numpy 练习题

2151131 朱沙桐 数据科学与大数据技术

numpy 的array操作

1.导入numpy库

```
import numpy as np

print("NumPy version:", np.__version__)
```

2.建立一个一维数组 a 初始化为[4,5,6], (1)输出a 的类型（type）(2)输出a的各维度的大小（shape）(3)输出 a的第一个元素（值为4）

```
a = np.array([4, 5, 6])

# (1) 输出a的类型（type）

print("Type of a:", type(a))

# (2) 输出a的各维度的大小（shape）

print("Shape of a:", a.shape)

# (3) 输出a的第一个元素（值为4）

print("First element of a:", a[0])
```

3.建立一个二维数组 b,初始化为 [[4, 5, 6],[1, 2, 3]] (1)输出各维度的大小（shape）(2)输出 b(0,0)， b(0,1),b(1,1) 这三个元素（对应值分别为4,5,2)

```
# 建立二维数组 b

b = np.array([[4, 5, 6], [1, 2, 3]])

# (1) 输出各维度的大小（shape）

print("Shape of b:", b.shape)

# (2) 输出 b(0,0)，b(0,1),b(1,1) 这三个元素

print("Element b(0,0):", b[0, 0])

print("Element b(0,1):", b[0, 1])

print("Element b(1,1):", b[1, 1])
```

4. (1)建立一个全0矩阵 a, 大小为 3x3; 类型为整型（提示: dtype = int）(2)建立一个全1矩阵b,大小为4x5; (3)建立一个单位矩阵c ,大小为4x4; (4)生成一个随机数矩阵d,大小为 3x2.

```
# (1) 建立一个全0矩阵 a, 大小为 3x3; 类型为整型

a = np.zeros((3, 3), dtype=int)

print("(1) Matrix a (3x3) filled with zeros:\n", a)

# (2) 建立一个全1矩阵b,大小为4x5

b = np.ones((4, 5))

print("\n(2) Matrix b (4x5) filled with ones:\n", b)

# (3) 建立一个单位矩阵c ,大小为4x4

c = np.eye(4)

print("\n(3) Identity matrix c (4x4):\n", c)

# (4) 生成一个随机数矩阵d,大小为 3x2

d = np.random.random((3, 2))

print("\n(4) Random matrix d (3x2):\n", d)
```

5. 建立一个数组 a,(值为[[1, 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12]]) ,(1)打印a; (2)输出下标为(2,3),(0,0) 这两个数组元素的值

```
# 建立数组a

a = np.array([[1, 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12]])

# (1) 打印a

print("(1) Array a:\n", a)

# (2) 输出下标为(2,3),(0,0) 这两个数组元素的值

print("(2) Element at index (2,3):", a[2, 3])

print("    Element at index (0,0):", a[0, 0])
```

6.把上一题的 a数组的 0到1行 2到3列，放到b里面去，（此处不需要从新建立a,直接调用即可）(1),输出b;(2) 输出b 的（0,0）这个元素的值

```
# 切片操作获取数组a的0到1行、2到3列的子数组，并赋值给b

b = a[0:2, 2:4]

# (1) 输出b

print("(1) Array b:\n", b)

# (2) 输出b的(0,0)这个元素的值

print("(2) Element at index (0,0) of b:", b[0, 0])
```

7. 把第5题中数组a的最后两行所有元素放到 c中，（提示： a[1:2, :]）(1)输出 c ; (2) 输出 c 中第一行的最后一个元素（提示，使用 -1 表示最后一个元素）

```
# 获取数组a的最后两行所有元素，并赋值给c

c = a[1:, :]

# (1) 输出c

print("(1) Array c:\n", c)

# (2) 输出c中第一行的最后一个元素

print("(2) Last element of the first row in c:", c[0, -1])
```

8.建立数组a,初始化a为[[1, 2], [3, 4], [5, 6]]，输出 （0,0） （1,1） （2,0）这三个元素（提示： 使用 print(a[[0, 1, 2], [0, 1, 0]]))

```
# 建立数组a并初始化

a = np.array([[1, 2], [3, 4], [5, 6]])

# 输出指定位置的元素

print("Elements at specified indices:", a[[0, 1, 2], [0, 1, 0]])
```

9.建立矩阵a ,初始化为[[1, 2, 3], [4, 5, 6], [7, 8, 9], [10, 11, 12]]，输出(0,0),(1,2),(2,0),(3,1) (提示使用 b = np.array([0, 2, 0, 1])　　print(a[np.arange(4), b]))

```
# 建立矩阵a并初始化

a = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9], [10, 11, 12]])

# 定义索引数组b

b = np.array([0, 2, 0, 1])  # 这是纵坐标的索引

# 输出指定位置的元素

print("Elements at specified indices:", a[np.arange(4), b]) # arranges(4)生成[0,1,2,3]
```

10.对9 中输出的那四个元素，每个都加上10，然后重新输出矩阵a.(提示：a[np.arange(4), b] += 10)

```
# 对指定位置的元素加上10

a[np.arange(4), b] += 10

# 重新输出矩阵a

print("Matrix a after addition:", a)
```

array 的数学运算

11. 执行 x = np.array([1, 2])，然后输出 x 的数据类型

```
# 执行操作

x = np.array([1, 2])

# 输出x的数据类型

print("Data type of x:", x.dtype)
```

12.执行 x = np.array([1.0, 2.0])，然后输出 x 的数据类类型

```
# 执行操作

x = np.array([1.0, 2.0])

# 输出x的数据类型

print("Data type of x:", x.dtype)
```

13.执行 x = np.array([[1, 2], [3, 4]], dtype=np.float64) ， y = np.array([[5, 6], [7, 8]], dtype=np.float64)，然后输出 x+y ,和 np.add(x,y)

```
# 执行操作

x = np.array([[1, 2], [3, 4]], dtype=np.float64)

y = np.array([[5, 6], [7, 8]], dtype=np.float64)

# 输出 x+y

print("Sum of x and y:")

print(x + y)

# 输出 np.add(x,y)

print("\nnp.add(x,y):")

print(np.add(x, y))
```

14. 利用 13题目中的x,y 输出 x-y 和 np.subtract(x,y)

```
# 输出 x-y

print("Difference of x and y:")

print(x - y)

# 输出 np.subtract(x,y)

print("\nnp.subtract(x,y):")

print(np.subtract(x, y))
```

15. 利用13题目中的x，y 输出 x*y ,和 np.multiply(x, y) 还有 np.dot(x,y),比较差异。然后自己换一个不是方阵的试试。

```
# 输出 x*y

print("Element-wise multiplication of x and y:")

print(x * y)

# 输出 np.multiply(x, y)

print("\nnp.multiply(x, y):")

print(np.multiply(x, y))

# 输出 np.dot(x, y)

print("\nnp.dot(x, y):")
```

```
16
17 print(np.dot(x, y))
```

16. 利用13题目中的x,y,输出 x / y .(提示： 使用函数 np.divide())

```
1 # 输出 x / y
2
3 print("Element-wise division of x and y:")
4
5 print(np.divide(x, y))
```

17. 利用13题目中的x,输出 x的 开方。(提示： 使用函数 np.sqrt())

```
1 # 输出 x / y
2
3 print("Element-wise division of x and y:")
4
5 print(np.divide(x, y))
```

18.利用13题目中的x,y ,执行 print(x.dot(y)) 和 print(np.dot(x,y))

```
1  # 输出 x.dot(y)
2
3  print("Result of x.dot(y):")
4
5  print(x.dot(y))
6
7  # 输出 np.dot(x, y)
8
9  print("\nResult of np.dot(x, y):")
10
11 print(np.dot(x, y))
```

19.利用13题目中的 x,进行求和。提示：输出三种求和 (1)print(np.sum(x)): (2)print(np.sum(x, axis =0)); (3)print(np.sum(x,axis = 1))

```
1  # 输出总和
2
3  print("(1) Sum of all elements in x:")
4
5  print(np.sum(x))
6
7  # 沿着列的方向求和
8
9  print("\n(2) Sum of elements along axis 0 (column-wise sum):")
10
11 print(np.sum(x, axis=0))
12
13 # 沿着行的方向求和
14
15 print("\n(3) Sum of elements along axis 1 (row-wise sum):")
16
17 print(np.sum(x, axis=1))
```

20.利用13题目中的 x,进行求平均数（提示：输出三种平均数 (1)print(np.mean(x)) (2)print(np.mean(x,axis = 0))(3) print(np.mean(x,axis =1)))

```
# 输出总平均数

print("(1) Mean of all elements in x:")

print(np.mean(x))

# 沿着列的方向计算平均数

print("\n(2) Mean of elements along axis 0 (column-wise mean):")

print(np.mean(x, axis=0))

# 沿着行的方向计算平均数

print("\n(3) Mean of elements along axis 1 (row-wise mean):")

print(np.mean(x, axis=1))
```

21.利用13题目中的x，对x 进行矩阵转置，然后输出转置后的结果， （提示： x.T 表示对 x 的转置)

```
# 对x进行转置，并输出结果

print("Transpose of x:")

print(x.T)
```

22.利用13题目中的x,求e的指数（提示： 函数 np.exp()）

```
# 对x求e的指数，并输出结果

print("Exponential of x:")

print(np.exp(x))
```

23.利用13题目中的 x,求值最大的下标（提示(1)print(np.argmax(x)) ,(2) print(np.argmax(x, axis =0))(3)print(np.argmax(x),axis =1))

```
# 输出值最大的元素的索引（扁平化的索引）

print("(1) Index of maximum value in x (flattened index):")

print(np.argmax(x))

# 沿着列的方向输出值最大的元素的索引

print("\n(2) Indices of maximum value along axis 0 (column-wise indices):")

```

```
print(np.argmax(x, axis=0))

# 沿着行的方向输出值最大的元素的索引

print("\n(3) Indices of maximum value along axis 1 (row-wise indices):")

print(np.argmax(x, axis=1))
```

24,画图，y=x*x 其中 x = np.arange(0, 100, 0.1) （提示这里用到 matplotlib.pyplot 库)

```
import matplotlib.pyplot as plt

# 生成x的值

x = np.arange(0, 100, 0.1)

# 计算对应的y值

y = x * x

# 绘制图形

plt.plot(x, y)

plt.title('y = x*x')

plt.xlabel('x')

plt.ylabel('y')

plt.grid(True)

plt.show()
```

25.画图。画正弦函数和余弦函数， x = np.arange(0, 3 * np.pi, 0.1)(提示：这里用到 np.sin() np.cos() 函数和 matplotlib.pyplot 库)

```
# 生成x的值

x = np.arange(0, 3 * np.pi, 0.1)

# 计算正弦函数和余弦函数的值

y_sin = np.sin(x)

y_cos = np.cos(x)

# 绘制正弦函数的图形

plt.plot(x, y_sin, label='Sin')

# 绘制余弦函数的图形

```

```
plt.plot(x, y_cos, label='Cos')

plt.title('Sin and Cos Functions')

plt.xlabel('x')

plt.ylabel('y')

plt.legend()

plt.grid(True)

plt.show()
```