**Report 2**

**Part 1: Course exercises**

1.1:array

1.2:二

1.3:数据类型 dtype

1.4:广播机制

1.5:高级索引

2.1:F

2.2:T

2.3:F

2.4:T

2.5:F

3.1:A

3.2:D

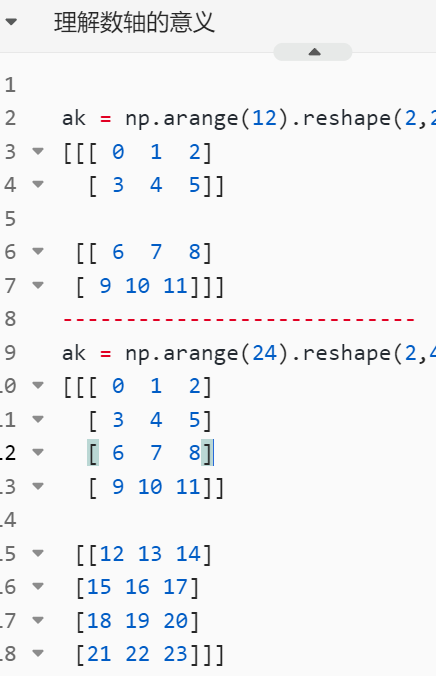
3.3:A

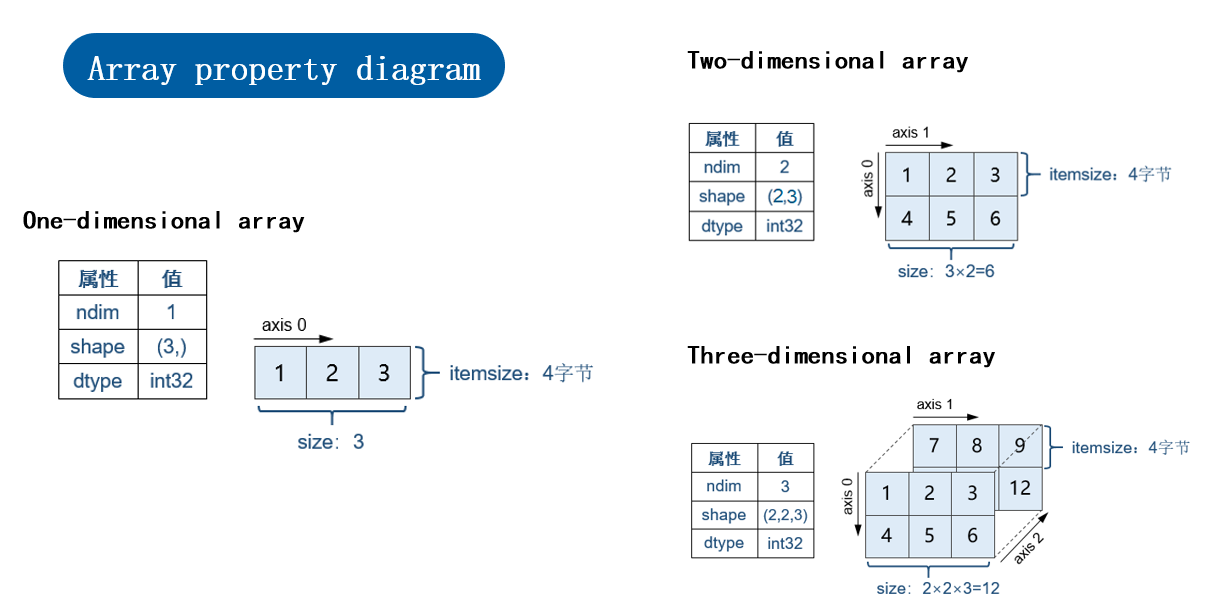
3.4:C

3.5:A

4.1:

轴是从外到内编号的，从 0 开始,对于一个 n 维数组，轴的范围是 0 到 n-1,每个轴对应数组的一个维度。先输出并且填充数值最大的那个轴，然后递减，打印的最内侧也是最大值的轴。



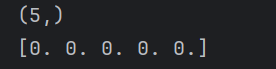


4.2满足其下条件之一即可：（1）、维度星等（2）、有一方维度为1

5.1:

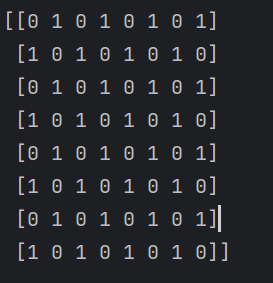
import numpy as np  
a = np.zeros(5)  
# a.shape(5,)  
print(a.shape)  
print(a)

# #运行结果：



5.2:

import numpy as np  
  
  
a = np.ndarray((8,8),int)  
for i in range(8) :  
 for j in range(8) :  
 if (i + j) % 2 == 0 :  
 a[i][j] = 0  
 else :  
 a[i][j] = 1  
print(a)

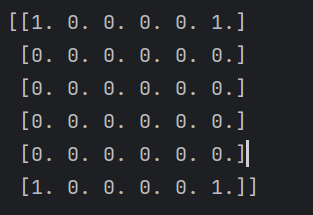


**Part 2:**

**Complete the following NumPy practices and screenshot your solutions and results.**

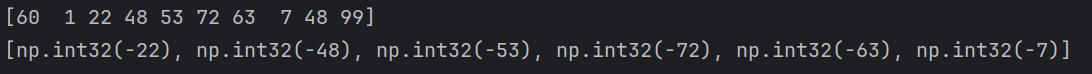
1. 用1来填充一个6\*6的全0数组的四个边界.

a = np.zeros((6,6))  
a[0][0] = 1  
a[5][0] = 1  
a[0][5] = 1  
a[5][5] = 1  
print(a)



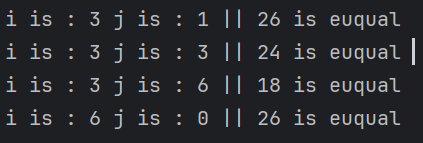
2. 随机生成一个长度为10的一维数组,求其索引为2到7的元素的相反数.

a = np.random.randint(1,100,10)  
print(a)  
ans = []  
for i in range(2,8) :  
 ans.append(-a[i])  
print(ans)



3. 随机生成两个8\*8的整形数组(取值范围10到30),求两个数组的公共元素.

a = np.random.randint(10,30,(8,8))  
b = np.random.randint(10,30,(8,8))  
  
  
for i in range(8):  
 for j in range(8):  
 if a[i][j] == b[i][j] :  
 print("i is : " + str(i)+ " j is : " + str(j) +" || "+ str(a[i][j]) + " is euqual ")



4. 获取你出生年份月份的所有日期.

list1 = []  
  
for i in range(1,32) :  
 print("8." + str(i))  
 list1.append("8."+str(i))

输出结果：

['8.1', '8.2', '8.3', '8.4', '8.5', '8.6', '8.7', '8.8', '8.9', '8.10', '8.11', '8.12', '8.13', '8.14', '8.15', '8.16', '8.17', '8.18', '8.19', '8.20', '8.21', '8.22', '8.23', '8.24', '8.25', '8.26', '8.27', '8.28', '8.29', '8.30', '8.31']

5. 用3种不同的方法提取一个随机6\*6浮点型数组中的整数部分.

arr = np.random.rand(6,6) \* 10  
print(arr)  
floor\_arr = np.floor(arr)  
print(floor\_arr)  
intarr = arr.astype(int)  
print(intarr)  
truncarr = np.trunc(arr)  
print(truncarr)

输出结果：

[[3.14692875 4.55499407 9.4008997 4.2900644 9.94294326 5.69106583]

[4.75672532 8.87076702 1.09177718 8.29766977 5.66910017 2.23527035]

[0.08948811 1.41266707 8.95162944 6.73541327 0.04909051 8.25102279]

[1.46725046 1.76384913 1.89789656 5.43362693 8.06226991 8.02544147]

[8.37402987 8.65199002 4.54034658 6.65392473 2.40179912 9.7931015 ]

[6.87967108 0.39934442 6.06130757 0.71152006 4.45022891 2.61222225]]

[[3. 4. 9. 4. 9. 5.]

[4. 8. 1. 8. 5. 2.]

[0. 1. 8. 6. 0. 8.]

[1. 1. 1. 5. 8. 8.]

[8. 8. 4. 6. 2. 9.]

[6. 0. 6. 0. 4. 2.]]

[[3 4 9 4 9 5]

[4 8 1 8 5 2]

[0 1 8 6 0 8]

[1 1 1 5 8 8]

[8 8 4 6 2 9]

[6 0 6 0 4 2]]

[[3. 4. 9. 4. 9. 5.]

[4. 8. 1. 8. 5. 2.]

[0. 1. 8. 6. 0. 8.]

[1. 1. 1. 5. 8. 8.]

[8. 8. 4. 6. 2. 9.]

[6. 0. 6. 0. 4. 2.]]

6. 创建一个长度为20的一维整形数组,取值范围是1到20,但不包括所有奇数.

a = np.random.randint(1,10,20) \* 2   
print(a)

输出结果：

[12 10 6 18 16 12 8 6 14 10 16 18 2 14 10 18 16 10 4 10]

7. 创建一个8\*8的随机数组,将该数组的最大值替换成1,最小值替换成0.

a = np.random.rand(8,8)  
  
a[np.unravel\_index(np.argmax(a),a.shape)] = 1  
a[np.unravel\_index(np.argmin(a),a.shape)] = 0  
print(np.unravel\_index(np.argmax(a),a.shape))  
print(np.unravel\_index(np.argmin(a),a.shape))  
  
print(a)

输出结果：

(np.int64(7), np.int64(3))

(np.int64(7), np.int64(6))

[[0.06325742 0.44312757 0.22039403 0.13562151 0.79579712 0.56258947

0.30461733 0.80921939]

[0.74541263 0.93529589 0.88107183 0.01384047 0.38390667 0.86197899

0.42120707 0.22664119]

[0.45812754 0.39533767 0.07393284 0.7113294 0.88835513 0.37687935

0.05498692 0.5223025 ]

[0.05695384 0.13182288 0.43899987 0.97070413 0.17622693 0.09619656

0.45640117 0.60441663]

[0.73255672 0.05186107 0.10194673 0.74571481 0.0945203 0.09995313

0.8662013 0.01318759]

[0.91808763 0.31979618 0.76764515 0.12943921 0.12625427 0.49162324

0.79662157 0.51793586]

[0.39723004 0.17838526 0.68317173 0.62502661 0.74857578 0.52864635

0.58291164 0.62529239]

[0.59041368 0.04875739 0.38613617 1. 0.03076002 0.93182736

0. 0.53191102]]

8. 创建一个6\*6的随机数组,让每一行的各个元素减去整个数组的平均值.

a = np.random.rand(6,6) \* 10  
print(a)  
print(a.mean())  
a = a - a.mean()  
print(a)

输出结果：

[[6.48660724 1.54178278 6.233258 1.05757551 1.82051191 7.63072839]

[8.48553881 1.67585185 1.64882169 1.27630674 6.02059001 3.08859597]

[2.14691529 9.06462325 1.64363202 1.91940822 3.97943007 5.88939179]

[1.17875606 2.3032877 1.60381191 7.08064539 6.35942104 8.58222955]

[6.53539889 0.84132279 2.89158384 3.74259136 5.65792639 9.62703557]

[3.65927799 4.77103172 2.28324595 8.86455573 3.5719055 1.15471355]]

4.231064180028789

[[ 2.25554306 -2.6892814 2.00219382 -3.17348867 -2.41055227 3.39966421]

[ 4.25447463 -2.55521233 -2.58224249 -2.95475744 1.78952583 -1.14246821]

[-2.08414889 4.83355907 -2.58743216 -2.31165596 -0.25163411 1.65832761]

[-3.05230812 -1.92777648 -2.62725227 2.84958121 2.12835686 4.35116537]

[ 2.30433471 -3.38974139 -1.33948034 -0.48847282 1.42686221 5.39597139]

[-0.57178619 0.53996754 -1.94781823 4.63349155 -0.65915868 -3.07635063]]

9. 创建一个最小值为10,最大值为30,差值为5的等差数列,向该数列中相邻两个元素间填充4个1.

# 创建等差数列  
array = np.arange(5, 31, 5)   
  
fill\_value = 1  
fill\_count = 4  
result = np.repeat(array, fill\_count + 1)   
result[1::fill\_count + 1] = fill\_value   
  
result = result[:-fill\_count]  
  
print("结果数组:", result)

结果数组: [10 1 1 1 1 15 1 1 1 1 20 1 1 1 1 25 1 1 1 1 30]

10. **(加分题)** 创建一个0到999的等差数列, 用两种方法求其数值第6小的元素(输出值为5).

arr = np.arange(0,1000)  
# kind 1  
cnt = 0  
for i in arr :  
 cnt += 1  
 if cnt == 6 :  
 print(i)  
  
# kind 2  
print(arr[5])

输出结果：

5

5