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	電通四甲
姓　　名	趙啟翔
學　　號	05050130
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 9 題，應得 100 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。

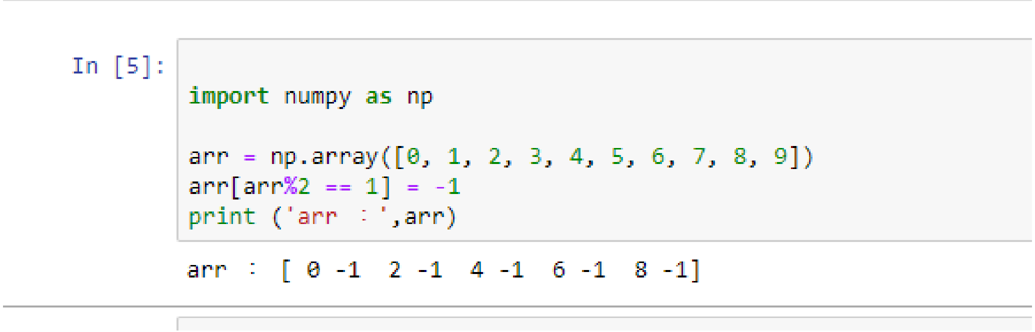
EX 1: 將 arr 中的所有奇數替換成 -1。

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 ：',arr)
```



In [5]:

```
import numpy as np

arr = np.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
arr[arr%2 == 1] = -1
print ('arr ：',arr)
```

```
arr ： [ 0 -1  2 -1  4 -1  6 -1  8 -1]
```

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]])
```

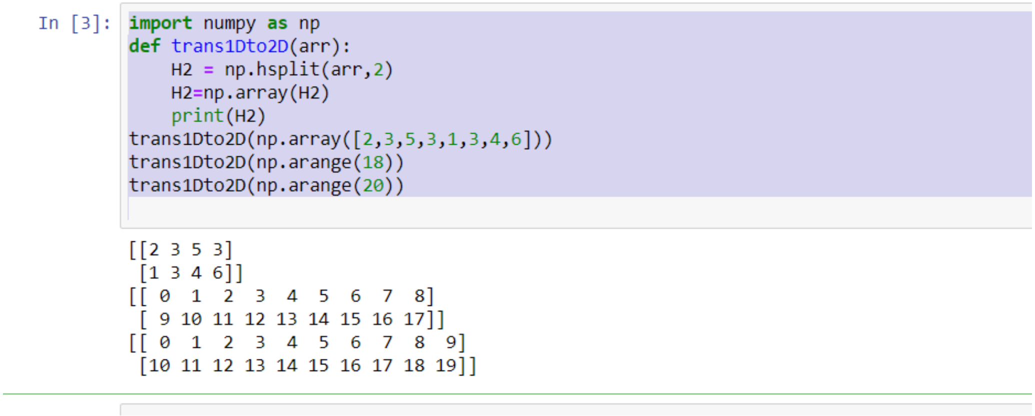
輸入：trans1Dto2D(np.arange(20))

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(arr):
    H2 = np.hsplit(arr,2)
    H2=np.array(H2)
    print(H2)
trans1Dto2D(np.array([2,3,5,3,1,3,4,6]))
trans1Dto2D(np.arange(18))
trans1Dto2D(np.arange(20))
```



In [3]:

```
import numpy as np
def trans1Dto2D(arr):
    H2 = np.hsplit(arr,2)
    H2=np.array(H2)
    print(H2)
trans1Dto2D(np.array([2,3,5,3,1,3,4,6]))
trans1Dto2D(np.arange(18))
trans1Dto2D(np.arange(20))
```

```
[[2 3 5 3]
 [1 3 4 6]]
[[ 0  1  2  3  4  5  6  7  8]
 [ 9 10 11 12 13 14 15 16 17]]
[[ 0  1  2  3  4  5  6  7  8  9]
 [10 11 12 13 14 15 16 17 18 19]]
```

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。

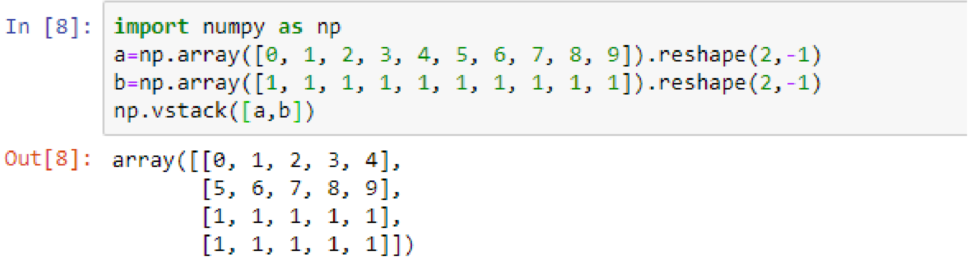
輸入：a = array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])

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]).reshape(2,-1)
b=np.array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1]).reshape(2,-1)
np.vstack([a,b])
```



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

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

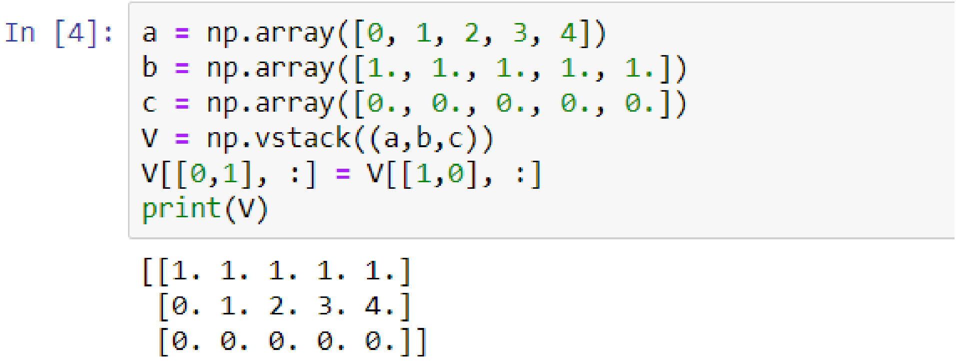
輸入：a = array([0, 1, 2, 3, 4])

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.]])
```



In [4]:

```
a = np.array([0, 1, 2, 3, 4])
b = np.array([1., 1., 1., 1., 1.])
c = np.array([0., 0., 0., 0., 0.])
V = np.vstack((a,b,c))
V[[0,1], :] = V[[1,0], :]
print(V)
```

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

```
a = np.array([0, 1, 2, 3, 4])
b = np.array([1., 1., 1., 1., 1.])
c = np.array([0., 0., 0., 0., 0.])
V = np.vstack((a,b,c))
V[[0,1], :] = V[[1,0], :]
print(V)
```

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：

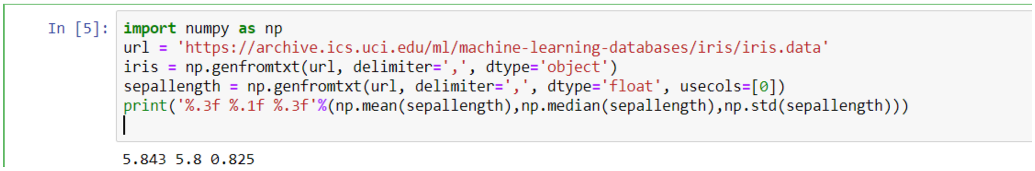
url = 'https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data'

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
```



```
In [5]: 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])
        print('%.3f %.1f %.3f'%(np.mean(sepallength),np.median(sepallength),np.std(sepallength)))

        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])
print('%.3f %.1f
%.3f'%(np.mean(sepallength),np.median(sepallength),np.std(sepallength)))
```

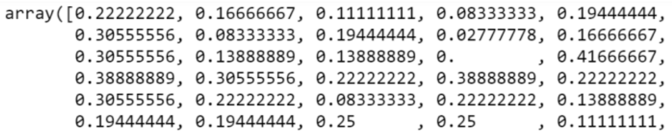
EX 6: 承續上題，試將 iris sepallength 的資料進行正規化，使其值的分布介於 0 到 1 之間。

輸入：

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

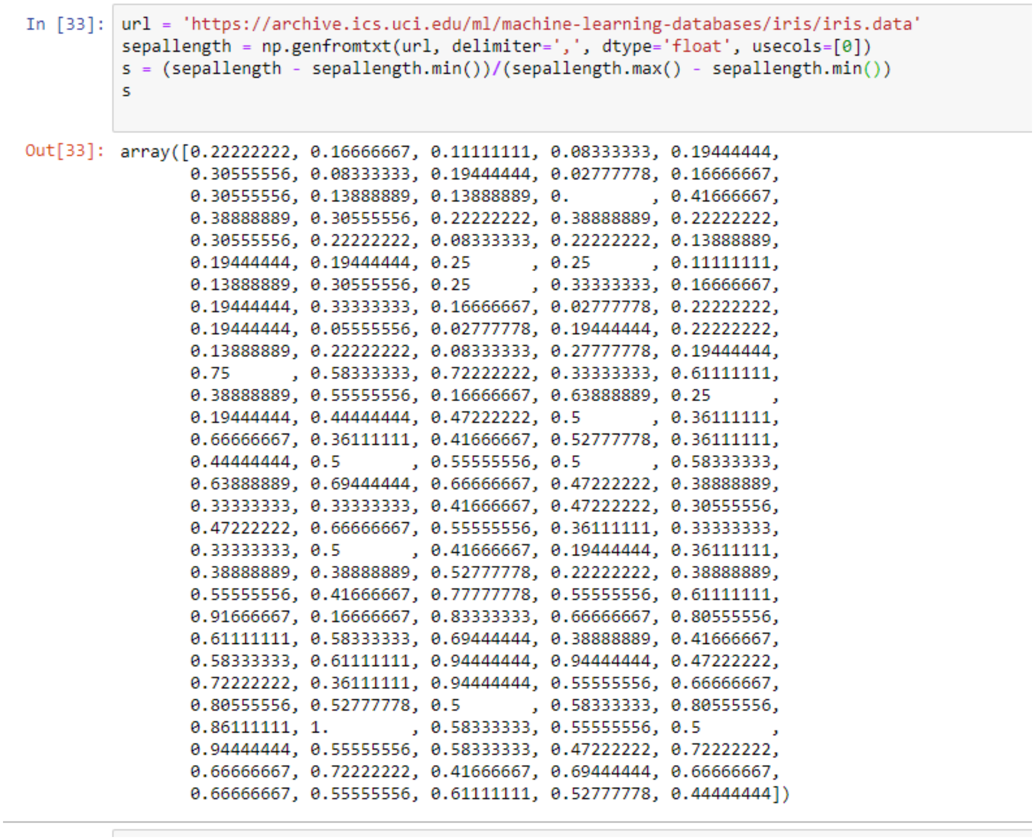
sepallength = np.genfromtxt(url, delimiter=',', dtype='float', usecols=[0])

期望輸出：



```
array([0.22222222, 0.16666667, 0.11111111, 0.08333333, 0.19444444,
       0.30555556, 0.08333333, 0.19444444, 0.02777778, 0.16666667,
       0.30555556, 0.13888889, 0.13888889, 0.        , 0.41666667,
       0.38888889, 0.30555556, 0.22222222, 0.38888889, 0.22222222,
       0.30555556, 0.22222222, 0.08333333, 0.22222222, 0.13888889,
       0.19444444, 0.19444444, 0.25      , 0.25      , 0.11111111,
```

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



```
In [33]: url = 'https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data'
         sepallength = np.genfromtxt(url, delimiter=',', dtype='float', usecols=[0])
         s = (sepallength - sepallength.min())/(sepallength.max() - sepallength.min())
         s
```

```
Out[33]: array([0.22222222, 0.16666667, 0.11111111, 0.08333333, 0.19444444,
                0.30555556, 0.08333333, 0.19444444, 0.02777778, 0.16666667,
                0.30555556, 0.13888889, 0.13888889, 0.        , 0.41666667,
                0.38888889, 0.30555556, 0.22222222, 0.38888889, 0.22222222,
                0.30555556, 0.22222222, 0.08333333, 0.22222222, 0.13888889,
                0.19444444, 0.19444444, 0.25      , 0.25      , 0.11111111,
                0.13888889, 0.30555556, 0.25      , 0.33333333, 0.16666667,
                0.19444444, 0.33333333, 0.16666667, 0.02777778, 0.22222222,
                0.19444444, 0.05555556, 0.02777778, 0.19444444, 0.22222222,
                0.13888889, 0.22222222, 0.08333333, 0.27777778, 0.19444444,
                0.75      , 0.58333333, 0.72222222, 0.33333333, 0.61111111,
                0.38888889, 0.55555556, 0.16666667, 0.63888889, 0.25      ,
                0.19444444, 0.44444444, 0.47222222, 0.5       , 0.36111111,
                0.66666667, 0.36111111, 0.41666667, 0.52777778, 0.36111111,
                0.44444444, 0.5       , 0.55555556, 0.5       , 0.58333333,
                0.63888889, 0.69444444, 0.66666667, 0.47222222, 0.38888889,
                0.33333333, 0.33333333, 0.41666667, 0.47222222, 0.30555556,
                0.47222222, 0.66666667, 0.55555556, 0.36111111, 0.33333333,
                0.33333333, 0.5       , 0.41666667, 0.19444444, 0.36111111,
                0.38888889, 0.38888889, 0.52777778, 0.22222222, 0.38888889,
                0.55555556, 0.41666667, 0.77777778, 0.55555556, 0.61111111,
                0.91666667, 0.16666667, 0.83333333, 0.66666667, 0.80555556,
                0.61111111, 0.58333333, 0.69444444, 0.38888889, 0.41666667,
                0.58333333, 0.61111111, 0.94444444, 0.94444444, 0.47222222,
                0.72222222, 0.36111111, 0.94444444, 0.55555556, 0.66666667,
                0.80555556, 0.52777778, 0.5       , 0.58333333, 0.80555556,
                0.86111111, 1.        , 0.58333333, 0.55555556, 0.5       ,
                0.94444444, 0.55555556, 0.58333333, 0.47222222, 0.72222222,
                0.66666667, 0.72222222, 0.41666667, 0.69444444, 0.66666667,
                0.66666667, 0.55555556, 0.61111111, 0.52777778, 0.44444444])
```

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。

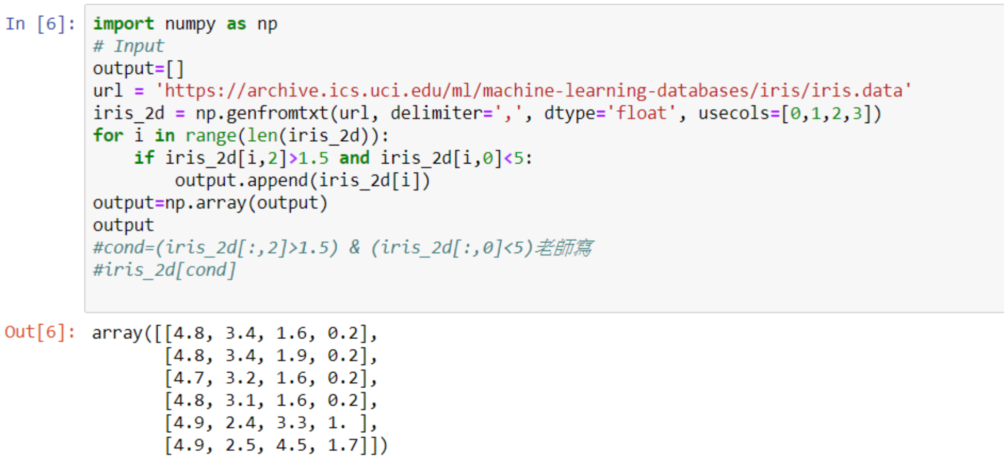
輸入：

```
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]])
```

```
import numpy as np
# Input
output=[]
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])
for i in range(len(iris_2d)):
   if iris_2d[i,2]>1.5 and iris_2d[i,0]<5:
      output.append(iris_2d[i])
output=np.array(output)
output
#cond=(iris_2d[:,2]>1.5) & (iris_2d[:,0]<5)老師寫
#iris_2d[cond]
```



```
In [6]: import numpy as np
        # Input
        output=[]
        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])
        for i in range(len(iris_2d)):
            if iris_2d[i,2]>1.5 and iris_2d[i,0]<5:
                output.append(iris_2d[i])
        output=np.array(output)
        output
        #cond=(iris_2d[:,2]>1.5) & (iris_2d[:,0]<5)老師寫
        #iris_2d[cond]

Out[6]: 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]])
```

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。

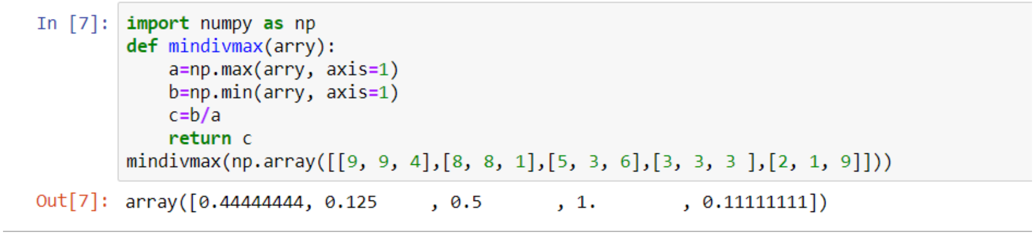
輸入：

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(arry):
    a=np.max(arry, axis=1)
    b=np.min(arry, 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]]))
```



```
In [7]: import numpy as np
        def mindivmax(arry):
            a=np.max(arry, axis=1)
            b=np.min(arry, 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]]))

Out[7]: array([0.44444444, 0.125     , 0.5       , 1.        , 0.11111111])
```

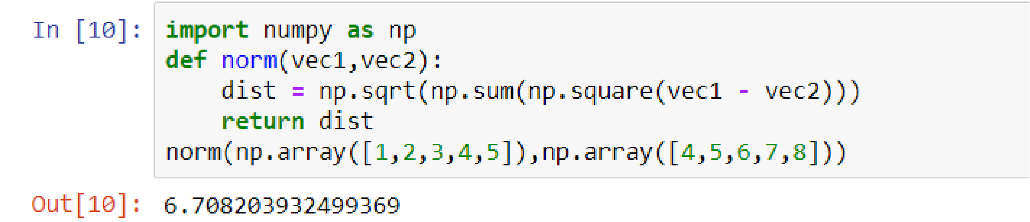
EX 9: 試實現一個能計算兩個 1D numpy 陣列之間的歐幾里得距離的函式 norm(a,b)。

輸入：

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

期望輸出：

```
6.7082
```



```
In [10]: import numpy as np
         def norm(vec1,vec2):
             dist = np.sqrt(np.sum(np.square(vec1 - vec2)))
             return dist
         norm(np.array([1,2,3,4,5]),np.array([4,5,6,7,8]))

Out[10]: 6.708203932499369
```

```
import numpy as np
def norm(vec1,vec2):
    dist = np.sqrt(np.sum(np.square(vec1 - vec2)))
    return dist
norm(np.array([1,2,3,4,5]),np.array([4,5,6,7,8]))
```