In [2]:

```
print("Test")
```

```
Test
```

!pip install numpy !pip install matplotlib !pip install scikit-learn !pip install panda !pip install seaborn

In [4]:

```
import numpy as np

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

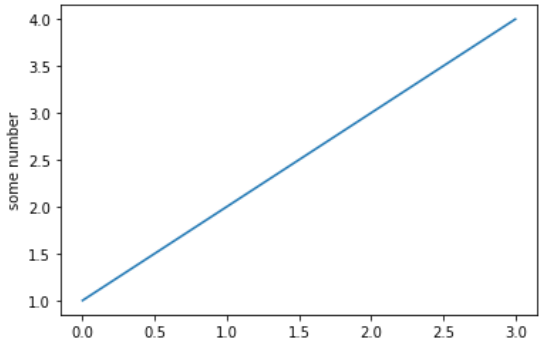
Out[4]:

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

In [7]:

```
import matplotlib.pyplot as plt

plt.plot([1,2,3,4])
plt.ylabel('some number')
plt.show()
```



Memahami Tipe Data Python

In [1]:

```
#tipe data Boolean
print(True)
#tipe data String
print("Ayo belajar Python")
print('Belajar Python Sangat Mudah')
#tipe data Integer
print(20)
#tipe data Float
print(3.14)
#tipe data Hexadecimal
print(0x10)
#tipe data Complex
print(5j)
#tipe data List
print([1,2,3,4,5])
print(["satu", "dua", "tiga"])
#tipe data Tuple
print((1,2,3,4,5))
print(("satu", "dua", "tiga"))
#tipe data Dictionary
print({"nama":"Budi", 'umur':20})
#tipe data Dictionary dimasukan ke dalam variabel biodata
biodata = {"nama":"Andi", 'umur':21} #proses inisialisasi variabel biodata
print(biodata) #proses pencetakan variabel biodata yang berisi tipe data Dictionary
#fungsi untuk mengecek jenis tipe data. akan tampil <class 'dict'> yang berarti dict adalah tipe data dictionary
print(type(biodata))
```

```
True
Ayo belajar Python
Belajar Python Sangat Mudah
20
3.14
16
5j
[1, 2, 3, 4, 5]
['satu', 'dua', 'tiga']
(1, 2, 3, 4, 5)
('satu', 'dua', 'tiga')
{'nama': 'Budi', 'umur': 20}
{'nama': 'Andi', 'umur': 21}
<class 'dict'>
```

Dasar Numpy Array

Attribute Array

In [13]:

```
import numpy as np
np.random.seed(0) # agar array selalu digenerate ulang setiap di run
x1 = np.random.randint(10, size=6) # One-dimensional array
x2 = np.random.randint(10, size=(3, 4)) # Two-dimensional array
x3 = np.random.randint(10, size=(3, 4, 5)) # Three-dimensional array

print("x3 ndim: ", x3.ndim)
print("x3 shape:", x3.shape)
print("x3 size: ", x3.size)
```

```
x3 ndim:  3
x3 shape: (3, 4, 5)
x3 size:  60
```

Apa hasil dari perintah diatas ? apa perbedaan shape dan size ?

shape

menghasilkan sebuah data yang berisikan panjang sebuah array pada setiap dimensi

size

menghasilkan jumlah perkalian elemen pada array multi-dimensi

Index Array

In [14]:

```
x1
```

Out[14]:

```
array([5, 0, 3, 3, 7, 9])
```

In [15]:

```
x1[0]
```

Out[15]:

```
5
```

In [16]:

```
x1[5]
```

Out[16]:

```
9
```

In [17]:

```
x1[6]
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
<ipython-input-17-9533a9909ead> in <module>
----> 1 x1[6]

IndexError: index 6 is out of bounds for axis 0 with size 6
```

In [18]:

```
x1[-2]
```

Out[18]:

```
7
```

Bagaimana jika mengambil index ke 6 di tuliskan x1[6], apa hasilnya ? bagaimana jika nilai index nya bernilai minus , sebagai contoh x1[-2], apa hasilnya ? apa kesimpulan ?

jika x1[6]

maka akan terjadi error dikarenakan tidak ada data pada array ke 6

jika x1[-2]

maka akan mengambil data dari array ke 2 dari belakang

In [19]:

```
x2
```

Out[19]:

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

In [30]:

```
x2[1,2]
```

Out[30]:

```
8
```

Bagaimana jika ingin mengambil nilai pada baris 2 kolom ke 3 ? silahkan di praktekkna di notebooknya

mengunakaan x2[1,2]

Slicing Array

In [31]:

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

Out[31]:

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

In [32]:

```
x[:5] # menampilkan 5 index array pertama
```

Out[32]:

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

In [33]:

```
x[5:] # menampilkan 5 index array terakhir
```

Out[33]:

```
array([5, 6, 7, 8, 9])
```

In [35]:

```
x[4:7] # menampilkan index array 4-6
```

Out[35]:

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

In [36]:

```
x[::2] # menampilkan setiap array ke 2 dari index 0
```

Out[36]:

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

In [42]:

```
x[1::2]
```

Out[42]:

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

In [43]:

```
x[::-1]
```

Out[43]:

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

In [47]:

```
x[5::-2]
```

Out[47]:

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

Jika kita mengetikkan x[1::2] apa hasilnya ? apa penjelasannya dari hal tersebut ? Bagimana jika menuliskan perintah x[::-1] apa hasilnya ? bagaimana jika x[5::-2] ?

x[1::2]

menampilkan setiap array ke 2 mulai dari index 1

x[::-1]

menampilkan setiap array dari array index 1 terakhir

x[5::-2]

menampilkan setiap array dari index 2 terakhir dan mulai dari index 5

Reshape Array

In [48]:

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

# row vector via reshape
x.reshape((1,3))
```

Out[48]:

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

In [49]:

```
# row vector via newaxis
x[np.newaxis, :]
```

Out[49]:

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

In [50]:

```
# column vector via reshape
x.reshape((3,1))
```

Out[50]:

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

In [51]:

```
# column vector via newaxis
x[:, np.newaxis]
```

Out[51]:

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

Array Concatenation

In [54]:

```
x = np.array([1,2,3])
y = np.array([3,2,1])
np.concatenate([x,y])
```

Out[54]:

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

In [55]:

```
z = [99,99,99]
print(np.concatenate([x,y,z]))
```

```
[ 1  2  3  3  2  1 99 99 99]
```

In [56]:

```
x = np.array([1,2,3])
grid = np.array([[9,8,7],
                 [6,5,4]])

# vertically stack the arrays
np.vstack([x,grid])
```

Out[56]:

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

In [57]:

```
y = np.array ([[99],
               [99]])
np.hstack([grid,y])
```

Out[57]:

```
array([[ 9,  8,  7, 99],
       [ 6,  5,  4, 99]])
```

Komputasi pada Numpy Array

In [59]:

```
x = np.arange(4)
print("x =", x)
print("x + 5 =", x+5)
print("x - 5 =", x-5)
print("x * 2 =", x*2)
print("x / 2 =", x/2)
print("x // 2 =", x//2)
```

```
x = [0 1 2 3]
x + 5 = [5 6 7 8]
x - 5 = [-5 -4 -3 -2]
x * 2 = [0 2 4 6]
x / 2 = [0.  0.5 1.  1.5]
x // 2 = [0 0 1 1]
```

Apa hasil dari penulisan kode dibawah ini :

In [60]:

```
print("-x =", -x)
print("x ** 2 =", x**2)
print("x % 2 =", x%2)
```

```
-x = [ 0 -1 -2 -3]
x ** 2 = [0 1 4 9]
x % 2 = [0 1 0 1]
```

Kesimpulan

x-1

menghasilkan negatif

x**2

hasil pangkat 2

x%2

hasil dari pembagian 2

Aggregation

In [63]:

```
big_array = np.random.rand(1000000)
big_array
```

Out[63]:

```
array([0.47331084, 0.5557414 , 0.64807403, ..., 0.39021301, 0.63082444,
       0.00358493])
```

In [64]:

```
print(big_array.min(),big_array.max(),big_array.sum(),big_array.mean())
```

```
1.4057692298008462e-06 0.9999994392723005 500240.5075662998 0.5002405075662998
```

In [65]:

```
M = np.random.random((3,4))
print(M)
M.sum()
```

```
[[0.98368561 0.38440673 0.76800784 0.81882921]
 [0.28839555 0.68014811 0.38171885 0.84190643]
 [0.74473335 0.85762176 0.81418625 0.09946649]]
```

Out[65]:

```
7.663106167360589
```

In [66]:

```
M.min(axis=0)
```

Out[66]:

```
array([0.28839555, 0.38440673, 0.38171885, 0.09946649])
```

In [67]:

```
M.max(axis=1)
```

Out[67]:

```
array([0.98368561, 0.84190643, 0.85762176])
```

Komparasi, mask dan boolean logic

In [68]:

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

Out[68]:

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

In [69]:

```
x <= 3
```

Out[69]:

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

In [70]:

```
x >= 3
```

Out[70]:

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

In [71]:

```
x != 3
```

Out[71]:

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

In [72]:

```
x == 3
```

Out[72]:

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

In [74]:

```
rng = np.random.RandomState(0)
x = rng.randint(10,size=(3,4))
x
```

Out[74]:

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

In [75]:

```
x<6
```

Out[75]:

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

Kita juga bisa melakuan operasi counting entries (menghitung jumlah), seperti untuk menjawab pertanyaan berapa jumlah yang nilainya lebih kecil dari 6 pada sebuah array ?

In [76]:

```
x < 6
```

Out[76]:

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

In [77]:

```
np.count_nonzero(x<6)
```

Out[77]:

```
8
```

In [78]:

```
# berapa jumlah yang lebih kecildari 6 di setiap baris ?
np.sum(x<6,axis=1)
```

Out[78]:

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

In [79]:

```
# apakah ada yang nilainya lebih besar dari 8 ?
np.any(x>8)
```

Out[79]:

```
True
```

In [80]:

```
# apakah ada yang nilainya lebih kecil dari 0 ?
np.any(x<0)
```

Out[80]:

```
False
```