一.numpy 的 array 操作
1.导入 numpy 库
import numpy as np
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)
代码：

```
a=np.array([4,5,6])
print(a.dtype)#data_type
print(a.shape)
print(a[0])
```

输出：

```
int32
(3,)
4
```

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)
代码：

```
b=np.array([[4,5,6],
                [1,2,3]])
print(b.shape)
print(b[0,0],b[0,1],b[1,1])
```

输出：

```
(2, 3)
4 5 2
```

4.
(1)建立一个全 0 矩阵 a, 大小为 3×3；类型为整型(提示：dtype=int)；
(2)建立一个全 1 矩阵 b, 大小为 4×5；
(3)建立一个单位矩阵 c，大小为 4×4；
(4)生成一个随机数矩阵 d, 大小为 3×2；
代码：

```
a=np.zeros((3,3),dtype=int)
b=np.ones((4,5),dtype=int)
c=np.identity((4),dtype=int)
d=np.random.randint(1,10,dtype=int,size=(3,2))#randint 从给定的上下限范围内随机选取整数
print(a)
print(b)
print(c)
print(d)
```

输出：

```
[[0 0 0]
 [0 0 0]
```

```
  [0 0 0]]
[[1 1 1 1 1]
  [1 1 1 1 1]
  [1 1 1 1 1]
  [1 1 1 1 1]]
[[1 0 0 0]
  [0 1 0 0]
  [0 0 1 0]
  [0 0 0 1]]
[[9 2]
  [8 9]
  [1 4]]
```

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

代码：

```
a=np.array([[1,2,3,4],
                    [5,6,7,8],
                    [9,10,11,12]])
print(a)
print(a[2,3],a[0,0])
```

输出：

```
[[ 1   2   3   4]
 [ 5   6   7   8]
 [ 9 10 11 12]]
12 1
```

6.把上一题的 a 数组的[0 到 1 行,2 到 3 列]，放到 b 里面去，(此处不需要重新建立 a,直接调用即可)

(1)输出 b

(2)输出 b 的(0,0)处元素的值

代码：

```
a=np.array([[1,2,3,4],
                    [5,6,7,8],
                    [9,10,11,12]])
b=a[0:1,2:3]
print(b[0,0])
```

输出：

```
3
```

7.把第 5 题的数组 a 的最后两行的所有元素放到 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)；

代码：

```
a=np.array([[1,2,3,4],
                    [5,6,7,8],
```

```
                [9,10,11,12]])
c=a[1:2,:]
print(c)
print(c[0,-1])
```
输出：
```
[[5 6 7 8]]
```
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)处这三个元素（提示：使用 print(a[[0,1,2],[0,1,0]])
代码：
```
a=np.array([[1,2],
                [3,4],
                [5,6]])
print(a[[0,1,2],[0,1,0]])
```
输出：
```
[1 4 5]
```
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)处的元素(提示使用 b=np.array([0,2,0,1]) print(a[np.arange(4),b]))
代码：
```
a=np.array([[1,2,3],
                [4,5,6],
                [7,8,9],
                [10,11,12]])
b=np.array([0,2,0,1])
print(a[np.arange(4),b])
print(a[[0,1,2,3],[0,2,0,1]])
```
输出：
```
[ 1    6    7 11]
```
说明：
arrange(4)表示数组[0,1,2,3]
10.对 9 中输出的四个元素，每个都加上 10，然后重新输出矩阵 a。(提示：a[np.arange(4),b]+=10)
代码：
```
a=np.array([[1,2,3],
                [4,5,6],
                [7,8,9],
                [10,11,12]])
b=np.array([0,2,0,1])
a[np.arange(4),b]+=10
#a[np.arange(4),b]=a[np.arange(4),b]+10
print(a)
```
输出：
```
[[11    2    3]
 [ 4    5 16]
 [17    8    9]
 [10 21 12]]
```

二.array 的数学运算

11.执行 x 的 np.array([1,2]),然后输出 x 的数据类型。（答案是 int64）

代码：

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

输出:

```
int32
```

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

代码：

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

输出：

```
float64
```

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)

输出：

```
[[ 6.  8.]
 [10. 12.]]
[[ 6.  8.]
 [10. 12.]]
```

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

输出:

```
[[-4. -4.]
 [-4. -4.]]
[[-4. -4.]
 [-4. -4.]]
```

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

输出:

```
[[ 5. 12.]
 [21. 32.]]
[[ 5. 12.]
 [21. 32.]]
[[19. 22.]
 [43. 50.]]
```

说明：

x*y 与 np.multiply(x,y)等价，即矩阵中对应元素相乘。

np.dot(x,y)：矩阵乘法。

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

结果：

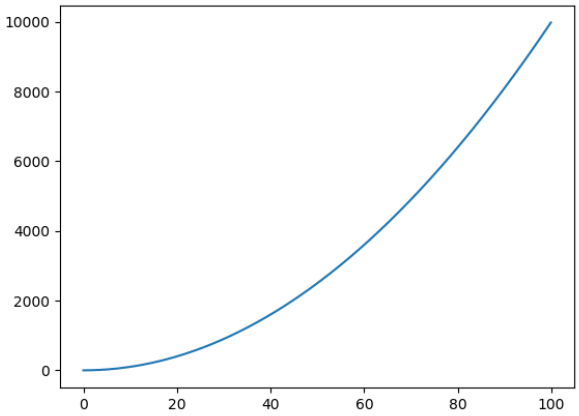
```
[[0.2        0.33333333]
 [0.42857143 0.5       ]]
[[0.2        0.33333333]
 [0.42857143 0.5       ]]
```

说明：
x/y 与 np.divide(x,y)等价，即矩阵中对应元素相除。
17.利用 13 题目中的 x，输出 x 的开方。(提示：使用函数 np.sqrt())
输出：
[[1. 1.41421356]
 [1.73205081 2.]]
说明：
np.sqrt(x)对矩阵 x 中每个元素进行开方。
18.利用 13 题目中的，x,y ,执行 print(x.dot(y)) 和 print(np.dot(x,y))
输出：
[[19. 22.]
 [43. 50.]]
[[19. 22.]
 [43. 50.]]
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))
输出：
10.0
[4. 6.]
[3. 7.]
说明：
np.sum(x)表示对矩阵中所有元素求和， np.sum(x,axis=0)表示每一列元素的和，np.sum(x,axis=1)表示每一行元素的和。
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：
2.5
[2. 3.]
[1.5 3.5]
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）
代码：
```
x=np.array([[1,2],
            [3,4]],dtype=np.float64)
x=x.T
print(x)
```
输出：
[[1. 3.]
[2. 4.]]
说明：矩阵 x 的[i][j]位置元素与转置后矩阵 xT 的[j][i]位置元素相等。

22.利用 13 题目中的 x,求 e 的指数(提示：函数 np.exp(x))
输出：
[[2.71828183 7.3890561]
 [20.08553692 54.59815003]]
说明:np.exp(x)表示对矩阵中每个元素求 e 的指数。
23.画图，y=x*x, x = np.arange(0, 100, 0.1) （提示这里用到 matplotlib.pyplot 库）
代码：

```
import numpy as np
import matplotlib.pyplot as plt
x=np.arange(0,100,0.1)
y=x*x
plt.plot(x,y)
plt.show()
```

输出：



24.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)
代码：

```
import numpy as np
import matplotlib.pyplot as plt
x=np.arange(0,3*np.pi,0.1)
y=np.sin(x)
plt.plot(x,y)
plt.show()
y=np.cos(x)
plt.plot(x,y)
plt.show()
```

25.执行下面的语句，解释运算结果，了解 nan 和 inf 的含义。
代码：

```
import numpy as np
print(np.nan)
print(np.nan==np.nan)
print(np.inf>np.nan)
print(np.nan-np.nan)
```

print(0.3==3*0.1)
输出：
nan
False
False
nan
False
说明：nan 是 not a number 的缩写，inf 是 infinity 的缩写，inf 表示正无穷,-inf 表示负无穷