Case Study 1 – NumPy

Problem Statement:

You work in XYZ Company as a Python developer. The company officials want you to build a Python program. Tasks To Be Performed: parameters. The method should return a dictionary with 'mean' and

1. Create a function that takes dimensions as tuples e.g. (3, 3) and a numeric value and returns a NumPy array of the given dimension filled with the given value e.g.: solve((3, 3), 5) will return [[5, 5, 5], [5, 5, 5], [5, 5, 5]]
2. Create a method that takes n NumPy arrays of the same dimensions, sums them and returns the answer.
3. Given a 2 D Array of N X M Dimension, write a function that accepts this array as well as two numbers N and M. The method should return the top-left N X M sub matrix, e.g: [[1, 2, 3], [4, 5, 6], [7, 8, 9],] top_left_sub_matrix (matrix, 2, 2) -> should return: [[1, 2] [4, 5]]
4. Given a 2 D Array of N X M Dimension, write a function that accepts this array as well as two numbers N and M. The method should return the bottom-right N X M sub matrix, e.g: [[1, 2, 3], [4, 5, 6], [7, 8, 9],] sub_matrix(matrix, 1, 1) -> should return : (Keep in mind these arrays are zero indexed) [[5, 6] [8, 9]]
5. Given a 1 D NumPy Array. Write a function that accepts this array as

'std_dev' as key and array's mean and array's standard deviation as values: [1, 1, 1] solution(arr) -> should return : {'mean': 1.0, 'std_dev': 0.0}

In [2]:
```
import numpy as np
```

In [8]:
```
def f(d,n):
    return np.full(d,n)
```

In [9]:
```
f((3,3),5)
```

Out[9]:
```
array([[5, 5, 5],
       [5, 5, 5],
       [5, 5, 5]])
```

In [59]:
```
def f2(*argv):
    s = np.full(np.array(argv[0]).shape, 0)
    for i in argv:
        i = np.array(i)
        print(i)
#         s = np.sum(s, i)
        s=s+i
    return s
```

In [61]:
```
f2([1,2,3],[4,5,6],[7,8,9])
```

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

Out[61]:
```
array([12, 15, 18])
```

In [84]:
```
def f3(mat,n,m):
    return(mat[:n,:m])
```

In [85]:
```
mat = np.array([ [1, 2, 3], [4, 5, 6], [7, 8, 9], ] )
f3(mat,2,2)
```

Out[85]:
```
array([[1, 2],
       [4, 5]])
```

In [92]:
```
def f4(mat,n,m):
    return(mat[-(n+1):, -(m+1):])
```

In [93]:
```
mat = np.array([ [1, 2, 3], [4, 5, 6], [7, 8, 9], ] )
f4(mat,1,1)
```

Out[93]:
```
array([[5, 6],
       [8, 9]])
```

In [94]:
```
def f5(a):
    return({'mean':a.mean(), 'std_dev':a.std()})
```

In [95]:
```
f5(np.array([1,2,3,4,5]))
```

```
Out[95]:  {'mean': 3.0, 'std_dev': 1.4142135623730951}
```