Department of Computer Technology and Information Systems

CTIS264 – Computer Algorithms

Spring 2019 - 2020

Lab Guide 3 - Week 4

Instructor : Füsun YÜRÜTEN **Assistant** : Leyla SEZER
OBJECTIVE: • **Numpy module** • **Two-dim arrays with multiplication table** • **Timing**

Numpy provides**:**

- Multi-dimensional arrays (data structures) for fast, efficient data storage and manipulation.
- Useful functions to manipulate arrays of data.
- Tools for importing and exporting data to and from programs.

- To use the numpy package, it must first be imported as before.

```
import numpy as np
np.functionName(…)
```

- Table shows some functions to create arrays.

Function	Output
array	Creates a numpy array object
arange	Creates a numpy array with values within the specified range (start,stop,step)
ones	Creates a numpy array containing all ones, with the given shape.
zeros	Creates a numpy array containing all zeros, with the given shape.

Two-Dim array creation:

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

Output:

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

arange function:

```
b = np.arange(1,10,2)
print(b)
```

Output:

```
array([1, 3, 5, 7, 9])
```

```
t = np.arange(1,10)
print(t)
```

Output:

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

zeros and ones functions:

```
arr = np.zeros((3,8))
print(arr)
```

Output:

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

```
arr = np.ones((2,3))
print(arr)
```

Output:

```
[[1. 1. 1.]
[1. 1. 1.]]
```

reshape function:

```
m = np.arange(1,10)
x = m.reshape((3,3))
```

Output:

```
print(x)
[[1 2 3]
 [4 5 6]
 [7 8 9]]

print(m)
[1 2 3 4 5 6 7 8 9]
```

- Table shows some numpy array built-in functions.

Function Name	Description
`np.sum()`	Computes the sum of values in given array.
`np.mean()`	Computes the mean(average) value in given array.
`np.max()`	Computes the maximum value in given array.
`np.min()`	Computes the minimum value in given array.
`np.argmax()`	Computes the index of the maximum value in given array.
`np.argmin()`	Computes the index of the minimum value in given array.

- In numpy there is a sub-module, random, that contains a number of functions for generating random arrays.
  For example:
  - rand(n) – generates an array of n random values between 0-1.
  - randint(l,h,n) – generates an array of n random integer values between low (inclusive) and high (exclusive).

```
r_vals = np.random.rand(10)
print(r_vals)
```

Output:

```
[0.07584916 0.46695138 0.27263685 0.00480137 0.29172315 0.15344114
 0.80224808 0.36345661 0.26616129 0.38245762]
```

```
r_vals = np.random.randint(1,26,12)
print(r_vals)
```

Output:

```
[18 24 20 13 10 20 15 20 5 6 1 18]
```

- In numpy it is easy to multiply each array element with any integer(for example 3). You can also make, addition, subtraction, division, power and mod in a same way.

```
r_vals = r_vals * 3
print(r_vals)
```

Output:

```
[54 72 60 39 30 60 45 60 15 18 3 54]
```

- You can use Boolean arrays to select a subset of an array. If we want to find the indexes of the elements that meet the given condition(s), you can do so using the where function.

```
over_30 = r_vals > 30
print(over_30)
```

Output:

```
[ True  True  True  True False  True  True  True False False False  True]
```

```
print(r_vals[over_30])
```

Output:

```
[54 72 60 39 60 45 60 54]
```

```
print(np.where(over_30))
```

Output:

```
(array([ 0,  1,  2,  3,  5,  6,  7, 11], dtype=int64),)
```

Q1.

Write a Python program that;

- Initialize two one dimensional arrays, TeamA and TeamB to store the scores of Football games. There are 10 games in a season. Initialize both arrays to store 10 random integer values between 0 and 10.
- Calculate and display the number of games won by TeamA.
- Display the scores of the games that TeamB lost.
- Calculate the maximum score for TeamA.
- Calculate the average score for all games (both teams scores should be included in the average). To join the two arrays, you can concatenate, just like we have concatenated strings. Use the np.concatenate((a , b)) function to create a new array which contains elements from both a and b. Notice that the function takes the two arrays as a tuple.
- Calculate the point spread (difference between each team's scores) for each game. The point spread should always be a positive number.

Output:

```
Team A: [ 6  0 10  3  1  6  8  6  8  7]
Team B: [ 3 10  4  9  1  0  0  4  1  0]

Team A has won 7 games
Scores for team b losses: [3 4 0 0 4 1 0]
Maximum score for Team A: 10
Average score(all games): 4.35
Point difference for each game: [ 3 10  6  6  0  6  8  2  7  7]
```

Q2.

- Analyze the following algorithms theoretically. Which algorithm is more efficient, with numpy arrays or without numpy?
- Make empirical analysis of time efficiency.

Without numpy arrays

```
import time

def mult_tble():
    start = time.time()
    mtb = []
    for row in range(1,300):
        tbitem = []
        for col in range(1,300):
            tbitem.append(row * col)
        mtb.append(tbitem)
        end = time.time()
    print(mtb)
    return end-start


print("Function performed %10.15f seconds"%mult_tble())
```

With numpy arrays

```
import numpy as np
import time

def mult_table(n):
    start = time.time()
    rng = np.arange(1, n+1)
    rng = rng * rng[:,None]
    end = time.time()
    print(rng)
    return end-start

print("Function performed %10.15f seconds"%mult_table(300))
```

Q3. Write a Python program that gets a list of words from the user until STOP is entered, then program puts the words which has greater than 5 characters to a new list and displays the new list content in the screen.

a) Using standard for loop
b) Using single line for loop

Output:

```
Enter a word: December
Enter a word: October
Enter a word: April
Enter a word: July
Enter a word: May
Enter a word: June
Enter a word: September
Enter a word: STOP
Words the size of 5 ['December', 'October', 'September']
```