange    1
grape     3
Name: Rank, dtype: int64
```

```
[48]: df.loc[:,'Rank']

apple     2
orange    1
grape     3
Name: Rank, dtype: int64
```

Q5-b: Obtain the second row of the DataFrame `df` in at least two ways.

```
[49]: df.iloc[1,:]

Number    6
Rank      1
Name: orange, dtype: int64

[50]: df.loc['grape',:]

Number    4
Rank      3
Name: grape, dtype: int64
```

Q5-c: Obtain the rank of orange in at least four ways.

```
[51]: df['Rank']['orange']

1

[52]: df['Rank'][2]

3

[53]: df.iloc[2,1]

3

[54]: df.loc['orange','Rank']

1
```

3.3 Manipulating a DataFrame

```
[55]: df+1

        Number  Rank
apple        6     3
orange       7     2
grape        5     4

[56]: df+df

        Number  Rank
apple       10     4
orange      12     2
grape        8     6
```

```
[57]: np.exp(df)
```

```
            Number       Rank
apple   148.413159   7.389056
orange  403.428793   2.718282
grape    54.598150  20.085537
```

```
[58]: df.sort_index()
```

```
        Number  Rank
apple        5     2
grape        4     3
orange       6     1
```

```
[59]: df.sort_index(axis=1,ascending=False)
```

```
        Rank  Number
apple      2       5
orange     1       6
grape      3       4
```

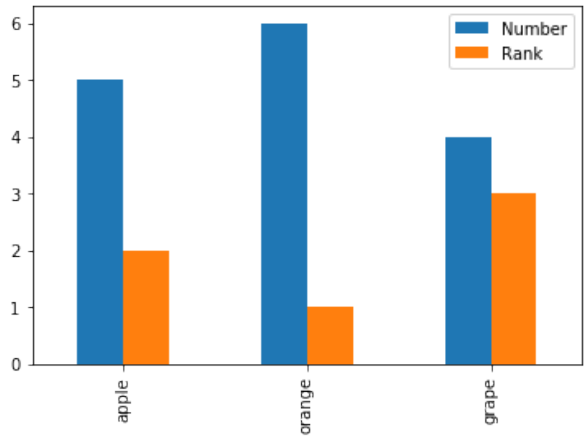
```
[60]: df.sort_values(by='Rank')
```

```
        Number  Rank
orange       6     1
apple        5     2
grape        4     3
```

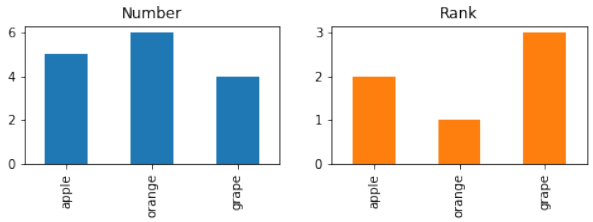
```
[61]: df.sort_values(by='orange',axis=1)
```

```
        Rank  Number
apple      2       5
orange     1       6
grape      3       4
```

```
[62]: df.plot(kind='bar')
      plt.show()
```




```
[63]: df.plot(kind='bar',subplots=True,legend=False,layout=(1,2),figsize=(8,2))
      plt.show()
```



[64]:
```
df.plot(x='Number',y='Rank',kind='scatter')
plt.show()
```