1. Google Colab

> Jupyter notebook là một công cụ cho phép ta viết và thực thi code python một cách dễ dàng thông qua web browser.

2. Các kiểu dữ liệu cơ bản trong python

Int, Float, Bool, Tuple, Dictionary

2.1. Kiểu số: số nguyên (Integer), số thực (Float)

```
x = 4
print(x)
print(type(x))
print(x+1)
print(x-1)
print(x*2)
print(x**2)

y = 3.5
print(type(y))
print(y+1., y-1., y*2., y**2)
```

```
4
<class 'int'>
5
3
8
16
<class 'float'>
4.5 2.5 7.0 12.25
```

2.2. Kiểu dữ liệu logic (Boolean)

```
t = True
f = False
print(type(t))
print(t and f)  # logical AND
print(t or f)   # logical OR
print(not t)    # logical NOT
print(t != f)   # logical XOR
```

```
<class 'bool'>
False
True
False
True
```

2.3. Kiểu dữ liệu chuỗi (string)

```
string1 = "cai nay"
string2 = "de ma"
print(string1)
print(len(string1))
print(string1 + " " + string2)
```

```
cai nay
7
cai nay de ma
```

2.4. Containers

> Một container (vùng chứa) là một lớp, một cấu trúc dữ liệu để lưu trữ các đối tượng một cách có tổ chức. Python bao gồm 1 số loại vùng chứa như: list, dictionaries, sets, tuples.

2.4.1. Danh sách (List)

```
xs = [1, 2, 3]
print(xs, xs[1], xs[-1])
xs[1] = 'abc'
print(xs)
xs.append('cde')
print(xs)
xs.remove('abc')
print(xs)
```

```
[1, 2, 3] 2 3
[1, 'abc', 3]
[1, 'abc', 3, 'cde']
[1, 3, 'cde']
```

```
animals = ['cat', 'dog', 'monkey']

for animal in animals:
  print(animal)
```

```
cat
dog
monkey
```

2.4.2. Dictionaries

> Một dictionary lưu trữ các cặp (khóa: giá trị)

```
d = {'cat': 'cute', 'dog': 'furry'}
print(d['cat'])
print('cat' in d)
d['fish'] = 'wet'
print(d)
print(d.get('fish'))
print(d.get('monkey'))
del d['fish']
print(d.get('fish'))
```

```
cute
True
{'cat': 'cute', 'dog': 'furry', 'fish': 'wet'}
wet
None
None
```

```
d = {'bird': 2, 'cat': 4, 'spider': 8}
for animal in d:
  print('A %s has %d legs' % (animal, d[animal]))
```

```
A bird has 2 legs
A cat has 4 legs
A spider has 8 legs
```

2.4.3. Sets

> Một set được coi là một tập hợp không có thứ tự của các phần tử riêng biệt

```
set1 = {'abc', 3, 4.5, True}
print(type(set1))
print(3 in set1)
set1.add('def')
print(set1)
set1.remove('abc')
print(set1)
```

```
<class 'set'>
True
{True, 3, 4.5, 'def', 'abc'}
{True, 3, 4.5, 'def'}
```

```
animals = {'cat', 'dog', 'fish'}
for ind, animal in enumerate(animals):
  print('#%d: %s' % (ind+1, animal))
```

```
#1: cat
#2: dog
#3: fish
```

2.4.4. Tuples

> Tuple là một collection các giá trị được sắp xếp có thứ tự (bất biến)

```
t = (2, 3)
print(type(t))

d = {(x, x+1): x for x in range(10)}
print(d)
print(d[(1, 2)])
print(d[t])
```

```
<class 'tuple'>
{(0, 1): 0, (1, 2): 1, (2, 3): 2, (3, 4): 3, (4, 5): 4, (5, 6): 5, (6, 7): 6, (7, 8): 7, (8, 9): 8, (9, 10): 9}
1
2
```

2.5. Functions

> Các hàm Python được định nghĩa bằng def.

```
def first(x):
  if x > 0:
    return 'Positive'
  elif x < 0:
    return 'Negative'
  else:
    return 'Zero'
a = [-2, 0, 2]
for i in a:
  print(first(i))
```

```
Negative
Zero
Positive
```

```
def second(a, b, x):
  result = a*x + b
  return result

a = second(10, 1, 2)
print(a)
```

```
21
```

2.6. Lớp (Classes)

```
class Employee:
  empCount = 0

  def __init__(self, name, salary):
    self.name = name
    self.salary = salary
    Employee.empCount += 1

  def displayCount(self):
```

```
    print("Total Employee %d" % Employee.empCount)

  def displayEmployee(self):
    print("Name: ", self.name, ", Salary: ", self.salary)

emp1 = Employee("Zara", 2000)
emp2 = Employee("Manni", 5000)
emp1.displayEmployee()
emp2.displayEmployee()
print("Total Employee %d" % Employee.empCount)
```

```
Name:  Zara , Salary:  2000
Name:  Manni , Salary:  5000
Total Employee 2
```

3. Numpy

> Numpy là thư viện cốt lõi cho tính toán khoa học bằng Python. Nó cung cấp một đối tượng mảng đa chiều hiệu xuất cao và các công cụ để làm việc với mảng này.

3.1. Mảng (Array)

> Mảng là một lưới các giá trị, tất cả đều thuộc cùng một kiểu dữ liệu duy nhất

```
import numpy as np

a = np.array([1, 2, 3]) # Create a rank 1 array
print(type(a))          # Prints "<class 'numpy.ndarray'>"
print(a.shape)          # Prints "(3,)"
print(a[0], a[1], a[2]) # Prints "1 2 3"
a[0] = 5                # Change an element of the array
print(a)                # Prints "[5, 2, 3]"

b = np.array([[1, 2, 3], [4, 5, 6]]) # Create a rank 2 array
print(b.shape)                       # Prints "(2, 3)"
print(b[0, 0], b[0, 1], b[1, 0])     # Prints "1 2 4"
```

```
<class 'numpy.ndarray'>
(3,)
1 2 3
[5 2 3]
(2, 3)
1 2 4
```

```
import numpy as np

a = np.zeros((2, 2))
print(a)
print("-------------------------")
b = np.ones((1, 2))
print(b)
print("-------------------------")
c = np.full((2,2), 7)
print(c)
print("-------------------------")
d = np.eye(2)
print(d)
print("-------------------------")
e = np.random.random((2, 2))
print(e)
```

```
[[0. 0.]
 [0. 0.]]
-------------------------
[[1. 1.]]
-------------------------
[[7 7]
 [7 7]]
-------------------------
[[1. 0.]
 [0. 1.]]
-------------------------
```

```
[[0.60058459 0.85049575]
 [0.70993262 0.37658381]]
```

3.2. Array indexing

```
import numpy as np

a = np.array([[1, 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12]])
print(a)
print("------------------------")
b = a[:2, 1:3]
print(b)
print("------------------------")
print(a[0, 1])
print("------------------------")
b[0, 0] = 77
print(a[0, 1])
```

```
[[ 1  2  3  4]
 [ 5  6  7  8]
 [ 9 10 11 12]]
------------------------
[[2 3]
 [6 7]]
------------------------
2
------------------------
77
```

3.3. Loại dữ liệu (datatypes)

> Mọi mảng numpy là một lưới các phần tử cùng kiểu. Numpy cung cấp một tập hợp lớn các kiểu dữ liệu số mà bạn có thể sử dụng để tạo mảng. Numpy cố gắng đoán kiểu dữ liệu khi bạn tạo một mảng, nhưng các hàm tạo mảng thường cũng bao gồm một đối số tùy chọn để chỉ định rõ ràng kiểu dữ liệu.

```
import numpy as np

x = np.array([1, 2])
print(x.dtype)

x = np.array([1.0, 2.0])
print(x.dtype)

x = np.array([1, 2], dtype=np.int64)
print(x.dtype)
```

```
int64
float64
int64
```

3.4. Broadcasting

> Broadcasting là một cơ chế mạnh mẽ cho phép numpy làm việc với các mảng có hình dạng khác nhau khi thực hiện các phép toán số học.

```
import numpy as np

x = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9], [10, 11, 12]])
v = np.array([1, 0, 1])
vv = np.tile(v, (4, 1))

print(vv)
print("------------------------")
y = x + vv
print(y)
```

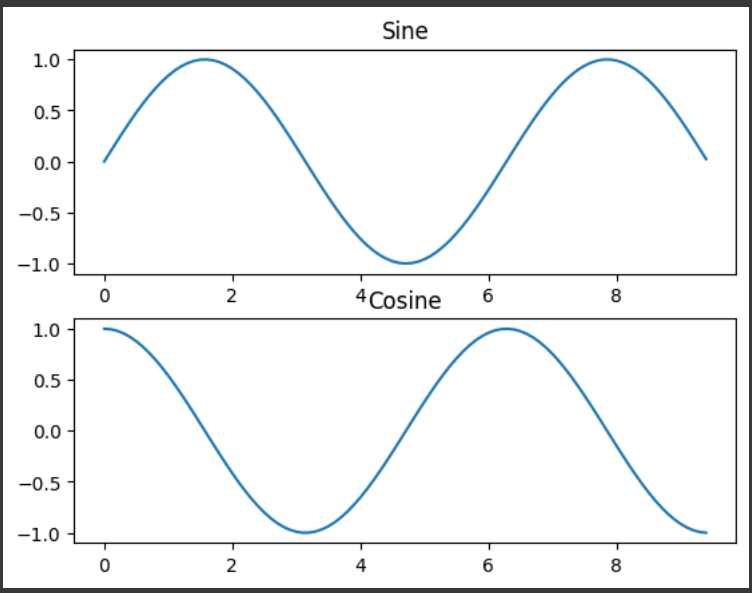
```
[[1 0 1]
 [1 0 1]
 [1 0 1]
```

```
 [1 0 1]]
------------------------
[[ 2  2  4]
 [ 5  5  7]
 [ 8  8 10]
 [11 11 13]]
```

4. Matplotlib

> Matplotlib là một thư viện toàn diện để tạo hình ảnh trực quan tĩnh, động và tương tác bằng Python. Matplotlib biến những điều dễ dàng trở nên dễ dàng và những điều khó khăn trở thành hiện thực.

```
import numpy as np
import matplotlib.pyplot as plt

# Compute the x and y coordinates for points on sine and cosine curves
x = np.arange(0, 3 * np.pi, 0.1)
y_sin = np.sin(x)
y_cos = np.cos(x)

# Set up a subplot grid that has height and width 1.
# and set the first such subplot as active.
plt.subplot(2, 1, 1)

# Make the first plot
plt.plot(x, y_sin)
plt.title('Sine')

# Set the second subplot as active, and make the second plot.
plt.subplot(2, 1, 2)
plt.plot(x, y_cos)
plt.title('Cosine')

# Show the figure
plt.show()
```



4.1. Matplotlib with Opencv

```
# Import Library
import cv2
from matplotlib import pyplot as plt
```

```
# Read image as cv2.imread()
img = cv2.imread('phenika.jpeg')
plt.imshow(img)
```

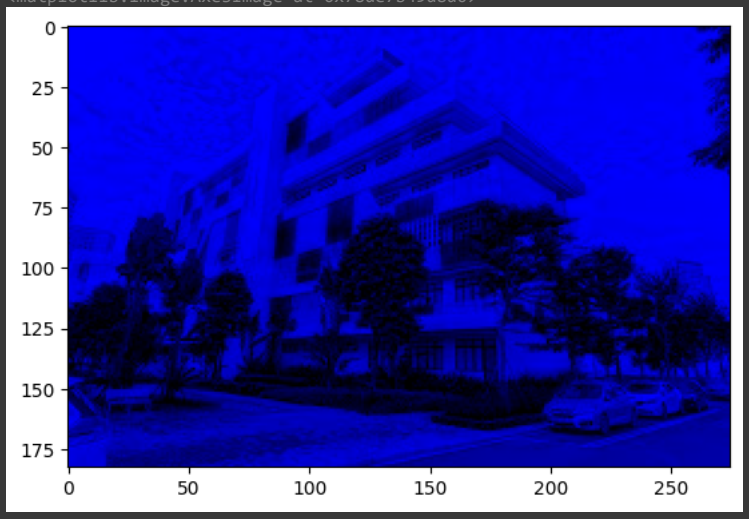
```
<matplotlib.image.AxesImage at 0x78de7b7fe9b0>
```

```
# Convert Image from BGR to RGB
img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
plt.imshow(img)
```

```
<matplotlib.image.AxesImage at 0x78de86a8d5d0>
```

```
# Take a color channel from input image (EX: blue)
blue = img.copy()
blue[:, :, [0, 1]] = 0
plt.imshow(blue)
```

```
<matplotlib.image.AxesImage at 0x78de7b49d8d0>
```



```
# Show shape
print(img.shape)
```

```
(183, 275, 3)
```

```
# Show intensity in a pixel
print(img[0, 0])
```

```
[254 254 254]
```

```
# Convert image from RGB to GRAY
gray_img = cv2.cvtColor(img, cv2.COLOR_RGB2GRAY)
plt.imshow(gray_img, cmap='gray')
```

```
<matplotlib.image.AxesImage at 0x78de7b597160>
```


```
# Resize image
new_img = cv2.resize(img, (300, 150))
plt.imshow(new_img)
```

```
<matplotlib.image.AxesImage at 0x78de7b5e95a0>
```