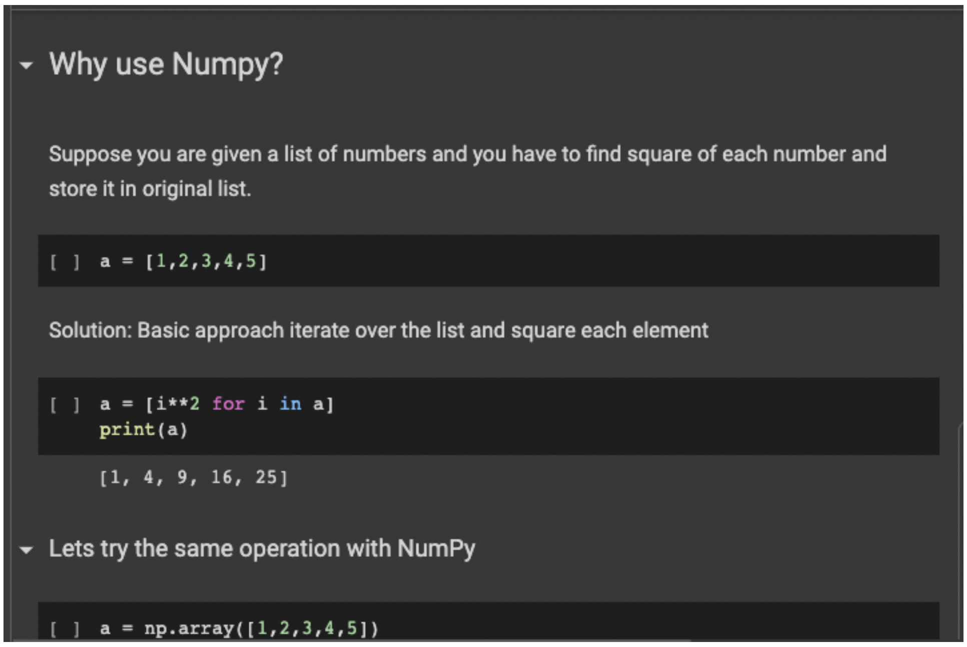
Notebook Link: https://colab.research.google.com/drive/1ltHf0RiKYEbpEpsq-HMl6IfiuHzKLbJ_?usp=sharing

Logistics

1. ~2 hrs class + 15-30 mins QnA (optional)
2. Live Lecture to start from **9:05 PM**
3. Revision till 9:15 (until we are going too slow)
4. 4-5 Quizzes per class
5. Mini break at 10PM for 5 mins (usually).
6. **Questions in the "Question Tab"** - Instructor may miss it in the chat
7. Use chat window for interaction and answering.
8. **Proper Revision Notes will be provided** - check on the dashboard for this class
9. NumPy -4 lectures



Why use Numpy?

Suppose you are given a list of numbers and you have to find square of each number and store it in original list.

```
[ ] a = [1,2,3,4,5]
```

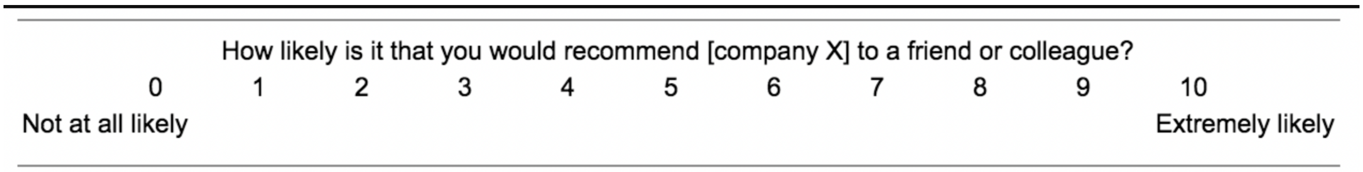
Solution: Basic approach iterate over the list and square each element

```
[ ] a = [i**2 for i in a]
    print(a)

    [1, 4, 9, 16, 25]
```

Lets try the same operation with NumPy

```
[ ] a = np.array([1,2,3,4,5])
```



How likely is it that you would recommend [company X] to a friend or colleague?

0	1	2	3	4	5	6	7	8	9	10
Not at all likely										Extremely likely

```
import numpy as np
```

```
# why?
```

```
a = [1, 2, 3, 4, 5]
a = [i*2 for i in a]
print(a)

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

```
a_np = np.array([1, 2, 3, 4, 5])
print(a_np * 2)

    [ 2  4  6  8 10]
```

```
l = range(1000000)
%timeit [i*2 for i in l]

    147 ms ± 10.3 ms per loop (mean ± std. dev. of 7 runs, 10 loops each)
```

```
l_np = np.array(range(1000000))
%timeit l_np*2
```

```
     2.29 ms ± 153 µs per loop (mean ± std. dev. of 7 runs, 100 loops each)
```

```
arr = np.array([1, 2, 3])
arr
```

```
     array([1, 2, 3])
```

```
arr.ndim
```

```
     1
```

```
arr.shape
```

```
     (3,)
```

```
np.arange(1, 5)
```

```
     array([1, 2, 3, 4])
```

```
np.arange(1, 10, 2) # np.range
```

```
     array([1, 3, 5, 7, 9])
```

```
np.arange(1, 10, 2.5)
```

```
     array([1. , 3.5, 6. , 8.5])
```

```
range(1, 10, 2.5)
```

```
     ---------------------------------------------------------------------------
     TypeError                                 Traceback (most recent call last)
     <ipython-input-18-3e909c0c53b1> in <module>
     ----> 1 range(1, 10, 2.5)

     TypeError: 'float' object cannot be interpreted as an integer
```

SEARCH STACK OVERFLOW

```
type(arr)
```

```
     numpy.ndarray
```

```
# homework: Post Reads (Optional) --> np.linspace()
```

```
np.array([1, 2, 3, 4])
```

```
     array([1, 2, 3, 4])
```

```
[1, 2, 3, 4]
```

```
     [1, 2, 3, 4]
```

```
[1, 2, 3, 4.5]
```

```
     [1, 2, 3, 4.5]
```

```
np.array([1, 2, 3, 4.5]).dtype
```

```
     dtype('float64')
```

```
[1, 2, 4.5, "Anant"]
```

```
     [1, 2, 4.5, 'Anant']
```

```
np.array([1, 2, 4.5, "Anant"], dtype="<U100")
```

```
     array(['1', '2', '4.5', 'Anant'], dtype='<U100')
```

```
np.array([1, 2, 3, 4.5], dtype="int")
```

```
     array([1, 2, 3, 4])
```

```
m1 = np.arange(12)
m1

     array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11])
```

```
m1[0]

     0
```

```
m1[-1]

     11
```

```
m1[12]

     ---------------------------------------------------------------------------
     IndexError                                Traceback (most recent call last)
     <ipython-input-33-0abd94d7097d> in <module>
     ----> 1 m1[12]

     IndexError: index 12 is out of bounds for axis 0 with size 12

      SEARCH STACK OVERFLOW
```

```
l = [ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11]
l[[0, 5, 6]]

     ---------------------------------------------------------------------------
     TypeError                                 Traceback (most recent call last)
     <ipython-input-34-01136d20fe35> in <module>
           1 l = [ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11]
     ----> 2 l[[0, 5, 6]]

     TypeError: list indices must be integers or slices, not list

      SEARCH STACK OVERFLOW
```

```
m1[[0, 5, 6]]

     array([0, 5, 6])
```

```
m1[[5, 6, 6]] # you can index an element multiple times also

     array([5, 6, 6])
```

```
m1 = np.array([100,200,300,400,500,600])
m1[[2,3,4,1,2,2]]

     array([300, 400, 500, 200, 300, 300])
```

```
m1 = np.arange(12)
m1

     array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11])
```

```
m1[:5]

     array([0, 1, 2, 3, 4])
```

```
m1[-5:-1]

     array([ 7,  8,  9, 10])
```

```
m1[-5:-1:-1]

     array([], dtype=int64)
```

```
# homework: go back and revise the syntax for reversing (start:end with neg index)
```

```
# indexing using a wrong index will lead to an error
# slicing using wrong slicing arguments will lead to an empty array
```

```
# Fancy Indexing, Boolean Masking, Boolean Indexing
```

```
m1 = np.arange(12)
m1 < 6
```

```
    array([ True,  True,  True,  True,  True,  True, False, False, False,
           False, False, False])
```

```
m1[m1 < 6] # filter
```

```
    array([0, 1, 2, 3, 4, 5])
```

```
m1[m1 % 2 == 0]
```

```
    array([ 0,  2,  4,  6,  8, 10])
```

```
a = np.arange(11)
a[(a %2 == 0) | (a%5 == 0)] # OR
```

```
    array([ 0,  2,  4,  5,  6,  8, 10])
```

```
a = np.arange(11)
a[(a %2 == 0) & (a%5 == 0)] # brackets are mandatory
```

```
    array([ 0, 10])
```

```
np.arange(4) + 3
```

```
    array([3, 4, 5, 6])
```

```
a = np.array([1, 2, 3])
b = np.array([2, 3, 4])
a + b
```

```
    array([3, 5, 7])
```

```
a = np.array([1, 2, 3])
b = np.array([2, 3, 4, 5])
a + b
```

```
    ---------------------------------------------------------------------------
    ValueError                                Traceback (most recent call last)
    <ipython-input-53-ff42284aa8d2> in <module>
          1 a = np.array([1, 2, 3])
          2 b = np.array([2, 3, 4, 5])
    ----> 3 a + b

    ValueError: operands could not be broadcast together with shapes (3,) (4,)
```

SEARCH STACK OVERFLOW

```
a = np.arange(10)
a
```

```
    array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

```
np.sum(a)
```

```
    45
```

```
np.mean(a)
```

```
    4.5
```

```
np.min(a)
```

```
    0
```

```
np.max(a)
```

```
    9
```

```
!gdown 1c0ClC8SrPwJq5rrkyMKyPn80nyHcFikK
```

```
    Downloading...
    From: https://drive.google.com/uc?id=1c0ClC8SrPwJq5rrkyMKyPn80nyHcFikK
    To: /content/survey.txt
    100% 2.55k/2.55k [00:00<00:00, 2.83MB/s]
```

```
!ls

    sample_data  survey.txt
```

```
score = np.loadtxt("survey.txt", dtype="int")
score

    array([ 7, 10,  5, ...,  5,  9, 10])
```

```
score.ndim

    1
```

```
score.shape

    (1167,)
```

```
np.min(score)

    1
```

```
np.max(score)

    10
```

```
np.mean(score)

    7.250214224507284
```

✓ 0s completed at 23:05