Assignment 2 sol

November 21, 2018

```
In [ ]: Q1. Create a 1D array of numbers from 0 to 20
```

```
In [63]: import numpy as np

         x = np.arange(21)
         print(x,"size:", x.size,  type(x))
```

```
[ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20] size: 21 <class 'numpy.ndarray
```

```
In [ ]: Q2. Create a 1D array of numbers from 1 to 20
```

```
In [64]: y = np.arange(1,21,1)
         print(y, y.size)
```

```
[ 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20] 20
```

```
In [ ]: Q3. Create a 1D array of numbers from 1 to 10 with a step of 4
```

```
In [65]: z = np.arange(1, 10, 4)
         print(z)
```

```
[1 5 9]
```

```
In [ ]: Q4. Display the dimension,shape,size,type of the array
```

```
In [66]: z = np.arange(1, 10, 4)
         print(z.ndim, z.shape, z.size, type(z))
```

```
1 (3,) 3 <class 'numpy.ndarray'>
```

```
In [ ]: Q5. Create the following:
        a. A list of 5 integers
        b. A multidimensional array of dimensions 3x3
        c. A multidimensional array of zeros of dimensions 4x3
        d. A multidimensional array of zeros of dimensions 2x3 of type int32
        e. Use linspace to create a list of floats between 3 and 5
        f. Use linspace to create a list of 5 floats between 1 and 4
```

In [67]:
```
#a.
i = [5, 4, 6, 3, 2]
print("list of five integers :", i)

#b.
m = [(2,3,9), (5,9,7), (4,7,3)]
print("multi-dimentional array of 3*3 :", m,"Shape :", np.shape(m))

#c.
z = np.zeros((4,3))
print(z, "shape :",np.shape(z))

#d.
k = np.zeros((2,3), dtype = 'int32')
print(k, "shape :",np.shape(k))

#e.
l = np.linspace(3,5)
print(l)

#f.
ln = np.linspace(1, 4, 5)
print(ln)
```

```
list of five integers : [5, 4, 6, 3, 2]
multi-dimentional array of 3*3 : [(2, 3, 9), (5, 9, 7), (4, 7, 3)] Shape : (3, 3)
[[0. 0. 0.]
 [0. 0. 0.]
 [0. 0. 0.]
 [0. 0. 0.]] shape : (4, 3)
[[0 0 0]
 [0 0 0]] shape : (2, 3)
[3.         3.04081633 3.08163265 3.12244898 3.16326531 3.20408163
 3.24489796 3.28571429 3.32653061 3.36734694 3.40816327 3.44897959
 3.48979592 3.53061224 3.57142857 3.6122449  3.65306122 3.69387755
 3.73469388 3.7755102  3.81632653 3.85714286 3.89795918 3.93877551
 3.97959184 4.02040816 4.06122449 4.10204082 4.14285714 4.18367347
 4.2244898  4.26530612 4.30612245 4.34693878 4.3877551  4.42857143
 4.46938776 4.51020408 4.55102041 4.59183673 4.63265306 4.67346939
 4.71428571 4.75510204 4.79591837 4.83673469 4.87755102 4.91836735
 4.95918367 5.        ]
[1.   1.75 2.5  3.25 4.  ]
```

In [ ]: Q6. Create an array of dimensions 3x4 having random values

In [53]:
```
r = np.random.rand(3,4)
print(r)
```

```
[[0.9919892  0.68072627 0.72258729 0.79880306]
 [0.16693437 0.5484666  0.63753367 0.85581742]
 [0.35623045 0.1399731  0.0307411  0.72834278]]
```

```
In [ ]: Q7. Create an array of dimensions 4x4 with random values and perform the following:
        a. Find maximum value from the array
        b. Find the row whose sum of values is maximum
        c. Find the column whose sum of values is maximum
        d. Find minimum value from the array
        e. Find mean value from the array
        f. Find median value from the array
        g. Find standard value from the array
        h. Find sum of the values from the array
```

```
In [60]: import numpy as np
         r = np.random.rand(4,4)
         print(r,"\n","Max:","\n",r.max(),"\n","Max row sum:","\n", r.sum(axis=1).max(),"\n",
               "Max column sum:","\n",r.sum(axis=0).max(),"\n","Min :","\n", r.min(),"\n",
               "Mean:","\n", r.mean(),"\n","Median:","\n", np.median(r),"\n","Standard value:"
               r.std(),"\n",      "Sum of all values:","\n",r.sum())
```

```
[[0.68186236 0.86963951 0.35740793 0.3023483 ]
 [0.21478367 0.3001718  0.19084794 0.68556656]
 [0.44741853 0.95195254 0.28151093 0.48137316]
 [0.52066537 0.84182055 0.74640318 0.00131779]]
 Max:
 0.9519525350568292
 Max row sum:
 2.211258106509007
 Max column sum:
 2.9635843918524953
 Min :
 0.0013177923453747686
 Mean:
 0.49219313259329717
 Median:
 0.46439584360221253
 Standard value:
 0.2693971955318487
 Sum of all values:
 7.875090121492755
```

```
In [ ]: Q8. Reshape the above dataset to dimensions 8x2,2x8,16x1
```

```
In [52]: import numpy as np
         r = np.random.rand(4,4)
         a = np.reshape(r,(8,2))
```

```
        b = np.reshape(r,(2,8))
        c = np.reshape(r,(16,1))
        print("Reshaped dim 8*2:","\n", a)
        print("Reshaped dim 2*8:","\n", b)
        print("Reshaped dim 16*1:","\n",c)
```

```
Reshaped dim 8*2:
 [[0.69497467 0.18033636]
 [0.0786883  0.59273859]
 [0.58055881 0.28030627]
 [0.10663019 0.57089343]
 [0.73744404 0.74634207]
 [0.65763002 0.29818385]
 [0.56015251 0.80355356]
 [0.12308376 0.71906822]]
Reshaped dim 2*8:
 [[0.69497467 0.18033636 0.0786883  0.59273859 0.58055881 0.28030627
  0.10663019 0.57089343]
 [0.73744404 0.74634207 0.65763002 0.29818385 0.56015251 0.80355356
  0.12308376 0.71906822]]
Reshaped dim 16*1:
 [[0.69497467]
 [0.18033636]
 [0.0786883 ]
 [0.59273859]
 [0.58055881]
 [0.28030627]
 [0.10663019]
 [0.57089343]
 [0.73744404]
 [0.74634207]
 [0.65763002]
 [0.29818385]
 [0.56015251]
 [0.80355356]
 [0.12308376]
 [0.71906822]]
```

```
In [ ]: Q9. Create an array of dimension 3x4 and perform the following:
        a. Print the first row of the array
        b. Print the first element of the row
        c. Print the element in the 2nd row and 3rd column
        d. Slice the array such that the new array has 2nd and 3rd row
        e. Slice the array such that the new array has the second element from the 2nd and 3rd
```

```
In [69]: import numpy as np
         d = np.random.rand(3,4)
```

```
print(d)
#a.
print("First row:","\n", d[0],"\n")
#b
print("First elements of the first row:","\n", d[:,0],"\n")
#c
print("Element in the 2nd row and 3rd column:","\n", d[1,2],"\n")
#d
print("2nd and 3rd row:","\n", d[1:3,],"\n")
#e
print("The second elements from the 2nd and 3rd row:","\n", d[1:3,1], "\n")
```

```
[[0.77989058 0.86467636 0.57711729 0.94171374]
 [0.04213365 0.6973578  0.33709458 0.76148757]
 [0.94068262 0.80737142 0.73743008 0.83935904]]
First row:
 [0.77989058 0.86467636 0.57711729 0.94171374]

First elements of the first row:
 [0.77989058 0.04213365 0.94068262]

Element in the 2nd row and 3rd column:
 0.3370945806918436

2nd and 3rd row:
 [[0.04213365 0.6973578  0.33709458 0.76148757]
 [0.94068262 0.80737142 0.73743008 0.83935904]]

The second elements from the 2nd and 3rd row:
 [0.6973578  0.80737142]
```