Trombini Quentin Week3

S03-01

S03-01-a

```
import numpy as np

array = np.linspace(0,18,10)

print(f"array = {array}")
```

S03-01-b

```
array = [ 0.  2.  4.  6.  8. 10. 12. 14. 16. 18.]
```

S03-01-c

this code can be divided in 2 major parts:

- **import numpy as np** is importing the numpy library and create an alias so we don’t need to write numpy each time
- **np.linspace(0,18,10)** will create an array with 10 float value between 0 and 18. these value will have the same difference between each of them

---

S03-02

S03-02-a

```
import numpy as np

matrix = np.matrix([[6,7],[5,6]])
matrix = np.matrix.astype(matrix,float)

inverse = np.linalg.inv(matrix)

print(f"the inverse of \n{matrix} \nis \n{inverse}")
```

S03-02-b

```
the inverse of
[[6. 7.]
 [5. 6.]]
is
[[ 6. -7.]
 [-5.  6.]]
```

S03-02-c

```
matrix = np.matrix([[6,7],[5,6]])
matrix = np.matrix.astype(matrix,float)
```

These two line will genereate a matrix with the wanted number and then turn the number into floating number.

the next line : **np.linalg.inv(matrix)** will use the inv() function from the numpy library to compute the matrix inverse.

S03-03

S03-03-a

```
import numpy as np

rng = np.random.Generator(np.random.PCG64())

array = rng.normal(100,15,1000)

print(f"the array has a mean of {np.mean(array)} a std of {np.std(array)} for a total of {array.size} values")
```

S03-03-b

```
the array has a mean of 101.1664223386854 a std of 14.717293792750363 for a total of 1000 values
```

S03-03-c

First we create a random number generator with a function from numpy :
np.random.Generator(np.random.PCG64()) in this call the last part is to specify that we want a PCG64 type of generator.

Then we use : **rng.normal(100,15,1000)** which is a function from the random number generator to create 1000 random value with a gaussian distribution around 100 and a standard deviation around 15.