**Python程式設計**

**範圍： Numpy的應用**

**銘傳大學電腦與通訊工程系**

|  |  |
| --- | --- |
| 班 級 | 電通四乙 |
| 姓 名 | 謝曜陽 |
| 學 號 | 05051071 |
| 成 績 | 應繳作業共 9 題，前9題每題10分，第10題20分，滿分為100分  共完成 題，應得 分 |
| 授課教師 | 陳慶逸 |

※請確實填寫自己寫完成題數，並且計算得分。填寫不實者(如上傳與作業明顯無關的答案，或是計算題數有誤者)，本次作業先扣50分。

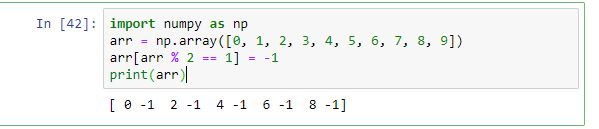
EX 1: 將 arr 中的所有奇數替換成 -1。

輸入：arr = np.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])

期望輸出：array([ 0, -1, 2, -1, 4, -1, 6, -1, 8, -1])

程式碼：

|  |
| --- |
| **import numpy as np**  **arr = np.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])**  **arr[arr % 2 == 1] = -1**  **print(arr)** |



EX 2: 式寫一函式trans1Dto2D(array)，可任意輸入1D numpy陣列，回傳為2列的2D numpy陣列。

輸入：trans1Dto2D(np.array([2,3,5,3,1,3,4,6]))

期望輸出：

array([[2, 3, 5, 3],

[1, 3, 4, 6]])

輸入：trans1Dto2D(np.arange(18))

期望輸出：

array([[ 0, 1, 2, 3, 4, 5, 6, 7, 8],

[ 9, 10, 11, 12, 13, 14, 15, 16, 17]])

輸入：trans1Dto2D(np.arange(20))

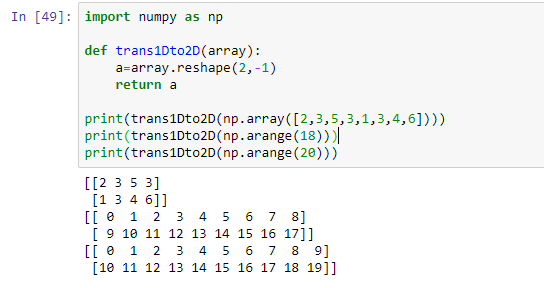
期望輸出：

array([[ 0, 1, 2, 3, 4, 5, 6, 7, 8, 9],

[10, 11, 12, 13, 14, 15, 16, 17, 18, 19]])

程式碼：

|  |
| --- |
| **import numpy as np**  **def trans1Dto2D(array):**  **a=array.reshape(2,-1)**  **return a**  **print(trans1Dto2D(np.array([2,3,5,3,1,3,4,6])))**  **print(trans1Dto2D(np.arange(18)))**  **print(trans1Dto2D(np.arange(20)))** |



EX 3: 試產生下面兩個1D numpy陣列，在轉成2D numpy陣列後，將之垂直堆疊起來。

輸入：a = array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])

b = array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1])

期望輸出：

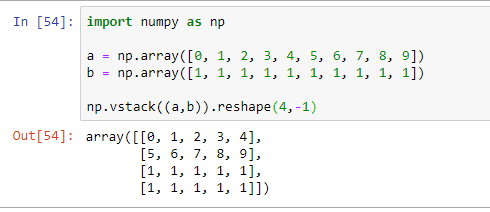
array([[0, 1, 2, 3, 4],

[5, 6, 7, 8, 9],

[1, 1, 1, 1, 1],

[1, 1, 1, 1, 1]])

|  |
| --- |
| **import numpy as np**  **a = np.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])**  **b = np.array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1])**  **np.vstack((a,b)).reshape(4,-1)** |



EX 4: 若a,b,c等三個1D numpy陣列分別如下，試垂直堆疊a, b, c以得到一個2D numpy陣列arr。再將arr中的第一列(row)與第二列進行交換。

輸入：a = array([0, 1, 2, 3, 4])

b = array([1., 1., 1., 1., 1.])

b = array([0., 0., 0., 0., 0.])

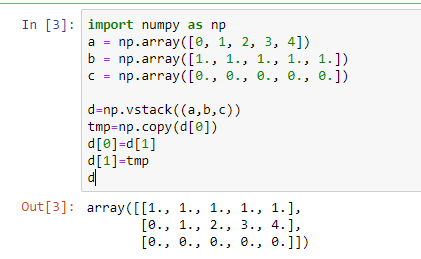
期望輸出：

array([[1., 1., 1., 1., 1.],

[0., 1., 2., 3., 4.],

[0., 0., 0., 0., 0.]])

|  |
| --- |
| **import numpy as np**  **a = np.array([0, 1, 2, 3, 4])**  **b = np.array([1., 1., 1., 1., 1.])**  **c = np.array([0., 0., 0., 0., 0.])**  **d=np.vstack((a,b,c))**  **tmp=np.copy(d[0])**  **d[0]=d[1]**  **d[1]=tmp**  **d** |



EX 5: 對於txt資料，在Numpy裡可以使用.loadtxt或是np.genfromtxt來讀取它。下面輸入的程式可以下載iris data的第一個維度(花萼的長度)，共150資料，試求其平均值(np.mean())、中位數(np.median())和標準差(np.std())。

輸入：

url = 'https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data'

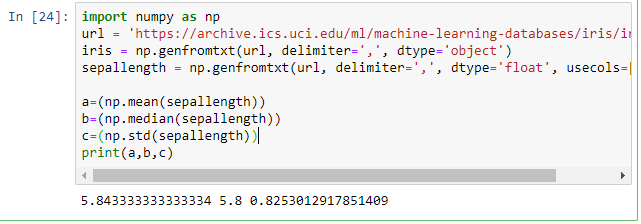
iris = np.genfromtxt(url, delimiter=',', dtype='object')

sepallength = np.genfromtxt(url, delimiter=',', dtype='float', usecols=[0])

期望輸出：

5.843 5.8 0.825

|  |
| --- |
| **import numpy as np**  **url = 'https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data'**  **iris = np.genfromtxt(url, delimiter=',', dtype='object')**  **sepallength = np.genfromtxt(url, delimiter=',', dtype='float', usecols=[0])**  **a=(np.mean(sepallength))**  **b=(np.median(sepallength))**  **c=(np.std(sepallength))**  **print(a,b,c)** |



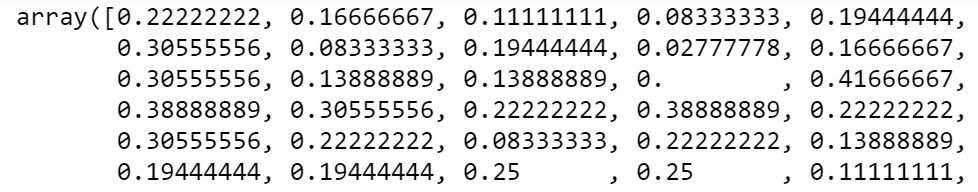
EX 6: 承續上題，試將iris sepallength的資料進行正規化，使其值的分布介於0到1之間。

輸入：

url = 'https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data'

sepallength = np.genfromtxt(url, delimiter=',', dtype='float', usecols=[0])

期望輸出：



|  |
| --- |
| **url = 'https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data'**  **sepallength = np.genfromtxt(url, delimiter=',', dtype='float', usecols=[0])** |

EX 7: 過濾iris\_2d的資料，找出滿足petallength(第三行) > 1.5 和sepallength(第一行) < 5.0的所有列。

輸入：

url = 'https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data'

iris\_2d = np.genfromtxt(url, delimiter=',', dtype='float', usecols=[0,1,2,3])

期望輸出：

array([[4.8, 3.4, 1.6, 0.2],

[4.8, 3.4, 1.9, 0.2],

[4.7, 3.2, 1.6, 0.2],

[4.8, 3.1, 1.6, 0.2],

[4.9, 2.4, 3.3, 1. ],

[4.9, 2.5, 4.5, 1.7]])

|  |
| --- |
| **# Input**  **url = 'https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data'**  **iris\_2d = np.genfromtxt(url, delimiter=',', dtype='float', usecols=[0,1,2,3])** |

EX 8: 試撰寫一個函式mindivmax(array)，該函式能將傳入的numpy 2D陣列之所有列(row)的最大值與最小值求出，並且回傳每一列計算最小值/最大值(min-by-max)的結果。

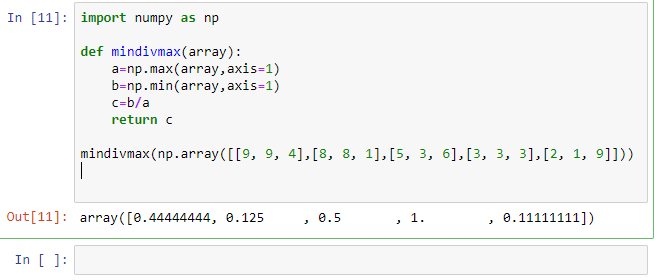
輸入：

mindivmax(np.array([[9, 9, 4],[8, 8, 1],[5, 3, 6],[3, 3, 3],[2, 1, 9]]))

期望輸出：

array([0.44444444, 0.125 , 0.5 , 1. , 0.11111111])

|  |
| --- |
| **import numpy as np**  **def mindivmax(array):**  **a=np.max(array,axis=1)**  **b=np.min(array,axis=1)**  **c=b/a**  **return c**  **mindivmax(np.array([[9, 9, 4],[8, 8, 1],[5, 3, 6],[3, 3, 3],[2, 1, 9]]))** |



EX 9: 試實現一個能計算兩個1D numpy陣列之間的歐幾里得距離的函式norm(a,b)。

輸入：

norm(np.array([1,2,3,4,5]),np.array([4,5,6,7,8]))

期望輸出：

6.7082

|  |
| --- |
|  |