Exercise Week 8 Numpy

Part 1 User input

Exercise 1 Element-Wise Operation

1,2

In [2]:

```
numsList = list(range(2,10))
```

In [3]:

```
import numpy as np
numsArrays = np.array(numsList)
print(numsArrays)
```

```
[2 3 4 5 6 7 8 9]
```

3,4

How to add 3 to each element and substract 2

In [3]:

```
#using list comprehension on numsList
numsList = [x + 3 for x in numsList]
print(numsList)
```

```
[5, 6, 7, 8, 9, 10, 11, 12]
```

In [4]:

```
#using numpy
numsArrays += 3
print(numsArrays)
```

```
[ 5  6  7  8  9 10 11 12]
```

In [5]:

```
# here we have our list and array back to their original value
numsList = list(range(2,10))
numsArrays = np.array(numsList)

numsList = [x - 2 for x in numsList]
print(numsList)
numsArrays -= 2
print(numsArrays)
```

```
[0, 1, 2, 3, 4, 5, 6, 7]
[0 1 2 3 4 5 6 7]
```

5

In [6]:

```
x = np.linspace(1,10,10)
print(x)
```

```
[ 1.  2.  3.  4.  5.  6.  7.  8.  9. 10.]
```

linespasce takes as parameter the start, stop and number of point respontively, The function also accept other arguments for more information please refer to the documentation found here (https://numpy.org/doc/stable/reference/generated/numpy.linspace.html)

6

In [7]:

```
list(map(type,x))
```

Out[7]:

```
[numpy.float64,
 numpy.float64,
 numpy.float64,
 numpy.float64,
 numpy.float64,
 numpy.float64,
 numpy.float64,
 numpy.float64,
 numpy.float64,
 numpy.float64]
```

As mentioined above the function linespace takes multiple arguments one of which is the type, if no argument is given for the type, it type of the array is infered from the arguments

In [8]:

```
x_int64 = np.linspace(1,10,10, dtype = np.int64)
list(map(type,x_int64))
```

Out[8]:

```
[numpy.int64,
 numpy.int64,
 numpy.int64,
 numpy.int64,
 numpy.int64,
 numpy.int64,
 numpy.int64,
 numpy.int64,
 numpy.int64,
 numpy.int64]
```

7,8

In [9]:

```
x = x*2
print(x[0:5])
x = x/2
print(x[x%2 == 0])
```

```
[ 2.  4.  6.  8. 10.]
[ 2.  4.  6.  8. 10.]
```

Exercise 2- Shaping arrays

1,2,3

In [10]:

```
# using the one function we will generate  a matrix of one and multiply it by 2
y = np.ones((2,3))*2
print(y)
```

```
[[2. 2. 2.]
 [2. 2. 2.]]
```

In [11]:

```
y.reshape(3,2)
```

Out[11]:

```
array([[2., 2.],
       [2., 2.],
       [2., 2.]])
```

In [12]:

```
y.reshape(4,2)
```

```
---------------------------------------------------------------------------
-
ValueError                                Traceback (most recent call las
t)
<ipython-input-12-1df94cd01fc8> in <module>
----> 1 y.reshape(4,2)

ValueError: cannot reshape array of size 6 into shape (4,2)
```

As you can see you cannot resize an array into a larger one here the new array would have to contain 8 elements, y only has 6

4,5

Notice that this is just the sequence from 1 to 6 reshaped You could use of for a loop and access the indices to replace the values of the y array

In [13]:

```
y = np.arange(1.,7.).reshape(3,2)
print(y)
```

```
[[1. 2.]
 [3. 4.]
 [5. 6.]]
```

In [14]:

```
z = y*y
print(z)
```

```
[[ 1.  4.]
 [ 9. 16.]
 [25. 36.]]
```

This performs element-wise multiplication

6

In [15]:

```
np.dot(y.reshape(1,6),y.reshape(6,1))
```

Out[15]:

```
array([[91.]])
```

I have used the dot product on a two vectors so the solution is single integer see dot product definition (https://en.wikipedia.org/wiki/Dot_product)
Alternatively we can use the transpose method on y

In [16]:

```
print(y)
print(y.T)
```

```
[[1. 2.]
 [3. 4.]
 [5. 6.]]
[[1. 3. 5.]
 [2. 4. 6.]]
```

Part 2 Numpy Functionality

Exercise 3- Lottery Game

In [17]:

```
# Let's ask the user for an input
GuessedNumbers = np.array([])
for i in range(3):
    tempnum = input("Please your number -"+str(i))
    GuessedNumbers = np.append(GuessedNumbers,int(tempnum))
```

In [18]:

```
print(GuessedNumbers)
```

```
[ 3. 14. 17.]
```

Now we need a way to make sure the user inputs not only a valid number and is within the defined range $[1, 20]$.
To that we can use a try and except block to catch any errors

In [19]:

```
phrases = ["first","second","third"]
GuessedNumbers = np.array([])
counter = 0
while True:
    tempnum = input("Please your "+phrases[counter]+" guess")
    try:
        tempnum = int(tempnum)
    except:
        print("Please enter a valid number")
    if not (tempnum >= 1 and tempnum<=20):
        print("Please enter a number between 1 and 20")
    else:
        counter += 1
        GuessedNumbers = np.append(GuessedNumbers,tempnum)
    if counter >2:
        break
```

In [20]:

```
print(GuessedNumbers)
```

```
[13. 13. 13.]
```

The numbers above are derived from user input you can test the code for yourself and try to input a number out of range or a string to see what happens. We will now proceed to make the lottery portion of this game. To that end we will use the function random from number which allows us to generate and array of random numbers

In [25]:

```
lotterynumbers = np.random.randint(1,20,3).astype(np.float64)
print(lotterynumbers)
```

```
[19. 18. 13.]
```

We have generated our lottery number and have the user input, let's sort the two lists and compare them

In [26]:

```
if np.array_equiv(np.sort(GuessedNumbers),np.sort(lotterynumbers)):
    print("Congratulations you won")
else:
    print("Sorry try again later")
```

```
Sorry try again later
```

This program allows the user to repeat the same number for each of the guesses and randint can give you the same number twice. To aid to that we will made the user imput rule stricter

In [28]:

```
phrases = ["first","second","third"]
GuessedNumbers = np.array([])
counter = 0
while True:
    tempnum = input("Please your "+phrases[counter]+" guess")
    try:
        tempnum = int(tempnum)
    except:
        print("Please enter a valid number")
    if not (tempnum >= 1 and tempnum<=20):
        print("Please enter a number between 1 and 20")
    if tempnum in GuessedNumbers.tolist():
        print("Please enter different numbers")
    else:
        counter += 1
        GuessedNumbers = np.append(GuessedNumbers,tempnum)
    if counter >2:
        break
```

```
Please enter different numbers
```

In [8]:

```
# the numbers will be now generated using the following line of code
lotterynumbers  = np.random.choice(range(1,21),3, replace=False).astype(np.float64)
print(lotterynumbers)
```

```
[10. 15.  7.]
```

In [35]:

```
if np.array_equiv(np.sort(GuessedNumbers),np.sort(lotterynumbers)):
    print("Congratulations you won")
else:
    print("Sorry try again later")
    print("You guessed", len(set(GuessedNumbers) & set(lotterynumbers)),"numbers correc
tly")
```

```
Sorry try again later
You guessed 1 numbers correctly
```

The last print of the cell above relates to the number of guess the user got right, by transforming both list to set it is possible to find the length of the intersection

In [ ]:

```
```