**IT8701 Introduction to Programming for Data Science**

**Lab 02 – Data Manipulation using the Numpy module**

**Practical 2 – Submission Question**

# 6. Manipulating Array Content

### Task 1: Concatenate arrays (SUBMISSION REQUIRED)

<https://docs.scipy.org/doc/numpy/reference/generated/numpy.concatenate.html>

|  |
| --- |
| 1. Write Python code to concatenate the following arrays on the **axis=0**   **a = np.array([[ 1,2,3],[ 4,5,6],[7,8,9]])**  **b = np.full((3,3),1.5)**  **c = np.arange(0,15).reshape(5,3)** |
| 1. Write Python code to concatenate the following arrays on the **axis=1**   **a = np.array([[ 1,2,3,4],[ 4,5,6,7],[7,8,9,10],[11,12,13,14]])**  **b = np.random.randint(100,200,(4,6))**  **c = np.arange(0,40).reshape(4,10)** |

# 8. Sorting arrays

### Task 1: Basic sorting (SUBMISSION REQUIRED)

|  |
| --- |
| 1. Write Python code to sort the following 1-d array **in-place** using the sort method   **arr\_1 = np.random.randint(100,200,10)**  Your output should be similar to that below. |
| 1. Write Python code to sort the following 2-d array *in-place* by columns   **arr\_2 = np.random.randint(1,20,(3,5))**  Your output should be similar to that below. |
| 1. Write Python code to sort the following 2-d array by rows *without affecting the original* .   **arr\_3 = np.random.randint(100,200,(2,5))** |

# 9. Subsetting and Indexing

### Task 2: Boolean indexing (SUBMISSION REQUIRED)

|  |
| --- |
| Given the following Numpy array, answer the questions from (a) to (b) below.  **a = np.array((np.arange(0,10),  np.arange(10,20),  np.arange(20,30),  np.arange(30,40)))**  **b = np.random.randint(100,200,(3,3))** |
| 1. Write Python code to return all the even numbers in array a 2. Write Python code to return all the numbers that are greater than 150 in array b |

# 

# 11. Statistical methods

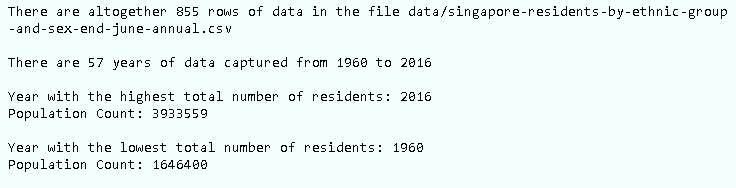
### Task 1: sum, mean (SUBMISSION REQUIRED)

|  |
| --- |
| **a = np.array((np.arange(0,10),  np.arange(10,20),  np.arange(20,30),  np.arange(30,40)))**   1. What is the sum of all the numbers in array a? 2. What is their mean? 3. What is the sum of all the numbers in each row?   Using the Numpy statistical functions **sum()** and **mean()**, compute the information for array a as specified above. Your output should resemble the screenshot shown below. |
|  |
|  |

# 12. File I/O on Numpy arrays

### Task 1: Load and save a numpy array (SUBMISSION REQUIRED)

Complete the following code that uses the Numpy **loadtxt** method to read the contents of an external csv file and display pertinent information about the data inside the file as shown on the sample screens below



|  |
| --- |
| import numpy as np  **### Read the csv file with the loadtxt() function**  fname = "data/singapore-residents-by-ethnic-group-and-sex-end-june-annual.csv"  data = np.loadtxt(fname, skiprows=1,dtype=[('year','i8'),('level\_1','U50'), ('value','i8')],delimiter=",")  **### Print out total rows of data in the file**  print("There are altogether " + **\_\_\_\_\_\_\_\_\_** + " rows of data in the file " + **\_\_\_\_\_\_\_\_** )  print()  **### Print out the number of years of data captured**  data\_years = data['year'] # Just extract the year column  years = np.unique(data\_years) # Get the unique values in this column  print("There are " + **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** )  print()  **### Extract only the rows with “Total Residents" in the “level\_1” column**  keyword = 'Total Residents'  column\_to\_search = data['level\_1']  out = [i for i, v in enumerate(column\_to\_search) if keyword in v]  data\_totalresidents = data[out]  **### Print out the year which has the highest total number of residents**  max = data\_totalresidents['value'].max()  argmax = data\_totalresidents['value'].argmax()  print("Year with the highest total number of residents: " + str(data\_totalresidents[argmax]['year']))  print("Population Count: " + str(max))  print()  **### Print out the year which has the lowest total number of residents**  min = **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  argmin = **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  print(**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**)  print(**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**)  print() |