Lab7-WorkSheetOnly

September 24, 2020

[2]:
```
%%html
<!--Script block to left align Markdown Tables-->
<style>
  table {margin-left: 0 !important;}
</style>
```

[2]:
```
<IPython.core.display.HTML object>
```

[3]:
```
# Preamble script block to identify host, user, and kernel
import sys
! hostname
! whoami
print(sys.executable)
print(sys.version)
print(sys.version_info)
```

```
project-7320abff-2260-4945-8a16-865423659f2e
user
/usr/bin/python3
3.8.2 (default, Jul 16 2020, 14:00:26)
[GCC 9.3.0]
sys.version_info(major=3, minor=8, micro=2, releaselevel='final', serial=0)
```

0.1 Full name: Kahla Archibald

0.2 R#: 11706661

0.3 HEX: 0xb2a125

0.4 Lab 7

0.5 Date: 9/15/2020

0.5.1 Example- 1D Arrays

Let's create a 1D array from the 2000s (2000-2009):

[4]:
```
import numpy as np              #First, we need to impoty "numpy"
mylist = [2000,2001,2002,2003,2004,2005,2006,2007,2008,2009]          #Create a␣
 ↪list of the years
```

```
print(mylist)                    #Check how it looks
np.array(mylist)                 #Define it as a numpy array
```

```
[2000, 2001, 2002, 2003, 2004, 2005, 2006, 2007, 2008, 2009]
```

[4]:
```
array([2000, 2001, 2002, 2003, 2004, 2005, 2006, 2007, 2008, 2009])
```

0.5.2 Example- n-Dimensional Arrays

Let's create a 5x2 array from the 2000s (2000-2009):

[8]:
```
myotherlist = [[2000,2001],[2002,2003],[2004,2005],[2006,2007],[2008,2009]]
 ↪#Since I want a 5x2 array, I should group the years two by two
print(myotherlist)                #See how it looks as a list
np.array(myotherlist)             #See how it looks as a numpy array
```

```
[[2000, 2001], [2002, 2003], [2004, 2005], [2006, 2007], [2008, 2009]]
```

[8]:
```
array([[2000, 2001],
       [2002, 2003],
       [2004, 2005],
       [2006, 2007],
       [2008, 2009]])
```

0.5.3 Exercise 1

Using the numbers below: #### 9,18,333,22,25,120,7,43,50,3,7,94,953,57,11,154

A) Create a 1D array B) Create a 4x4 array so that: - all numbers in the first row are single digits - all numbers in the second row are two digit and even numbers - all numbers in the third row are two digit and odd numbers - all numbers in the fourth row are three digit numbers

[0]:
```
import numpy as np
anotherlist = [[9,18,25,333],[7,22,43,120],[3,50,57,953],[7,94,11,154]]
print(anotherlist)
np.array(anotherlist)
```

0.5.4 Example- 1D Array Arithmetic

- Define a 1D array with [0,12,24,36,48,60,72,84,96]
- Multiple all elements by 2
- Take all elements to the power of 2
- Find the maximum value of the array and its position
- Find the minimum value of the array and its position
- Define another 1D array with [-12,0,12,24,36,48,60,72,84]
- Find the summation and subtraction of these two arrays
- Find the multiplication of these two arrays

[29]:
```
import numpy as np          #import numpy
Array1 = np.array([0,12,24,36,48,60,72,84,96])     #Step1: Define Array1
print(Array1)
print(Array1*2)     #Step2: Multiple all elements by 2
print(Array1**2)     #Step3: Take all elements to the power of 2
print(np.power(Array1,2)) #Another way to do the same thing, by using a␣
 ↪function in numpy
print(np.max(Array1))     #Step4: Find the maximum value of the array
print(np.argmax(Array1))     ##Step4: Find the postition of the maximum value
print(np.min(Array1))     #Step5: Find the minimum value of the array
print(np.argmin(Array1))     ##Step5: Find the postition of the minimum value
Array2 = np.array([-12,0,12,24,36,48,60,72,84])     #Step6: Define Array2
print(Array2)
print(Array1+Array2)          #Step7: Find the summation of these two arrays
print(Array1-Array2)          #Step7: Find the subtraction of these two arrays
print(Array1*Array2)          #Step8: Find the multiplication of these two arrays
```

```
[ 0 12 24 36 48 60 72 84 96]
[  0  24  48  72  96 120 144 168 192]
[   0  144  576 1296 2304 3600 5184 7056 9216]
[   0  144  576 1296 2304 3600 5184 7056 9216]
96
8
0
0
[-12   0  12  24  36  48  60  72  84]
[-12  12  36  60  84 108 132 156 180]
[12 12 12 12 12 12 12 12 12]
[   0    0  288  864 1728 2880 4320 6048 8064]
```

0.5.5 Example- n-Dimensional Array Arithmetic

- Define a 2x2 array with [5,10,15,20]
- Define another 2x2 array with [3,6,9,12]
- Find the summation and subtraction of these two arrays
- Find the minimum number in the multiplication of these two arrays
- Find the position of the maximum in the multiplication of these two arrays
- Find the mean of the multiplication of these two arrays
- Find the mean of the first row of the multiplication of these two arrays

[42]:
```
import numpy as np          #import numpy
Array1 = np.array([[5,10],[15,20]])     #Step1: Define Array1
print(Array1)
Array2 = np.array([[3,6],[9,12]])     #Step2: Define Array2
print(Array2)
print(Array1+Array2)     #Step3: Find the summation
print(Array1-Array2)     #Step3: Find the subtraction
```

```
MultArray = Array1@Array2          #Step4: To perform a typical matrix␣
 ↪multiplication (or matrix product)
MultArray1 = Array1.dot(Array2)          #Step4: Another way To perform a ␣
 ↪matrix multiplication
print(MultArray)
print(MultArray1)
print(np.min(MultArray))      #Step4: Find the minimum value of the␣
 ↪multiplication
print(np.argmax(MultArray))      ##Step5: Find the postition of the maximum␣
 ↪value
print(np.mean(MultArray))      ##Step6: Find the mean of the multiplication of␣
 ↪these two arrays
print(np.mean(MultArray[0,:]))      ##Step7: Find the mean of the first row of␣
 ↪the multiplication of these two arrays
```

```
[[ 5 10]
 [15 20]]
[[ 3  6]
 [ 9 12]]
[[ 8 16]
 [24 32]]
[[2 4]
 [6 8]]
[[105 150]
 [225 330]]
[[105 150]
 [225 330]]
105
3
202.5
127.5
```

0.5.6 Exercise 2

Using the numbers below: #### 88,-5,9,22,46,2,-1,0

A) Create a 2x4 array and print it out B) Add 6 to all elements, store it as a new object, and print it out C) Multiply all elements by 5, store it as a new object, and print it out D) Divide all elements by 7, store it as a new object, and print it out

[0]:
```
array3 = np.array([[88,-5,9,22],[46,2,-1,0]])
print(array3)
array4 = array3+6
print(array4)
array5 = array4*5
print(array5)
array6 = array5/7
print(array6)
```

0.5.7 Example- 1D Array Comparison

- Define a 1D array with [1.0,2.5,3.4,7,7]
- Define another 1D array with [5.0/5.0,5.0/2,6.8/2,21/3,14/2]
- Compare and see if the two arrays are equal
- Define another 1D array with [6,1.4,2.2,7.5,7]
- Compare and see if the first array is greater than or equal to the third array

[1]:
```
import numpy as np          #import numpy
Array1 = np.array([1.0,2.5,3.4,7,7])     #Step1: Define Array1
print(Array1)
Array2 = np.array([5.0/5.0,5.0/2,6.8/2,21/3,14/2])     #Step2: Define Array1
print(Array2)
print(np.equal(Array1, Array2))             #Step3: Compare and see if the two␣
 ↪arrays are equal
Array3 = np.array([6,1.4,2.2,7.5,7])     #Step4: Define Array3
print(Array3)
print(np.greater_equal(Array1, Array3))             #Step3: Compare and see if␣
 ↪the two arrays are equal
```

```
[1.  2.5 3.4 7.  7. ]
[1.  2.5 3.4 7.  7. ]
[ True  True  True  True  True]
[6.  1.4 2.2 7.5 7. ]
[False  True  True False  True]
```

0.5.8 Exercise 3- n-Dimensional Array Comparison

Using the numbers below: #### 4,9,23,39,40,55,68,72

A) Create a 2x2 array with all the even numbers and print it out B) Create a 2x2 array with all the odd numbers and print it out C) Compare the two arrays using the "less_equal" function from numpy package

[5]:
```
import numpy as np
array1 = np.array([4,40,68,72])
print(array1)
array2 = np.array([9,23,39,55])
print(array2)
print(np.less_equal(array1, array2))
```

```
[ 4 40 68 72]
[ 9 23 39 55]
[ True False False False]
```

0.5.9 Example- Copying and Deleting Arrays and Elements

- Define a 1D array, named "x" with [1,2,3]
- Define "y" so that "y=x"
- Define "z" as a copy of "x"

- Discuss the difference between y and z
- Delete the second element of x

[7]:
```
import numpy as np          #import numpy
x = np.array([1,2,3])      #Step1: Define x
print(x)
y = x                      #Step2: Define y as y=x
print(y)
z = np.copy(x)             #Step3: Define z as a copy of x
print(z)
# For Step4: They look similar but check this out:
x[1] = 8                   # If we change x ...
print(x)
print(y)
print(z)
# By modifying x, y changes but z remains as a copy of the initial version of x.
x = np.delete(x, 1)       #Step5: Delete the second element of x
print(x)
```

```
[1 2 3]
[1 2 3]
[1 2 3]
[1 8 3]
[1 8 3]
[1 2 3]
[1 3]
```

0.5.10 Exercise 4

Using the numbers below: #### 1,2,3,4,5,6,7,8,9

A) Create a 1D array and print it out B) Delete all the even numbers and print out the new version C) Make a copy of the array from step B and print it out

[0]:
```
import numpy as np
x = np.array([1,2,3,4,5,6,7,8,9])
print(x)
x = np.delete(x, 1)
x = np.delete(x, 2)
x = np.delete(x, 3)
x = np.delete(x, 4)

print(x)
y = np.copy(x)
pr
```

0.5.11 Example- Sorting 1D Arrays

Define a 1D array as ['FIFA 2020','Red Dead Redemption','Fallout','GTA','NBA 2018','Need For Speed'] and print it out. Then, sort the array alphabetically.

[10]:
```
import numpy as np          #import numpy
games = np.array(['FIFA 2020','Red Dead Redemption','Fallout','GTA','NBA␣
 ↪2018','Need For Speed'])
print(games)
print(np.sort(games))
```

```
['FIFA 2020' 'Red Dead Redemption' 'Fallout' 'GTA' 'NBA 2018'
 'Need For Speed']
['FIFA 2020' 'Fallout' 'GTA' 'NBA 2018' 'Need For Speed'
 'Red Dead Redemption']
```

0.5.12 Example- Sorting n-Dimensional Arrays

Define a 3x3 array with 17,-6,2,86,-12,0,0,23,12 and print it out. Then, sort the array.

[18]:
```
import numpy as np          #import numpy
a = np.array([[17,-6,2],[86,-12,0],[0,23,12]])
print(a)
print ("Along columns : \n", np.sort(a,axis = 0) ) #This will be sorting in␣
 ↪each column
print ("Along rows : \n", np.sort(a,axis = 1) ) #This will be sorting in each␣
 ↪row
print ("Sorting by default : \n", np.sort(a) )          #Same as above
print ("Along None Axis : \n", np.sort(a,axis = None) ) #This will be sorted␣
 ↪like a 1D array
```

```
[[ 17  -6   2]
 [ 86 -12   0]
 [  0  23  12]]
Along columns :
 [[  0 -12   0]
 [ 17  -6   2]
 [ 86  23  12]]
Along rows :
 [[ -6   2  17]
 [-12   0  86]
 [  0  12  23]]
Sorting by default :
 [[ -6   2  17]
 [-12   0  86]
 [  0  12  23]]
Along None Axis :
 [-12  -6   0   0   2  12  17  23  86]
```

0.5.13 Exercise 5

A) Google: "Python how to Sort Numpy", find a good link with appropriate explanation, and add the link to your notebook Then, use the 4x4 array from Exercise 1, part2 and ... B) Print it out C) Sort it along the rows and print it out D) Sort it along the columns and print it out

[0]:
```
import numpy as np
a = np.array([[9,18,25,333],[7,22,43,120],[3,50,57,953],[7,94,11,154]])
print(a)
np.sort(a)                # sort along the last axis
print(np.sort(a, axis = 0))     # sort the flattened array
print(np.sort(a, axis = 1))        # sort along the first axis
```

0.5.14 Example- Slicing 1D Arrays

Define a 1D array as [1,3,5,7,9], slice out the [3,5,7] and print it out.

[34]:
```
import numpy as np          #import numpy
a = np.array([1,3,5,7,9])     #Define the array
print(a)
aslice = a[1:4]              #slice the [3,5,7]
print(aslice)                #print it out
```

```
[1 3 5 7 9]
[3 5 7]
```

0.5.15 Example- Slicing n-Dimensional Arrays

Define a 5x5 array with "Superman, Batman, Jim Hammond, Captain America, Green Arrow, Aquaman, Wonder Woman, Martian Manhunter, Barry Allen, Hal Jordan, Hawkman, Ray Palmer, Spider Man, Thor, Hank Pym, Solar, Iron Man, Dr. Strange, Daredevil, Ted Kord, Captian Marvel, Black Panther, Wolverine, Booster Gold, Spawn" and print it out. Then: - Slice the first column and print it out - Slice the third row and print it out - Slice 'Wolverine' and print it out - Slice a 3x3 array with 'Wonder Woman, Ray Palmer, Iron Man, Martian Manhunter, Spider Man, Dr. Strange, Barry Allen, Thor, Daredevil'

[35]:
```
import numpy as np          #import numpy
Superheroes = np.array([['Superman', 'Batman', 'Jim Hammond', 'Captain␣
 ↪America', 'Green Arrow'],
                ['Aquaman', 'Wonder Woman', 'Martian Manhunter', 'Barry Allen',␣
 ↪'Hal Jordan'],
                ['Hawkman', 'Ray Palmer', 'Spider Man', 'Thor', 'Hank Pym'],
                ['Solar', 'Iron Man', 'Dr. Strange', 'Daredevil', 'Ted Kord'],
                ['Captian Marvel', 'Black Panther', 'Wolverine', 'Booster Gold',␣
 ↪'Spawn']])
print(Superheroes) #Step1
print(Superheroes[:,0])
```

```
print(Superheroes[2,:])
print(Superheroes[4,2])
print(Superheroes[1:4,1:4])
```

```
[['Superman' 'Batman' 'Jim Hammond' 'Captain America' 'Green Arrow']
 ['Aquaman' 'Wonder Woman' 'Martian Manhunter' 'Barry Allen' 'Hal Jordan']
 ['Hawkman' 'Ray Palmer' 'Spider Man' 'Thor' 'Hank Pym']
 ['Solar' 'Iron Man' 'Dr. Strange' 'Daredevil' 'Ted Kord']
 ['Captian Marvel' 'Black Panther' 'Wolverine' 'Booster Gold' 'Spawn']]
['Superman' 'Aquaman' 'Hawkman' 'Solar' 'Captian Marvel']
['Hawkman' 'Ray Palmer' 'Spider Man' 'Thor' 'Hank Pym']
Wolverine
[['Wonder Woman' 'Martian Manhunter' 'Barry Allen']
 ['Ray Palmer' 'Spider Man' 'Thor']
 ['Iron Man' 'Dr. Strange' 'Daredevil']]
```

0.5.16 Exercise 6

Using the names below:

"Cartman, Kenny, Kyle, Stan, Butters, Wendy, Chef, Mr. Mackey, Randy, Sharon, Sheila, Towelie"

A) Create a 3x4 array and print it out B) Sort the array alphabetically by column and print it out C) From the sorted array, slice out a 2x2 array with ('Cartman, Randy, Sharon, Wendy'), and print it out

[6]:
```
import numpy as np
names = np.
 ↪array([["Cartman","Kenny","Kyle","Stan"],["Butters","Wendy","Chef","Mr.
 ↪Mackey"],["Randy","Sharon","Sheila","Towelie"]])
print(names)
alpha = np.sort(names)
print(alpha)
beta = [[alpha[0,0],alpha[2,0]],[alpha[2,1],alpha[1,3]]]
print(beta)
```

```
[['Cartman' 'Kenny' 'Kyle' 'Stan']
 ['Butters' 'Wendy' 'Chef' 'Mr.Mackey']
 ['Randy' 'Sharon' 'Sheila' 'Towelie']]
[['Cartman' 'Kenny' 'Kyle' 'Stan']
 ['Butters' 'Chef' 'Mr.Mackey' 'Wendy']
 ['Randy' 'Sharon' 'Sheila' 'Towelie']]
[['Cartman', 'Randy'], ['Sharon', 'Wendy']]
```

[0]:
```
```