# 第 2~3 章

判断: FFFTF　　FTTTF　　FFFFT　　TFTFT　　FTTFT　　FFTTF　　FFFFF

选择: AABBD　BAABC　CBDDA　DBCDB　DBCCA　DCDAC DCDBA　CBB

简答题：

1. [name for names in all_data　for name in names　if len(set(name))<len(name)]

2. {1: 'red', 2: 'blue', 3: 'green', 4: 'white'}

3. a=1, b=2, c=3

    a=4,b=5, c=6

    a=7,b=8, c=9

4. strings = ['foo', 'card', 'bar', 'aaaa', 'abab']

    strings.sort(key=lambda x: len(set(list(x))))

    strings

5. [x for x in range(2,101) if not [y for y in range(2,x) if x % y == 0]]

6. [[ i*j for j in range(5)] for i in range(4)]

7. [name for names in all_data for name in names if name.count('e')>=2]

8. a = {1:1, 2:2, 3:3}

    print(",".join('%s' %id for id in a.keys()))

9. [x for tup1 in some_tuples for tup2 in tup1 for x in tup2 if (x.count('M') + x.count('m')) >

    0 ]

10.　　5　　　[1,2,2,3,3,3,4,4,5]

11. all = {key: ", ".join([sex[key], age[key], job[key]]) for key in sex if key in age and key in

job}

# 第 4 章

判断: FTTTT   TTFFF   TTFFF   FFTFF  TFTFF  FFFTF  TTFFF  FF

选择：CABCD   CACBA   DAAAD   ACBBD

CCBCB   CCCBA   BDAAB   ACBCD

AA

简答题：

1. [ 2   6   5 13]

2. data[index!='red' ]

3. array([[6, 6, 6, 6],

    [6, 6, 6, 6],

    [6, 6, 6, 6]])

4. result = [(x if c else y) for x, y, c in zip(xarr, yarr, cond)]

    Result（这种方法比较慢，而且对多维数组不起作用）

    Or：

    result= np.where(cond, xarr, yarr)（这种方法更好）

    result

5. import numpy as np

    Z = np.ones((10,10))

    Z[1:-1,1:-1] = 0

print(Z)

6. np.where(np.random.randint(0,2,1000)>0,1,-1).cumsum()

7. import numpy as np

   a=np.random.randint(1,11,size=[10])

   a.sort()

   max=a.max()

   min=a.min()

   var=a.var()

8. 1)整数部分：arrInt = np.modf(arr)[1],arrInt

   小数部分：arrFloat = np.modf(arr)[0],arrFloat

   2) arr = np.where(arr>0, arrFloat, np.abs(arrInt))

9. 1）arr[arr>0].cumsum()

   2）np.where(arr>0,arr,0).cumsum(0)

10. [[0 1 2]

   [3 4 5]

   [6 7 8]]

11. array([[ 9, 11, 10],

       [ 1,  3,  2],

       [ 5,  7,  6]])

12. [ True False]

13. array([[[0., 1., 2., 3.],

       [0., 1., 2., 3.]],

[[0., 1., 2., 3.],

[0., 1., 2., 3.]]])

14. [[ 0   2]

[ 9 11]]

15. array([[1., 1., 1.],          [4., 5., 6.]])

16. [[[ 0 1 2 3] [ 4 5 6 7]]

[[ 8 9 10 11] [12 13 14 15]]]


[[ 0 1 2 3]

[ 4 5 6 7]

[ 8 9 10 11]

[12 13 14 15]]

17. array([[[42, 42, 42],

[42, 42, 42]],

[[ 7,   8,   9],

[10, 11, 12]]])

18. a = np.eye(5) ; np.where(a == 1, '#', '$')

19. array([[1, 2, 3], [4, 5, 6]])

array([[[ 7,   8,   9], [10, 11, 12]]])

array([[ 7,   8,   9], [10, 11, 12]])

20. array([1, 2, 3, 4, 5, 6])

21. [[ 0  1  2  3  4  5]

    [18 19 20 21 22 23]]

# 第 5 章

判断：TTTTF    FTFTT    TTTFF    FFFTT    FFFTF    FFTTF

选择：ADCDD    C（ABCD）DB A    CDDDB    BBCBA    CBBCB    BCCB(CD)

简答题：

1.  1，frame1 = pd.DataFrame([[1,2,3],[4,5,6],[7,8,9]],index = ['a', 'b', 'c'],columns = ['d','f','g'])

    2，frame = pd.DataFrame({'d':{'a':1, 'b':4, 'c':7},'f':{'a':2, 'b':5, 'c':8},'g':{'a':3, 'b':6, 'c':9}})

2.

| | year | team | score |
|---|---|---|---|
| 3 | 2003.0 | Alex | 7.0 |
| 4 | 2004.0 | Thomson | 10.0 |
| 5 | NaN | NaN | NaN |
| 1 | 2001.0 | Alice | 8.0 |
| 2 | 2002.0 | Bob | 9.0 |

3.  ser1 = pd.Series(list('abcdefghijklmnopqrstuvwxyz'))

    [pd.Index(ser1).get_loc(i) for i in list('adhz')]

    [list(ser1.values).index(i) for i in list('adhz')]

4.

|   | b | a |
|---|---|---|
| 2 | -3 | 0 |
| 0 | 4 | 0 |
| 3 | 2 | 1 |
| 1 | 7 | 1 |

5. 
a    12.0

b    NaN

c    NaN

d    11.0

e    NaN

6.

|        | b   | d   | e   |
|--------|-----|-----|-----|
| Utah   | 0.0 | 0.0 | 0.0 |
| Ohio   | 3.0 | 3.0 | 3.0 |
| Texas  | 6.0 | 6.0 | 6.0 |
| Oregon | 9.0 | 9.0 | 9.0 |

7. s.drop_duplicates().sort_values(ascending=False)

8. 
0    NaN

1    NaN

2    4.0

3    4.0

4    5.0

dtype: float64

9.        d      c      b      a

| | | | | |
|---|---|---|---|---|
| three | 0 | 3 | 2 | 1 |
| one | 4 | 7 | 6 | 5 |

10.

| | b | a |
|---|---|---|
| 2 | -3 | 0 |
| 0 | 4 | 0 |
| 3 | 2 | 1 |
| 1 | 7 | 1 |

11.
```
import pandas as pd

data ={'Chinese':[66,95,95,90,80,80],'English':[65,85,92,88,90,90],'Math':[None,98,96,77,90,90]}

df=pd.DataFrame(data,index=['张飞','关羽','赵云','黄忠','典韦','典韦'],columns=['Chinese','Math','English'])

df = df.drop_duplicates()

df.rename(columns={'Chinese':'语文','English':'英语','Math':'数学'},inplace = True)

def total_score(df):

df['总分'] = df['语文']+df['数学']+df['英语']

return df

df['数学'].fillna(df['数学'].mean())

df.sort_values(['总分'],ascending=False,inplace=True)

print(df)
```

12.

| | Nevada | Ohio |
|---|---|---|
| 2000 | NaN | 1.5 |
| 2001 | 2.4 | 1.7 |
| 2002 | 2.9 | 3.6 |

13.

| 0 | 5.0 |
|---|---|
| 1 | 3.0 |
| 2 | 6.5 |
| 3 | 1.0 |
| 4 | 3.0 |
| 5 | 6.5 |
| 6 | 3.0 |

14.
```
dti = pd.date_range(start='2018-01-01',end='2018-12-31',freq='D')
s = pd.Series(np.random.rand(len(dti)),index=dti)
s[s.index.weekday == 2].sum()
s.resample('M').mean()
```

15.

| | year | team | score |
|---|---|---|---|
| 3 | 2003.0 | Alex | 7.0 |
| 4 | 2004.0 | Thomson | 10.0 |
| 5 | NaN | NaN | NaN |
| 1 | 2001.0 | Alice | 8.0 |
| 2 | 2002.0 | Bob | 9.0 |

# 第 6 章

判断∷FTTTF　TFFTT　TFFFT　TFTTF　TTTFF　TFTTF　TTTF

选择∷BBCDD　DDBAA　CDBAB　BB（第17题作废）DBC　DA

简答题：

1. data.to_csv('data.csv', sep='|')

   data.to_excel('data.xls')

2. import pandas as pd

   import requests

   url = 'http://api.github.com/repos/test'

   response = requests.get(url)

   data = response.json()

   a = pd.DataFrame(data, columns = ['number','title','labels','state'])

3. |       | row 1 | row 2 |
   |-------|-------|-------|
   | col 1 | a     | c     |
   | col 2 | b     | d     |

4. import sqlalchemy as sqla

   db=sqla.create_engine('sqlite:///mydata.sqlite')

   Pd.read_sql('select * from test',db)

5. chunker=pd.read_csv('ex6.csv',chunksize=1000)

   tot=pd.Series([])

   for piece in chunker:

```
        tot=tot.add(piece['key'].value_counts(),fill_value=0)
```

6. pd.read_csv('examples/ex2.csv', names=['a', 'b', 'c', 'd', 'index'], index_col='index')

7. import pandas as pd

   frame = pd.read_csv('C:/test/ex1.csv', names=['a','b','c'], skiprows=[0,1])

8. import pandas as pd

   data.to_csv('C:/data.csv', sep='、', na_rep='null')

9. pd.read_csv('example.csv',skiprows=[0,2])

10. pd.read_csv('example.csv',index_col=['key1','key2'])

11.

```
2000-01-01, 0
2000-01-02, 1
2000-01-03, 2
2000-01-04, 3
2000-01-05, 4
2000-01-06, 5
2000-01-07, 6
```

12. chunker=pd.read_csv('ex6.csv',chunksize=1000)

    tot=pd.Series([])

    for piece in chunker:

    tot=tot.add(piece['key'].value_counts(),fill_value=0)

13. lines = [['a', 'b', 'c'], ['1', '2', '3'], ['1', '2', '3']]

    header, values = lines[0], lines[1:]

    data_dict = {h: v for h, v in zip(header, zip(*values))}

    data_dict

14. import pandas as pd

import numpy as np

frame = pd.read_csv('../examples/ex1.csv')

frame.to_pickle('../examples/frame_pickle')

# 第 7 章

判断：FTFTF FTTTF FTTFF FFFTT TTFFT FFTT

选择：DABDC DCCBB DBADA DCABA ACA（BD）D AADBA DA

简答题：

1.    one    two    three

a    1.0    NaN    1.0

c    NaN    NaN    5.0

d    4.0    9.0    7.0

e    5.0    10.0    9.0

2.    k1    k2    k3

0    1    2    3

1    2    3    1

3    3    3    2

3. [(10, 18], (18, 25], (10, 18], (25, 30], (25, 30], NaN]

4. ages = [17, 22, 25, 27, 21, 23, 37, 31, 61, 45, 41, 132]

bins = [18, 25, 35, 60, 100]

```python
cats = pd.cut(ages, bins)

pd.value_counts(cats)
```

5. 
```python
import re

text = "foo    bar\t baz   \tqux,ok"

','.join(re.split('\s+', text)).split(',')
```

6. 
```python
sampler=np.random.permutation(5)

df.take(sampler)

df.sample(n=3)
```

7. 

```
0    1.0

1    NaN

2    2.0

3    NaN

4    0.0

5    3.0
```

8. `data.rename(index=str.title,columns=str.upper,inplace=True)`

9. `'::'.join([x.strip() for x in s.split(',')])`

10. 
```python
import re

re.findall(r'[1-9]\d{5}',text)
```

11. 
```python
import pandas as pd

from numpy import nan as NA

data=pd.Series([1,NA,1.0,NA,1])
```

```
data=data.dropna()
```

结果：

0 1.0

2 1.0

4 1.0

12. df.fillna({1:df[1].mean(), 2:df[2].mean()},inplace = True)

13. data['animal']=data['food'].map(lambda x: meat_to_animal[x.lower()])

    data['animal']=data['food'].apply(lambda x: meat_to_animal[x.lower()])

    data['animal']=data['food'].transform(lambda x: meat_to_animal[x.lower()])

14. data['b']=data['a'].map(lambda x : 'positive' if x > 0 else 'negative')

15. array([-1,  0,  2,  3, -1], dtype=int8)

16.

|   | 0   | 1   | 2   |
|---|-----|-----|-----|
| 0 | 1   | 6.5 | 3   |
| 1 | 1   | NaN | NaN |
| 2 | NaN | NaN | NaN |
| 3 | NaN | 6.5 | 3   |

17.
```
data=np.random.ranf(1000)

category=(0.0,0.08,0.24,0.48,1.0)

labels=('一等奖','二等奖','三等奖','参与奖')

result=pd.cut(data,category,labels=labels)

result=pd.value_counts(result)

result
```

18. 0    1.0

1    NaN

2    2.0

3    NaN

4    NaN

5    3.0

19. 0    1.0

1    2.0

2    2.0

3    3.0

# 第 8 章

判断：TFTTF　FTTFT　TTFTT FFTTF　TTTFT　TFTFF　　TFFT

选择：BAAAD　CABBA　CABDC　CBDDB　　BDBAB　CD

简答题：

1.

|   | key | variable | value |
|---|-----|----------|-------|
| **0** | a | A | 1 |
| **1** | b | A | 2 |
| **2** | a | B | 3 |
| **3** | b | B | 4 |

2.

|   | C | A | B |
|---|---|---|---|
| **0** | a | 1 | 3 |
| **1** | b | 2 | 4 |

3.

```
a  0      0
   1      0
b  0      1
   1      1
dtype: int64
```

4.

| | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| k1 | | | | | |
| a | 2.5 | 3.5 | 4.5 | 5.5 | 6.5 |
| b | 10.0 | 11.0 | 12.0 | 13.0 | 14.0 |

5.

| | key | data1 | data2 |
|---|-----|-------|-------|
| 0 | a | 1 | 2 |
| 1 | b | 3 | 4 |
| 2 | c | 5 | 6 |

|   | 3 | d | 7 | 8 |

6.

|   | lkey | data1 | rkey | data2 |
|---|------|-------|------|-------|
| 0 | b | 0 | b | 1 |
| 1 | b | 1 | b | 1 |
| 2 | b | 6 | b | 1 |
| 3 | a | 2 | a | 0 |
| 4 | a | 4 | a | 0 |
| 5 | a | 5 | a | 0 |

7.

| level1 | level | | | |
|--------|-------|-----|-------|------|
|   | one | two | three | four |
| a | 0 | 1 | 5.0 | 6.0 |
| b | 2 | 3 | NaN | NaN |
| c | 4 | 5 | 7.0 | 8.0 |

8. pd.melt(df, ['key'])

| key | variable | value |
|-----|----------|-------|
| 0 | foo | A | 1 |
| 1 | bar | A | 3 |
| 2 | foo | B | 5 |
| 3 | bar | B | 6 |

9.    key    value    group_val

| | | | |
|---|---|---|---|
| 0 | a | 0 | 3.5 |
| 2 | a | 2 | 3.5 |
| 3 | a | 3 | 3.5 |
| 1 | b | 1 | 7.0 |
| 4 | b | 4 | 7.0 |
| 5 | c | 5 | NaN |

10.

| | 0 | 1 |
|---|---|---|
| a | 0 | 0 |
| b | 1 | 1 |

11.

| | 1 | 2 | 3 |
|---|---|---|---|
| a | 0.0 | 1.0 | 2.0 |
| b | 3.0 | 4.0 | NaN |
| c | 5.0 | 6.0 | NaN |

12.

| upper | level1 | | level2 | |
|---|---|---|---|---|
| lower | one | two | three | four |
| A | 0 | 1 | 5.0 | 6.0 |
| B | 2 | 3 | NaN | NaN |
| C | 4 | 5 | 7.0 | 8.0 |

13. pd.pivot_table(df,index=["年代","产地"],values=["评分"])

14. pd.pivot_table(df,index=["产地"],values=["投票人数","评分"],aggfunc={"投票人数":np.sum,"评分":np.mean})

15. ①pd.merge(left2, right2, how='inner', left_index=True, right_index=True)

②left2.join(right2, how='inner')

16.

| | a | b |
|---|---|---|
| **0** | 1.0 | NaN |
| **1** | 5.0 | 2.0 |
| **2** | 5.0 | 2.0 |
| **3** | NaN | 6.0 |

17. data.swaplevel(1,0).unstack()

18. 4

19.

| | | a | b |
|---|---|---|---|
| **c** | **d** | | |
| **one** | 0 | 0 | 7 |
| | 1 | 1 | 6 |
| | 2 | 2 | 5 |
| **two** | 0 | 3 | 4 |
| | 1 | 4 | 3 |
| | 2 | 5 | 2 |
| | 3 | 6 | 1 |

20. a    1

c    6

d    7

21.

```
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11],
       [ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])
```

22.
```
one  a    0.0
     b    1.0
     c    2.0
     d    3.0
two  c    4.0
     d    5.0
     e    6.0
dtype: float64
```

| | key1 | key2_left | data1 | key2_right | data2 |
|---|---|---|---|---|---|
| **0** | b | one | 0 | two | 1 |
| **1** | b | two | 1 | two | 1 |

23.
```
评分A   一    4
       二    6
       三    8
评分B   一    5
       二    7
       三    9
dtype: int32
```

24.

# 第 9 章

判断：TTTTT TFTFT TTTFF FTTFF TTFFF TFTFF F

选择：ABACB DCDCB BDACA ACDBA BCCBB ABB

简答题：

1. 参考答案：绘制一个图片，输出服从 N~(0, 1)分布下的 50 次随机值的累加值，使用实

   线，颜色为红色，标记点为红点。

2. df = pd.DataFrame(np.random.rand(10, 4), columns=['a', 'b', 'c', 'd'])

   df.plot.bar(stacked=True)

3. import numpy as np

   ```
   import matplotlib.pyplot as plt

   t = np.arange(0., 6.,0.001)

   plt.plot(t, t**2, 'b')

   plt.show()
   ```

4. import matplotlib.pyplot as plt

   ```
   import numpy as np

   fig, axes = plt.subplots(2, 2, sharex=True, sharey=True)

   for i in range(2):

       for j in range(2):

           axes[i, j].hist(np.random.randn(50),bins=5, color='b')

   plt.subplots_adjust(wspace=0, hspace=0)
   ```

5. import matplotlib.pyplot as plt

   ```
   import numpy as np

   fig = plt.figure()

   ax = fig.add_subplot(1, 1, 1)

   ax.plot(np.random.randn(1000).cumsum())

   ticks = ax.set_xticks([0, 250, 500, 750, 1000])

   labels = ax.set_xticklabels(['one', 'two', 'three', 'four', 'five'])
   ```

6. 答：import matplotlib.pyplot as plt

   ```
   import numpy as np

   fig,axes=plt.subplots(2,2)
   ```

```python
axes[0,0].plot(np.random.randn(100),'k--')
```

或

```python
import matplotlib.pyplot as plt

import numpy as np

fig=plt.figure()

ax1=fig.add_subplot(2,2,1)

ax1.plot(np.random.randn(100),'k--')
```

7. 
```python
import matplotlib.pyplot as plt

import numpy as np

import pandas as pd

s=pd.Series(np.random.randn(1000).cumsum(),index=np.arange(0,1000))

fig=s.plot()

fig.set_xticks([0,250,500,750,1000])

fig.set_xticklabels(['one','two','three','four','five'],rotation=30)

fig.set_xlabel('stage')

fig.set_title('random')
```

8. 
```python
sns.catplot(data=tips[tips['tip_pct']<1],x='day',y='tip_pct',hue='time',col='smoker',kind='bar')
```

9. 
```python
for a,b in zip(x[::10],y[::10]):

    plt.text(a,b+20,b,ha='center',fontsize=10)

plt.annotate("2012年达到最大值",xy=(2012,data[2012]),xytext=(2020,2030),arrowprops=dict(facecolor="black"))
```

```python
    plt.text(1980,1000,"电影数量开始增长")
```

10. 
```python
plt.hist(df["评分"],bins=20,edgecolor='k',alpha=0.5)

plt.show()
```

11. 
```python
fig,axes=plt.subplots(2,2)

plt.subplots_adjust(wspace=0.5,hspace=0.5)

for i in range(2):

    axes[0,i].set_title('data%d'%(i+1))

    axes[1,i].set_title('data%d'%(i+3))

data1 = pd.Series([23,41,56,34,52,78],index=['A','B','C','D','E','F'])

data2 = pd.Series([21,42,56,21,12,34],index=['A','B','C','D','E','F'])

data3 = pd.Series([67,35,24,75,23,11],index=['A','B','C','D','E','F'])

data4 = pd.Series([45,64,23,11,24,67],index=['A','B','C','D','E','F'])

data1.plot.bar(ax=axes[0,0])

data2.plot.bar(ax=axes[0,1])

data3.plot.barh(ax=axes[1,0])

data4.plot.barh(ax=axes[1,1])
```

12. 
```python
fig,axes = plt.subplots()

axes.scatter(Ser,Ser.index)

Labels = axes.set_xticks([0,1,2,3,4])

ticks = axes.set_yticks([0,1,2,3,4])

for i in Ser:

    text = axes.text(i,i,'(' + str(i) + ',' + str(i) +')')
```

13. 
```python
x = np.linspace(-5,5,100)

y = np.where(x<0,0,x)

plt.plot(x,y)
```

14. 
```python
x=np.linspace(0,2*np.pi,50)

y=np.sin(x)

plt.plot(x,y)

plt.show()
```

15. 
```python
import numpy as np

import matplotlib.pyplot as plt

import seaborn


x = np.random.randn(10000)

plt.hist(x, 25, normed=1, facecolor='b', edgecolor = 'black')

seaborn.kdeplot(x)

plt.show()
```

16. 
```python
import numpy as np

import matplotlib.pyplot as plt

t = np.arange(0., 6.,0.001)

plt.plot(t, t**2, 'b')

plt.show()
```

17. 
```python
import matplotlib.pyplot as plt

fig = plt.figuru()
```

```
ax = fig.add_subplot(1,1,1)

rect = plt.Rectangle((0.2,0.75), 0.4, 0.15, color = 'k', alpha = 0.3)

ax.add_patch(rect)
```

# 第 10 章

判断：FTTFT　TTTTT　TTFFF　FTFTT　TFFTT　FFTFT　　TF

选择：DBADA　　CDBBD　　ABAAC　　ACBC

简答题：

1.　　　data1　data2

　　key

　　a　　3.0　　6.0

　　b　　1.5　　3.0

　　c　　3.0　　6.0

2.　(1) grouped = data.groupby('group').mean()

　　　print(grouped)

　　(2) tempdata1 = {'average': (data['part 1'] + data['part 2'] + data['part 3']) / 3}

　　　data2 = pd.DataFrame(tempdata1)

　　　print(data2.sort_values(by='average'))

　　(3) data['total'] = data['part 1'] + data['part 2'] + data['part 3']

　　　tmp = data.sort_values(by = 'total', ascending = False)

　　　print(tmp.iloc[0])
```

3.

|   | a | b | c | d | e |
|---|---|---|---|---|---|
| 3 | 25 | 28 | 31 | 34 | 37 |
| 5 | 5 | 6 | 7 | 8 | 9 |
| 6 | 20 | 21 | 22 | 23 | 24 |

4.
```
grouped = df.groupby('category')

get_wavg = lambda g: np.average(g['data'], weights=g['weights'])

grouped.apply(get_wavg)
```

5.
```
fill_values={'East':0.5,'West':-1}

fill_func=lambda g:g.fillna(fill_values[g.name])

data.groupby(group_key).apply(fill_func)
```

6.
```
df.groupby(['key1','key2'])['data1'].m
```
或
```
df['data1'].groupby([df['key1'],df['key2']]).mean()
```

7.

| Handedness | Left-handed | Right-handed | All |
|---|---|---|---|
| Nationality | | | |
| Japan | 2 | 3 | 5 |
| USA | 1 | 4 | 5 |
| All | 3 | 7 | 10 |

8.
```
df.groupby('key1').agg(lambda x: x.mean() - x.median())
```

9.
```
data=pd.DataFrame({'key1':['a','a','b','b','a'],'key2':['one','two','one','two','one'],'data1':[12,11,46,74,23],'data2':[15,11,24,7,35]})

grouped = data.groupby(['key1','key2'])

grouped.agg({'data1':'max','data2':'min'})
```

10.
```
def get_stats(group):
```

return

{'min':group.min(),'max':group.max(),'count':group.count(),'mean':group.mean()}

| key2 | one | two |
|------|-----|-----|
| key1 | | |
| a | 3 | 2 |
| b | 3 | 4 |

11.

| | data1 | data2 |
|---|-------|-------|
| a | 7 | 25 |
| b | 3 | 15 |

12.

| | 2 | 4 |
|---|---|---|
| 0 | 4 | 8 |
| 1 | 3 | 12 |

13.

14.

# 第 11 章

判断：TFTTT TFTTF TFTTF FTFFF FTTTF TFTFT

选择：CABBD ACBAB DCBBD DAACD BADAC DDDAC CCADA

简答题：

1.  2011-01-02    0

    2011-01-05    1

    2011-01-07    2

    2011-01-08    3

2011-01-10    4

2011-01-12    5


2.    2019-01-27    3

2019-02-03    4

2019-02-10    5

2019-02-17    6

2019-02-24    7

2019-03-03    8


3.    2019-01    11

2019-01    12

2019-01    13

2019-01    14

2019-02    15

2019-02    16

2019-02    17

2019-02    18

2019-03    19

2019-03    20

4.    ts.resample('5min',closed='right',label='right').sum()

5.    frame.resample('D').ffill()

6. `close_px['AAPL'].rolling(100).mean().plot()`

7. 　　 time_zones=[line['tz'] for line in records if 'tz' in line]

　　 from collections import Counter

　　 Counter(time_zones)

　 或

　　 import pandas as pd

　　 time_zones=[line['tz'] for line in records if 'tz' in line]

　　 s=pd.Series(time_zones)

　　 s.value_counts()

8.

2019-08-31　　0

2019-09-30　　1

2019-10-31　　2

2019-11-30　　3

2019-12-31　　4

9. （1）Period('2007-06-30', 'D')

　　（2）Period('2006-07', 'M')

10. from datetime import datetime

　　 now = datetime.now()

　　 now.strftime('%Y.%m.%d-%H:%M:%S')

11. grouped = df.groupby(level=0)

　　 grouped.mean()

12. ser_period = ser.to_period('M')

    ser_period.groupby(ser_period.index).sum()

13. 2019-11-30    0

    2019-12-31    1

    2020-01-31    2

    2020-02-29    3

    2020-03-31    4

    Freq: M, dtype: int32

14. from datetime import date,timedelta

    def getDate(year,weeks,weekday):

        start = date(year,1,1)

        for i in range(7):

            if start.isoweekday() == weekday:

                break

            start = start +timedelta(days=1)

        return start + timedelta(weeks=weeks-1)

    print(getDate(2020,2,5))

15. 2000-03-31    0

    2000-04-30    1

    2000-05-31    2

    2000-06-30    3

    Freq: M, dtype: int32

16. 2019-01-27     3

2019-02-03     4

2019-02-10     5

2019-02-17     6

2019-02-24     7

2019-03-03     8

17. 2019-10-31      4

2019-11-30     11

2019-12-31     17

# 第 12 章

判断：FTTFT  TFTTT  FTFTF  TTTFF  FTTTT  FTFTF  F

选择：CCDAB  CBCAA  AAABB  CCACC  C

简答题：

1.  g = data.groupby('key').value

print(g.transform(lambda x: x * x))

2.  [foo, bar, baz, foo, foo, bar]

Categories (3, object): [foo < bar < baz]

3.              num

2019-01-31    0

2019-03-31    3

2019-05-31    7

4. time_key=pd.Grouper(freq='5min')

   resampled=(df.set_index('time').groupby(['key',time_key]).sum())

   或

   time_key=pd.TimeGrouper('5min')

   resampled=(df.set_index('time').groupby(['key',time_key]).sum())

5. bins=pd.qcut(draws,4,labels=['Q1','Q2','Q3','Q4'])

   bins=pd.Series(bins,name='quartile')

   results=(pd.Series(draws)

           .groupby(bins)

           .agg(['count','mean'])

           .reset_index())

6. result=(df.pipe(f,arg1=v1)

           .pipe(g,v2,arg3=v3)

           .pipe(h,arg4=v4))

7. data.drop('category',axis=1).join(pd.get_dummies(data['category'],prefix='category'))

8. results=(pd.Series(draws).groupby(bins).agg(['count','min','max']).reset_index())

9. 0    地铁

   0    地铁

   1    高铁

0    地铁

0    地铁

1    高铁

10. pd.Categorical.from_codes([0, 0, 0, 1, 1, 1, 2, 2, 2], categories)

11. time_key = pd.TimeGrouper('15min')

resampled = df.set_index('time').groupby(['fruit',time_key]).sum()

12. df.groupby('key').value.transform(lambda x: x.mean())

13. 0    1.5

1    2.5

2    3.5

3    1.5

4    2.5

5    3.5

14. d    2

c    2

b    2

a    2

dtype: int64

15. df['gender'] = pd.Categorical.from_codes(

[1 if x == 'Female' else 0 for x in df['gender']],

['Female', 'Male'])

16.            num

2019-01-31    0

2019-03-31    3

2019-05-31    7

17.

```
0      4.0
1      4.0
2      4.0
3      3.0
4      3.0
5      3.0
6      2.0
7      2.0
8      2.0
9      1.0
10     1.0
11     1.0
Name: value, dtype: float64
```