Indexing + Fancy Indexing (1-D)

We can index a single element of a NumPy 1-D array similar to how we index a Python list.

```
# create a random numpy 1-D array
my_array = np.random.rand(10)
my_array
'Output': array([ 0.7736445 ,  0.28671796,  0.61980802,  0.42110553,
                  0.86091567,  0.93953255,  0.300224  ,  0.56579416,
                  0.58890282,   0.97219289])
# index the first element
my_array[0]
'Output': 0.77364449999999996
# index the last element
my_array[-1]
'Output': 0.97219288999999998
```

Fancy indexing in NumPy is an advanced mechanism for indexing array elements based on integers or boolean. This technique is also called *masking*.

Boolean Mask

Let's index all the even integers in the array using a boolean mask.

```
# create 10 random integers between 1 and 20
my_array = np.random.randint(1, 20, 10)
my_array
'Output': array([14,  9,  3, 19, 16,  1, 16,  5, 13,  3])
# index all even integers in the array using a boolean mask
my_array[my_array % 2 == 0]
'Output': array([14, 16, 16])
```

Observe that the code my_array % 2 == 0 outputs an array of booleans.

```
my_array % 2 == 0
'Output': array([ True, False, False, False,  True, False,  True, False,
False, False], dtype=bool)
```

Integer Mask

Let's select all elements with even indices in the array.

```
# create 10 random integers between 1 and 20
my_array = np.random.randint(1, 20, 10)
my_array
'Output': array([ 1, 18,  8, 12, 10,  2, 17,  4, 17, 17])
my_array[np.arange(1,10,2)]
'Output': array([18, 12,  2,  4, 17])
```

Remember that array indices are indexed from 0. So the second element, 18, is in index 1.

```
np.arange(1,10,2)
'Output': array([1, 3, 5, 7, 9])
```

Slicing a 1-D Array

Slicing a NumPy array is also similar to slicing a Python list.

```
my_array = np.array([14,  9,  3, 19, 16,  1, 16,  5, 13,  3])
my_array
'Output': array([14,  9,  3, 19, 16,  1, 16,  5, 13,  3])
# slice the first 2 elements
my_array[:2]
'Output': array([14,  9])
# slice the last 3 elements
my_array[-3:]
'Output': array([ 5, 13,  3])
```

Basic Math Operations on Arrays: Universal Functions

The core power of NumPy is in its highly optimized vectorized functions for various mathematical, arithmetic, and string operations. In NumPy these functions are called universal functions. We'll explore a couple of basic arithmetic with NumPy 1-D arrays.