Data Wrangling Lab

Importing pandas

Getting started and checking your pandas setup

Import pandas under the alias `pd`.

```
import numpy as np
import pandas as pd
```

DataFrame basics

A few of the fundamental routines for selecting, sorting, adding and aggregating data in DataFrames

Difficulty: *easy*

Note: remember to import numpy using:

```
import numpy as np
```

Consider the following Python dictionary `data` and Python list `labels`:

```
data = {'animal': ['cat', 'cat', 'snake', 'dog', 'dog', 'cat', 'snake', 'cat', 'dog', 'dog'],
        'age': [2.5, 3, 0.5, np.nan, 5, 2, 4.5, np.nan, 7, 3],
        'visits': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
        'priority': ['yes', 'yes', 'no', 'yes', 'no', 'no', 'no', 'yes', 'no', 'no']}

labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
```

(This is just some meaningless data I made up with the theme of animals and trips to a vet.)

1. Create a DataFrame `df` from this dictionary `data` which has the index `labels`.

```
#####Your code here
data = {'animal': ['cat', 'cat', 'snake', 'dog', 'dog', 'cat', 'snake', 'cat', 'dog', 'dog
        'age': [2.5, 3, 0.5, np.nan, 5, 2, 4.5, np.nan, 7, 3],
        'visits': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
```

✓ 0s completed at 7:29 PM

```
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

df = pd.DataFrame(data, index=labels)
```

2. Display a summary of the basic information about this DataFrame and its data (*hint: there is a single method that can be called on the DataFrame*).

```
#####Your code here
type(df)
```

```
pandas.core.frame.DataFrame
```

3. Return the first 3 rows of the DataFrame `df`.

```
#####Your code here
df.head(3)
```

	animal	age	visits	priority
a	cat	2.5	1	yes
b	cat	3.0	3	yes
c	snake	0.5	2	no

4. Select just the 'animal' and 'age' columns from the DataFrame `df`.

```
#####Your code here
df[['animal', 'age']]
```

	animal	age
a	cat	2.5

a	cat	2.5
b	cat	3.0
c	snake	0.5
d	dog	NaN
e	dog	5.0
f	cat	2.0
g	snake	4.5
h	cat	NaN
i	dog	7.0
j	dog	3.0

5. Select the data in rows `[3, 4, 8]` *and* in columns `['animal', 'age']`.

```
#####Your code here
df.loc[['d', 'e', 'i'], ['animal', 'age']]
```

	animal	age
d	dog	NaN
e	dog	5.0
i	dog	7.0

6. Select only the rows where the number of visits is greater than 3.

```
#####Your code here
df[df['visits'] > 3]
```

	animal	age	visits	priority

7. Select the rows where the age is missing, i.e. it is `NaN`.

```
#####Your code here
df[df['age'].isnull()]
```

	animal	age	visits	priority
d	dog	NaN	3	yes
h	cat	NaN	1	yes

8. Select the rows where the animal is a cat *and* the age is less than 3.

```
#####Your code here
df[(df['animal'] == 'cat') & (df['age'] < 3)]
```

	animal	age	visits	priority
a	cat	2.5	1	yes
f	cat	2.0	3	no

9. Select the rows the age is between 2 and 4 (inclusive).

```
#####Your code here
df[(df['age'] >= 2) & (df['age'] <= 4)]
```

	animal	age	visits	priority
a	cat	2.5	1	yes
b	cat	3.0	3	yes
f	cat	2.0	3	no
j	dog	3.0	1	no

10. Change the age in row 'f' to 1.5.

```
#####Your code here
df.loc['f', 'age'] = 1.5
```

11. Calculate the sum of all visits in `df` (i.e. find the total number of visits).

```
#####Your code here
df['visits'].sum()
```

```
19
```

12. Calculate the mean age for each different animal in `df`.

```
#####Your code here
df.groupby('animal')['age'].mean()
```

```
animal
cat      2.333333
dog      5.000000
snake    2.500000
Name: age, dtype: float64
```

13. Append a new row 'k' to `df` with your choice of values for each column. Then delete that row to return the original DataFrame.

```
#####Your code here
df.loc['k'] = ['hamster', 1.0, 2, 'no']
df = df.drop('k', axis=0)
```

14. Count the number of each type of animal in `df`.

```
#####Your code here
df['animal'].value_counts()
```

```
cat      4
dog      4
snake    2
Name: animal, dtype: int64
```

15. Sort `df` first by the values in the 'age' in *decending* order, then by the value in the 'visits' column in *ascending* order (so row `i` should be first, and row `d` should be last).

```
#####Your code here
```

```
df = df.sort_values(by=['age', 'visits'], ascending=[False, True])
df
```

	animal	age	visits	priority
i	dog	7.0	2	no
e	dog	5.0	2	no
g	snake	4.5	1	no
j	dog	3.0	1	no
b	cat	3.0	3	yes
a	cat	2.5	1	yes
f	cat	1.5	3	no
c	snake	0.5	2	no
h	cat	NaN	1	yes
d	dog	NaN	3	yes

16. For each animal type and each number of visits, find the mean age. In other words, each row is an animal, each column is a number of visits and the values are the mean ages (*hint: use a pivot table*).

```
#####Your code here
df_pivot = df.pivot_table(index='animal', columns='visits', values='age', aggfunc='mean')
df_pivot
```

visits	1	2	3
animal			
cat	2.5	NaN	2.25
dog	3.0	6.0	NaN
snake	4.5	0.5	NaN