Consider the following Python dictionary data and Python list labels:

data = {'birds': ['Cranes', 'Cranes', 'plovers', 'spoonbills', 'spoonbills', 'Cranes', 'plovers', 'Cranes', 'spoonbills', 'spoonbills'], 'age': [3.5, 4, 1.5, np.nan, 6, 3, 5.5, np.nan, 8, 4], 'visits': [2, 4, 3, 4, 3, 4, 2, 2, 3, 2], 'priority': ['yes', 'yes', 'no', 'yes', 'no', 'no', 'no', 'yes', 'no', 'no']}

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

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

```
import pandas as pd
from pandas import DataFrame, Series
import numpy as np
import matplotlib.pyplot as plt

data = {'birds': ['Cranes', 'Cranes', 'plovers', 'spoonbills', 'spoonbills', 'Cran

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

def df():
  colLength = len(data['birds'])
  columns = list()
  for i in range(0, colLength):
    row = dict()
    for key in data:
      row[key] = data[key][i]
    columns.append(row)
  d = {}
  for rowi in range(0, len(labels)):
    d[labels[rowi]] = columns[rowi]
  return d

birds = DataFrame.from_dict(df(), orient='index')
birds
```

	birds	age	visits	priority

2. Display a summary of the basic information about birds DataFrame and its data.

```
birds.describe()
```

	age	visits
count	8.000000	10.000000
mean	4.437500	2.900000
std	2.007797	0.875595
min	1.500000	2.000000
25%	3.375000	2.000000
50%	4.000000	3.000000
75%	5.625000	3.750000
max	8.000000	4.000000

**3. Print the first 2 rows of the birds dataframe **

```
birds.head(2)
```

	birds	age	visits	priority
a	Cranes	3.5	2	yes
b	Cranes	4.0	4	yes

4. Print all the rows with only 'birds' and 'age' columns from the dataframe

```
twoCol = birds[['birds', 'age']]
twoCol
```

	birds	age
a	Cranes	3.5
b	Cranes	4.0

5. select [2, 3, 7] rows and in columns ['birds', 'age', 'visits']

d	spoonbills	NaN

```
threeCol = birds[['birds', 'age', 'visits']]
threeCol.index = range(len(threeCol))
threeColdf = dict()
for (index, row) in threeCol.iterrows():
  if index == 2 or index == 3 or index == 7:
    threeColdf[index] = row
threeColDF = DataFrame.from_dict(threeColdf, orient='index')
threeColDF
```

	birds	age	visits
2	plovers	1.5	3
3	spoonbills	NaN	4
7	Cranes	NaN	2

6. select the rows where the number of visits is less than 4

```
dfRows = birds[birds['visits'] < 4]
dfRows
```

	birds	age	visits	priority
a	Cranes	3.5	2	yes
c	plovers	1.5	3	no
e	spoonbills	6.0	3	no
g	plovers	5.5	2	no
h	Cranes	NaN	2	yes
i	spoonbills	8.0	3	no
j	spoonbills	4.0	2	no

7. select the rows with columns ['birds', 'visits'] where the age is missing i.e NaN

```
missing = birds[np.isnan(birds['age'])]
missing[['birds', 'visits']]
```

	birds	visits
d	spoonbills	4

8. Select the rows where the birds is a Cranes and the age is less than 4

```
birdCranes = birds.loc[(birds['birds'] == 'Cranes') & (birds['age'] < 4)]
birdCranes
```

	birds	age	visits	priority
a	Cranes	3.5	2	yes
f	Cranes	3.0	4	no

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

```
ageBetween = birds.loc[(birds['age'] >= 2) & (birds['age'] <= 4)]
ageBetween
```

	birds	age	visits	priority
a	Cranes	3.5	2	yes
b	Cranes	4.0	4	yes
f	Cranes	3.0	4	no
j	spoonbills	4.0	2	no

10. Find the total number of visits of the bird Cranes

```
cranes = birds.loc[birds['birds'] == 'Cranes']
totalCranesVisit = cranes['visits'].sum()
totalCranesVisit
```

```
12
```

11. Calculate the mean age for each different birds in dataframe.

```
meanAge = birds.groupby('birds').mean().reset_index()
meanAge[['birds', 'age']]
```

	birds	age
0	Cranes	3.5
1	plovers	3.5
2	spoonbills	6.0

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

```
new_birds = birds.append(DataFrame.from_dict({'k': {'birds': 'parrot', 'age':6.0,
new_birds = new_birds.drop('k')
new_birds
```

	birds	age	visits	priority
a	Cranes	3.5	2	yes
b	Cranes	4.0	4	yes
c	plovers	1.5	3	no
d	spoonbills	NaN	4	yes
e	spoonbills	6.0	3	no
f	Cranes	3.0	4	no
g	plovers	5.5	2	no
h	Cranes	NaN	2	yes
i	spoonbills	8.0	3	no
j	spoonbills	4.0	2	no

13. Find the number of each type of birds in dataframe (Counts)

```
typeOfBirds = birds.groupby('birds').size()
typeOfBirds
```

```
birds
Cranes        4
plovers       2
spoonbills    4
dtype: int64
```

14. Sort dataframe (birds) first by the values in the 'age' in decending order, then by the value in the 'visits' column in ascending order.

```
descendingage = birds.sort_values(by='age', ascending=False)
descendingage.sort_values(by='visits')
```

	birds	age	visits	priority
g	plovers	5.5	2	no
j	spoonbills	4.0	2	no
a	Cranes	3.5	2	yes
h	Cranes	NaN	2	yes
i	spoonbills	8.0	3	no
e	spoonbills	6.0	3	no

15. Replace the priority column values with'yes' should be 1 and 'no' should be 0

```
birds['priority'] = [1 if prior == 'yes' else 0 for prior in birds['priority']]
birds
```

	birds	age	visits	priority
a	Cranes	3.5	2	1
b	Cranes	4.0	4	1
c	plovers	1.5	3	0
d	spoonbills	NaN	4	1
e	spoonbills	6.0	3	0
f	Cranes	3.0	4	0
g	plovers	5.5	2	0
h	Cranes	NaN	2	1
i	spoonbills	8.0	3	0
j	spoonbills	4.0	2	0

16. In the 'birds' column, change the 'Cranes' entries to 'trumpeters'.

```
birds['birds'] = birds['birds'].replace({'Cranes': 'trumpeters'})
birds
```

	birds	age	visits	priority
a	trumpeters	3.5	2	1
b	trumpeters	4.0	4	1
c	plovers	1.5	3	0
d	spoonbills	NaN	4	1
e	spoonbills	6.0	3	0
f	trumpeters	3.0	4	0
g	plovers	5.5	2	0
h	trumpeters	NaN	2	1
i	spoonbills	8.0	3	0
j	spoonbills	4.0	2	0