Pandas basic

傻瓜也會的pandas基礎操作

In [ ]:

```
```

Store your data using a dataframe

In [1]:

```
import pandas as pd
```

In [ ]:

```
```

In [14]:

```
df = pd.DataFrame(
    [
        ['Bob', 68],
        ['Jessica', 55],
        ['Mary', 77],
        ['John', 78],
        ['Mel', 73],
    ],
    columns=['name', 'age']
)
```

In [15]:

```
df
```

Out[15]:

	name	age
0	Bob	68
1	Jessica	55
2	Mary	77
3	John	78
4	Mel	73

In [ ]:

```
```

In [ ]:

```
```

In [18]:

```
# list of tuple
data = [('Bob', 68), ('Jessica', 55), ('Mary', 77), ('John', 78), ('Mel', 73)]
```

In [25]:

```
data
```

Out[25]:

```
[['Bob', 68], ['Jessica', 55], ['Mary', 77], ['John', 78], ['Mel', 73]]
```

In [21]:

```
df = pd.DataFrame(data,columns=['name', 'age'])
```

In [22]:

```
df
```

Out[22]:

	name	age
0	Bob	68
1	Jessica	55
2	Mary	77
3	John	78
4	Mel	73

In [28]:

```
# list of list
data = [['Bob', 68], ['Jessica', 55], ['Mary', 77], ['John', 78], ['Mel', 73]]
```

In [29]:

```
data
```

Out[29]:

```
[['Bob', 68], ['Jessica', 55], ['Mary', 77], ['John', 78], ['Mel', 73]]
```

In [ ]:

```
```

In [30]:

```
df = pd.DataFrame(data,columns=['name', 'age'])
```

In [31]:

```
df
```

Out[31]:

	name	age
0	Bob	68
1	Jessica	55
2	Mary	77
3	John	78
4	Mel	73

In [ ]:

```

```

In [ ]:

```

```

In [7]:

```
# zip two lists
names = ['Bob','Jessica','Mary','John','Mel']
ages = [68, 55, 77, 78, 73]
```

In [8]:

```
data = list(zip(names,ages))
data
```

Out[8]:

```
[('Bob', 68), ('Jessica', 55), ('Mary', 77), ('John', 78), ('Mel', 73)]
```

In [9]:

```
data
```

Out[9]:

```
[('Bob', 68), ('Jessica', 55), ('Mary', 77), ('John', 78), ('Mel', 73)]
```

In [ ]:

```

```

In [10]:

```
# transform list into dataframe
df = pd.DataFrame(data = data, columns=['name', 'age'])
df
```

Out[10]:

	name	age
0	Bob	68
1	Jessica	55
2	Mary	77
3	John	78
4	Mel	73

In [ ]:

```

```

In [ ]:

```

```

寫到CSV檔案

寫檔案時的參數: index索引可以不要 header欄位名稱可以不要

存檔:

- index = False
- header = False

讀檔案:

- header = None

In [34]:

```
# save dataframe into csv file without row number
df.to_csv('mydata_age.csv',index=False)
```

In [11]:

```
df.to_csv('mydata_age.csv', index=False, header=True)
```

```
%ls
```

In [ ]:

```
```

pandas read csv from file

讀CSV

In [38]:

```
import pandas as pd
```

In [39]:

```
df = pd.read_csv('mydata_age.csv')
```

In [40]:

```
df
```

Out[40]:

	name	age
0	Bob	68
1	Jessica	55
2	Mary	77
3	John	78
4	Mel	73

In [ ]:

```
```

寫到檔案 欄位名稱不要

In [45]:

```
df.to_csv('mydata_age2.csv',index=False,header=False)
```

In [48]:

```
df2 = pd.read_csv('mydata_age2.csv')
df2
```

Out[48]:

	Bob	68
0	Jessica	55
1	Mary	77
2	John	78
3	Mel	73

再讀一次 不讀欄位名稱

In [50]:

```
df2 = pd.read_csv('mydata_age2.csv', header=None)
df2
```

Out[50]:

	0	1
0	Bob	68
1	Jessica	55
2	Mary	77
3	John	78
4	Mel	73

讀出檔案，同時給欄位名稱

In [52]:

```
df = pd.read_csv('mydata_age2.csv', names=['name','age'])
df
```

Out[52]:

	name	age
0	Bob	68
1	Jessica	55
2	Mary	77
3	John	78
4	Mel	73

In [ ]:

```
```

In [ ]:

```
```

shape of a dataframe

In [57]:

```
df.head()
```

Out[57]:

	name	age
0	Bob	68
1	Jessica	55
2	Mary	77
3	John	78
4	Mel	73

In [60]:

```
df.head(2)
```

Out[60]:

	name	age
0	Bob	68
1	Jessica	55

In [61]:

```
df.tail(2)
```

Out[61]:

	name	age
3	John	78
4	Mel	73

In [ ]:

```
```

In [53]:

```
df.shape
```

Out[53]:

```
(5, 2)
```

In [54]:

```
len(df)
```

Out[54]:

```
5
```

In [55]:

```
df.size
```

Out[55]:

```
10
```

In [62]:

```
# get array values
df.values
```

Out[62]:

```
array([['Bob', 68],
       ['Jessica', 55],
       ['Mary', 77],
       ['John', 78],
       ['Mel', 73]], dtype=object)
```

In [ ]:

```

```

In [76]:

```
df.dtypes
```

Out[76]:

```
department     object
class          object
num1          float64
num2          float64
dtype: object
```

In [ ]:

```

```

Select data from column(s)

取某一個欄位

In [21]:

```
df['age']
```

Out[21]:

```
0    68
1    55
2    77
3    78
4    73
Name: age, dtype: int64
```

In [22]:

```
df['name']
```

Out[22]:

```
0        Bob
1    Jessica
2       Mary
3       John
4        Mel
Name: name, dtype: object
```

In [23]:

```
df.age
```

Out[23]:

```
0    68
1    55
2    77
3    78
4    73
Name: age, dtype: int64
```

In [24]:

```
# Note: it is pandas Series format
type(df.age)
```

Out[24]:

```
pandas.core.series.Series
```

In [25]:

```
type(df['age'])
```

Out[25]:

```
pandas.core.series.Series
```

In [ ]:

```

```

In [26]:

```
# Conver to list format
df.age.to_list()
```

Out[26]:

```
[68, 55, 77, 78, 73]
```

In [27]:

```
list(df.age)
```

Out[27]:

```
[68, 55, 77, 78, 73]
```

In [ ]:

```

```

Slice: column wise subset of dataframe

slice切片

In [28]:

```
df[ ['name'] ]
```

Out[28]:

	name
0	Bob
1	Jessica
2	Mary
3	John
4	Mel

In [29]:

```
df[ ['age','name'] ]
```

Out[29]:

	age	name
0	68	Bob
1	55	Jessica
2	77	Mary
3	78	John
4	73	Mel

In [ ]:

```
```

In [ ]:

```
```

Select row(s)

.loc .iloc 的用法

```
.loc selects data only by labels
.iloc selects data only by integer location
```

```
To print a specific row we have couple of pandas method

loc - It only get label i.e column name or Features
iloc - Here i stands for integer, actually row number

How to use for specific row

loc

df.loc[row,column]
For first row and all column

df.loc[0,:]

For first row and some specific column
df.loc[0,'column_name']


iloc

For first row and all column
df.iloc[0,:]

For first row and some specific column i.e first three cols
df.iloc[0,0:3]
```

In [30]:

```
df
```

Out[30]:

	name	age
0	Bob	68
1	Jessica	55
2	Mary	77
3	John	78
4	Mel	73

In [ ]:

```

```

loc(索引名稱) 選取紀錄使用索引名稱

In [31]:

```
# 輸入單獨的索引名
# 選取單一筆紀錄
df.loc[0]
```

Out[31]:

```
name    Bob
age      68
Name: 0, dtype: object
```

In [32]:

```
type(df.loc[0])
```

Out[32]:

```
pandas.core.series.Series
```

輸入**slice**

In [33]:

```
# 連續的多筆紀錄
df.loc[0:2]
```

Out[33]:

	name	age
0	Bob	68
1	Jessica	55
2	Mary	77

In [ ]:

```

```

In [34]:

```
# 挑選多筆紀錄
df.loc[[0,2]]
```

Out[34]:

	name	age
0	Bob	68
2	Mary	77

In [ ]:

```

```

[row, column] 輸入行與列

In [35]:

```
df.loc[0,'name']
```

Out[35]:

```
'Bob'
```

In [36]:

```
# this is the same
df.loc[0]['name']
```

Out[36]:

```
'Bob'
```

In [37]:

```
# this is the same
df.loc[0][0]
```

Out[37]:

```
'Bob'
```

In [ ]:

```

```

[row slice, column slice]

In [38]:

```
df.loc[[0],['name']]
```

Out[38]:

	name
0	Bob

In [ ]:

```

```

In [39]:

```
# 選取 row, column
df.loc[0:2,['name']]
```

Out[39]:

	name
0	Bob
1	Jessica
2	Mary

In [40]:

```
# 選取 row, column
df.loc[[0,2],['name']]
```

Out[40]:

	name
0	Bob
2	Mary

In [41]:

```
df.loc[0:2,['name','age']]
```

Out[41]:

	name	age
0	Bob	68
1	Jessica	55
2	Mary	77

In [42]:

```
df.loc[0:2,:] # ":" indicate all of them
```

Out[42]:

	name	age
0	Bob	68
1	Jessica	55
2	Mary	77

In [ ]:

```

```

In [43]:

```
# One of the row or column is a slice
```

In [44]:

```
df.loc[0,['name']]
```

Out[44]:

```
name    Bob
Name: 0, dtype: object
```

In [45]:

```
# 選取 row, column
df.loc[0:2,'name']
```

Out[45]:

```
0        Bob
1    Jessica
2       Mary
Name: name, dtype: object
```

In [ ]:

```
```

In [ ]:

```
```

iloc操作類似，row, column只能使用integer，不能用名稱操作

In [46]:

```
df.iloc[0]
```

Out[46]:

```
name    Bob
age      68
Name: 0, dtype: object
```

In [47]:

```
df.iloc[0,0]
```

Out[47]:

```
'Bob'
```

In [48]:

```
# this line is illegal.
# df.iloc[0,'name']
```

In [49]:

```
#
df.iloc[0,[1]]
```

Out[49]:

```
age    68
Name: 0, dtype: object
```

In [ ]:

```
```

In [50]:

```
df.iloc[[0,2],[0,1]]
```

Out[50]:

	name	age
0	Bob	68
2	Mary	77

In [ ]:

```
```

In [51]:

```
df.iloc[0:2,0:1]  #只能使用integer
```

Out[51]:

	name
0	Bob
1	Jessica

In [ ]:

```
```

dataframe with our own index

In [52]:

```
df = pd.DataFrame(
    [
    ['row0',1,2,3,4,5],
    ['row1',11,12,13,14,15],
    ['row2',21,22,23,24,25],
    ['row3',31,32,33,34,35],
    ['row4',41,42,43,44,45],
    ['row5',51,52,53,54,55]
    ],
    columns=['row_idx', 'b', 'c', 'd', 'e','f']
)
```

In [53]:

```
df
```

Out[53]:

	row_idx	b	c	d	e	f
0	row0	1	2	3	4	5
1	row1	11	12	13	14	15
2	row2	21	22	23	24	25
3	row3	31	32	33	34	35
4	row4	41	42	43	44	45
5	row5	51	52	53	54	55

In [54]:

```
# set index with row_idx
df.set_index("row_idx" , inplace=True)
```

In [55]:

```
df
```

Out[55]:

row_idx	b	c	d	e	f
row0	1	2	3	4	5
row1	11	12	13	14	15
row2	21	22	23	24	25
row3	31	32	33	34	35
row4	41	42	43	44	45
row5	51	52	53	54	55

In [ ]:

```

```

In [56]:

```
df.loc['row3', 'b']
```

Out[56]:

```
31
```

In [ ]:

```

```

```
This line is illegal.

df.iloc['row3', 'b']

ValueError: Location based indexing can only have [integer, integer slice (START point is
INCLUDED, END point is EXCLUDED), listlike of integers, boolean array] types
```

In [57]:

```
# This line is illegal.
# df.iloc['row3', 'b']
```

In [ ]:

```

```

In [58]:

```
# 讓index回復成原本的樣子
df.reset_index(inplace=True)
```

In [59]:

```
df
```

Out[59]:

	row_idx	b	c	d	e	f
0	row0	1	2	3	4	5
1	row1	11	12	13	14	15
2	row2	21	22	23	24	25
3	row3	31	32	33	34	35
4	row4	41	42	43	44	45
5	row5	51	52	53	54	55

In [ ]:

```

```

In [ ]:

```

```

求最大值，哪個人年紀最大?

In [77]:

```
import pandas as pd
```

In [78]:

```
df = pd.read_csv('mydata_age.csv')
```

In [79]:

```
df
```

Out[79]:

	name	age
0	Bob	68
1	Jessica	55
2	Mary	77
3	John	78
4	Mel	73

In [83]:

```
df.age.max()
```

Out[83]:

78

In [ ]:

```

```

排序

In [ ]:

```
# Method 1:
#sorted = df.sort(['age'], ascending=False)
df_sorted = df.sort_values(by='age', ascending=False)
```

In [81]:

```
df_sorted
```

Out[81]:

	name	age
3	John	78
2	Mary	77
4	Mel	73
0	Bob	68
1	Jessica	55

In [82]:

```
df_sorted.head(1)
```

Out[82]:

	name	age
3	John	78

In [ ]:

```

```

Select and Query

In [ ]:

```

```

In [ ]:

```

```

In [95]:

```
df[df['name'] == 'John']
```

Out[95]:

	name	age
3	John	78

In [99]:

```
df[df.name == 'John']
```

Out[99]:

	name	age
3	John	78

In [98]:

```
df.query('name == "John"')
```

Out[98]:

	name	age
3	John	78

In [ ]:

```
```

In [ ]:

```
```

In [85]:

```
df[df['age'] > 70]
```

Out[85]:

	name	age
2	Mary	77
3	John	78
4	Mel	73

In [86]:

```
df[df.age > 70]
```

Out[86]:

	name	age
2	Mary	77
3	John	78
4	Mel	73

In [87]:

```
df.query('age > 70')
```

Out[87]:

	name	age
2	Mary	77
3	John	78
4	Mel	73

In [ ]:

```
```

In [89]:

```
df[(df.age > 60) & (df.age < 75)]
```

Out[89]:

	name	age
0	Bob	68
4	Mel	73

In [ ]:

```

```

In [90]:

```
# df[df.age > 60 & df.age < 75] # lack of round brackets
```

In [92]:

```
df.query('60 < age < 75')
```

Out[92]:

	name	age
0	Bob	68
4	Mel	73

In [ ]:

```

```

In [ ]:

```

```

In [105]:

```
minAge = 70
```

In [106]:

```
# @ indicates a variable name
df.query('age > @minAge')
```

Out[106]:

	name	age
2	Mary	77
3	John	78
4	Mel	73

In [ ]:

```
```

In [ ]:

```
```

Query: another example

In [60]:

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

In [61]:

```
df = pd.DataFrame(np.random.randn(10, 3), columns=['a','b','c'])
```

In [62]:

```
df
```

Out[62]:

	a	b	c
0	-1.355998	0.778179	-0.407345
1	-1.268881	-1.783726	1.633037
2	-0.141692	-0.077769	-2.046616
3	1.886637	0.280846	0.105663
4	1.241718	1.536480	-1.705679
5	-0.437428	-0.088150	-0.179450
6	-0.265546	-1.134072	0.073807
7	1.141422	-0.056067	-1.310841
8	-0.459843	1.619861	0.594093
9	2.649519	0.284311	-0.850801

In [63]:

```
df.query('a > 0').query('0 < b < 2')
```

Out[63]:

	a	b	c
3	1.886637	0.280846	0.105663
4	1.241718	1.536480	-1.705679
9	2.649519	0.284311	-0.850801

In [64]:

```
df.query('a > 0 and 0 < b < 2')
```

Out[64]:

	a	b	c
3	1.886637	0.280846	0.105663
4	1.241718	1.536480	-1.705679
9	2.649519	0.284311	-0.850801

In [65]:

```
df.query('a > b')
```

Out[65]:

	a	b	c
1	-1.268881	-1.783726	1.633037
3	1.886637	0.280846	0.105663
6	-0.265546	-1.134072	0.073807
7	1.141422	-0.056067	-1.310841
9	2.649519	0.284311	-0.850801

In [ ]:

```

```

In [ ]:

```

```

In [ ]:

```

```

Groupby運算

In [4]:

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

In [67]:

```
df = pd.DataFrame({'department' : ['EE', 'IM', 'EE', 'IM',
                     'EE', 'IM', 'EE', 'EE'],
                   'class' : ['one', 'one', 'two', 'three',
                         'two', 'two', 'one', 'three'],
                   'num1' : np.random.randn(8),
                   'num2' : np.random.randn(8)})
```

In [68]:

```
df
```

Out[68]:

	department	class	num1	num2
0	EE	one	0.381410	0.364945
1	IM	one	-0.260750	0.071927
2	EE	two	-1.602220	-0.216720
3	IM	three	0.371315	1.506681
4	EE	two	-0.429950	1.041738
5	IM	two	-0.818868	1.144542
6	EE	one	-0.423023	0.491168
7	EE	three	1.553664	-1.068004

In [71]:

```
df.groupby('department').count()
```

Out[71]:

department	class	num1	num2
EE	5	5	5
IM	3	3	3

In [72]:

```
df.groupby('department').sum()
```

Out[72]:

department	num1	num2
EE	-0.520119	0.613127
IM	-0.708303	2.723150

In [73]:

```
df.groupby(['department','class']).count()
```

Out[73]:

		num1	num2
department	**class**		
EE	**one**	2	2
	three	1	1
	two	2	2
IM	**one**	1	1
	three	1	1
	two	1	1

In [74]:

```
df.groupby(['department','class']).sum()
```

Out[74]:

		num1	num2
department	**class**		
EE	**one**	-0.041613	0.856113
	three	1.553664	-1.068004
	two	-2.032170	0.825018
IM	**one**	-0.260750	0.071927
	three	0.371315	1.506681
	two	-0.818868	1.144542

In [ ]:

```

```

In [ ]:

```

```

Chart簡單繪圖

In [100]:

```
import pandas as pd
```

In [101]:

```
df = pd.read_csv('mydata_age.csv')
```

In [102]:

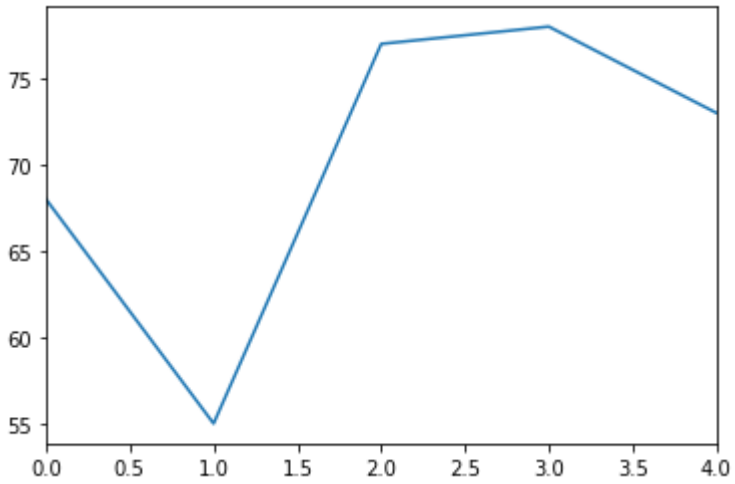
```
# inline function
%matplotlib inline
```

In [103]:

```
df['age'].plot()
```

Out[103]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x21bf8e98588>
```

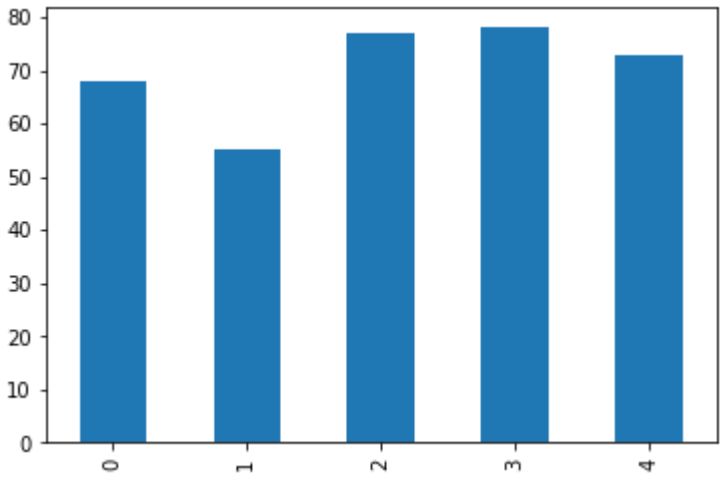


In [104]:

```
df['age'].plot(kind='bar')
```

Out[104]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x21bf8a22ba8>
```



In [ ]: