期中考试

November 6, 2019

选择题、简答略

1. 有如下列表，按照要求实现功能。（15 分）List = [‘Jack’,‘Tom’,‘Helen’]

（1）计算该列表的长度

[35]:
```
List = ['Jack','Tom','Helen']
print('序列的长度为：',len(List))
```

序列的长度为： 3

（2）列表中追加元素 “Boby”，并输出添加后的列表

[36]:
```
List.append('Boby')
print('增加后的内容是：',List)
```

增加后的内容是： ['Jack', 'Tom', 'Helen', 'Boby']

（3）在列表的第 3 个位置插入元素 “Liuli”，并输出添加后的列表

[37]:
```
List.insert(2,'Liuli')
print('增加后的内容是：',List)
```

增加后的内容是： ['Jack', 'Tom', 'Liuli', 'Helen', 'Boby']

（4）删除列表中的第 2 个元素，并输出删除后的列表

[38]:
```
List.pop(1)
# del List[1]
print('删除后的内容是：',List)
```

删除后的内容是： ['Jack', 'Liuli', 'Helen', 'Boby']

（5）将列表中的所有元素反转，并输出反转后的结果

[40]:
```
List.reverse()
print('反转后的内容是：',List)
```

反转后的内容是： ['Boby', 'Helen', 'Liuli', 'Jack']

2. 根据要求建立数组，并进行相应操作

（1）建立一个 4×6 的二维数组 arr1，其中的值为 1～99 之间（含 99）的随机数。

[42]:
```
import numpy as np
```

[47]:
```
arr1=np.random.randint(1,100,24).reshape(4,6)
print(arr1)
```

```
[[74 73 42 12 91 87]
 [68  9 82 35 53 12]
 [95 71 84  1 14 45]
 [79 24 25 79 33  5]]
```

(2) 在 1-50 之间建立一个等差序列的数组 arr2，长度为 24，建立完成后将其转成 4×6 的二维数组。

[49]:
```
arr2 = np.linspace(1,50,24).reshape(4,6)
print(arr2)
```

```
[[ 1.          3.13043478  5.26086957  7.39130435  9.52173913 11.65217391]
 [13.7826087  15.91304348 18.04347826 20.17391304 22.30434783 24.43478261]
 [26.56521739 28.69565217 30.82608696 32.95652174 35.08695652 37.2173913 ]
 [39.34782609 41.47826087 43.60869565 45.73913043 47.86956522 50.        ]]
```

(3) 对 arr1、arr2 进行水平和垂直堆叠，对堆叠后的结果进行输出。

[52]:
```
print(np.vstack((arr1,arr2)))
```

```
[[74.         73.         42.         12.         91.         87.        ]
 [68.          9.         82.         35.         53.         12.        ]
 [95.         71.         84.          1.         14.         45.        ]
 [79.         24.         25.         79.         33.          5.        ]
 [ 1.          3.13043478  5.26086957  7.39130435  9.52173913 11.65217391]
 [13.7826087  15.91304348 18.04347826 20.17391304 22.30434783 24.43478261]
 [26.56521739 28.69565217 30.82608696 32.95652174 35.08695652 37.2173913 ]
 [39.34782609 41.47826087 43.60869565 45.73913043 47.86956522 50.        ]]
```

[53]:
```
print(np.hstack((arr1,arr2)))
```

```
[[74.         73.         42.         12.         91.         87.
   1.          3.13043478  5.26086957  7.39130435  9.52173913 11.65217391]
 [68.          9.         82.         35.         53.         12.
  13.7826087  15.91304348 18.04347826 20.17391304 22.30434783 24.43478261]
 [95.         71.         84.          1.         14.         45.
  26.56521739 28.69565217 30.82608696 32.95652174 35.08695652 37.2173913 ]
 [79.         24.         25.         79.         33.          5.
  39.34782609 41.47826087 43.60869565 45.73913043 47.86956522 50.        ]]
```

(4) 对 arr1 和 arr2 进行加法操作，对生成的新数组进行乘积操作。

[54]:
```
print(arr1+arr2)
```

```
[[ 75.          76.13043478  47.26086957  19.39130435 100.52173913
   98.65217391]
```

```
 [ 81.7826087   24.91304348 100.04347826  55.17391304  75.30434783
   36.43478261]
 [121.56521739  99.69565217 114.82608696  33.95652174  49.08695652
   82.2173913 ]
 [118.34782609  65.47826087  68.60869565 124.73913043  80.86956522
   55.        ]]
```

[55]:
```
print(arr1*arr2)
```

```
[[  74.          228.52173913  220.95652174   88.69565217  866.47826087
  1013.73913043]
 [ 937.2173913   143.2173913  1479.56521739  706.08695652 1182.13043478
   293.2173913 ]
 [2523.69565217 2037.39130435 2589.39130435   32.95652174  491.2173913
  1674.7826087 ]
 [3108.47826087  995.47826087 1090.2173913  3613.39130435 1579.69565217
   250.        ]]
```

(5) 对 arr1 数组进行横向排序，对 arr2 数组进行纵向排序。

[62]:
```
arr1.sort(axis=1)
print(arr1)
```

```
[[12 42 73 74 87 91]
 [ 9 12 35 53 68 82]
 [ 1 14 45 71 84 95]
 [ 5 24 25 33 79 79]]
```

[63]:
```
arr2.sort(axis=0)
print(arr2)
```

```
[[ 1.          3.13043478  5.26086957  7.39130435  9.52173913 11.65217391]
 [13.7826087  15.91304348 18.04347826 20.17391304 22.30434783 24.43478261]
 [26.56521739 28.69565217 30.82608696 32.95652174 35.08695652 37.2173913 ]
 [39.34782609 41.47826087 43.60869565 45.73913043 47.86956522 50.        ]]
```

3. 请模拟读取文件 “price.csv”（5 分），并对该文件内容进行以下操作：

(1) 对文件内容进行排序（3 分）

[65]:
```
data = np.loadtxt('数据集/模块 3/price.csv')
data.sort()
print(data)
```

```
[   3.81    3.81    3.81    3.81    3.81    3.81    3.81    3.81    3.81
    3.81    3.81    3.81    3.81    3.81    3.81    3.81    3.81    3.81
    3.81    5.      5.      5.      5.      5.      5.      5.      5.
    5.      5.      5.      5.      5.      5.      5.      5.      5.
    5.      5.      5.      5.      6.1     6.1     6.1     6.1     6.1
    7.62    7.62    7.62    7.62    8.      8.      8.      8.      8.
```

```
  8.      8.      8.     10.     10.     10.     10.     15.     16.99
 17.3    17.99   19.7    20.     23.3    23.6    25.9    26.39   26.6
 27.     27.     27.6    29.3    29.4    29.99   29.99   30.     30.
 30.     30.6    30.8    31.29   31.6    31.8    32.5    33.1    33.4
 34.6    35.     36.     36.     36.8    37.4    38.6    38.7    38.7
 38.7    38.7    38.7    38.7    38.7    38.7    38.7    38.7    38.7
 38.7    38.7    38.7    38.7    38.7    39.     39.2    39.2    39.8
 39.99   40.     40.1    40.9    41.06   41.2    43.5    43.5    45.
 46.5    46.5    46.6    46.6    46.6    46.6    46.6    46.6    46.6
 46.6    46.6    47.1    47.4    47.4    47.4    47.6    48.     49.
 49.8    50.2    50.4    51.     51.3    51.3    52.2    52.6    53.04
 53.7    53.7    54.     54.     54.4    54.4    54.5    54.5    54.5
 54.5    54.5    54.5    54.5    54.5    54.5    54.5    54.5    54.5
 56.3    58.4    58.5    59.2    59.2    59.3    60.     60.5    61.6
 61.6    62.4    62.4    62.4    62.4    62.4    62.4    62.4    62.4
 62.4    62.4    62.4    62.4    62.4    62.4    62.4    62.4    62.4
 63.     63.     63.2    63.2    63.9    64.3    64.7    67.1    67.4
 69.1    70.3    70.3    70.3    70.8    70.9    71.1    71.8    72.6
 72.7    75.     77.4    78.     78.2    79.     79.     85.     85.3
 86.9    88.     88.     89.     89.6    94.     94.     94.8    95.
 98.     98.7   101.1   101.1   101.1   101.1   101.1   101.1   101.1
101.9   109.    109.    109.    109.8   109.8   118.5   118.5   118.5
126.4   132.7   144.71  150.    150.1   150.1   155.9   158.    164.38
171.    178.    178.    188.    188.    189.6   189.6   189.6   189.6
189.6   189.6   200.    209.5   211.7   219.6   237.    237.    237.
258.8   271.    278.    290.7   300.    300.    300.    314.4   349.
398.    405.    417.6   454.    500.    505.    540.    566.7   620.
734.   1198.   1278.2 ]
```

(2) 去除读取对象中重复内容（3 分）

[67]:
```
np.unique(data)
```

[67]:
```
array([   3.81,    5.  ,    6.1 ,    7.62,    8.  ,   10.  ,   15.  ,
         16.99,   17.3 ,   17.99,   19.7 ,   20.  ,   23.3 ,   23.6 ,
         25.9 ,   26.39,   26.6 ,   27.  ,   27.6 ,   29.3 ,   29.4 ,
         29.99,   30.  ,   30.6 ,   30.8 ,   31.29,   31.6 ,   31.8 ,
         32.5 ,   33.1 ,   33.4 ,   34.6 ,   35.  ,   36.  ,   36.8 ,
         37.4 ,   38.6 ,   38.7 ,   39.  ,   39.2 ,   39.8 ,   39.99,
         40.  ,   40.1 ,   40.9 ,   41.06,   41.2 ,   43.5 ,   45.  ,
         46.5 ,   46.6 ,   47.1 ,   47.4 ,   47.6 ,   48.  ,   49.  ,
         49.8 ,   50.2 ,   50.4 ,   51.  ,   51.3 ,   52.2 ,   52.6 ,
         53.04,   53.7 ,   54.  ,   54.4 ,   54.5 ,   56.3 ,   58.4 ,
         58.5 ,   59.2 ,   59.3 ,   60.  ,   60.5 ,   61.6 ,   62.4 ,
         63.  ,   63.2 ,   63.9 ,   64.3 ,   64.7 ,   67.1 ,   67.4 ,
         69.1 ,   70.3 ,   70.8 ,   70.9 ,   71.1 ,   71.8 ,   72.6 ,
         72.7 ,   75.  ,   77.4 ,   78.  ,   78.2 ,   79.  ,   85.  ,
         85.3 ,   86.9 ,   88.  ,   89.  ,   89.6 ,   94.  ,   94.8 ,
```

```
 95.   ,   98.   ,   98.7 ,  101.1 ,  101.9 ,  109.   ,  109.8 ,
118.5 ,  126.4 ,  132.7 ,  144.71,  150.   ,  150.1 ,  155.9 ,
158.   ,  164.38,  171.   ,  178.   ,  188.   ,  189.6 ,  200.   ,
209.5 ,  211.7 ,  219.6 ,  237.   ,  258.8 ,  271.   ,  278.   ,
290.7 ,  300.   ,  314.4 ,  349.   ,  398.   ,  405.   ,  417.6 ,
454.   ,  500.   ,  505.   ,  540.   ,  566.7 ,  620.   ,  734.   ,
1198.   , 1278.2 ])
```

(3) 对对象中的数值进行求和与均值（3 分）

[68]:
```
data.sum()
```

[68]:
```
26325.69
```

[69]:
```
data.mean()
```

[69]:
```
87.75229999999999
```

(4) 对对象中的数值求标准差与方差（3 分）

[71]:
```
data.std()
```

[71]:
```
140.61673711799034
```

[72]:
```
data.var()
```

[72]:
```
19773.066757710003
```

(5) 对对象中的数值求最大值与最小值（3 分）

[73]:
```
data.max()
```

[73]:
```
1278.2
```

[74]:
```
data.min()
```

[74]:
```
3.81
```

[ ]:
```

```